Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«Омский государственный технический университет (ОмГТУ)»

На правах рукописи

Сулавко Алексей Евгеньевич

ВЫСОКОНАДЕЖНАЯ БИОМЕТРИЧЕСКАЯ АУТЕНТИФИКАЦИЯ НА ОСНОВЕ ЗАЩИЩЕННОГО ИСПОЛНЕНИЯ НЕЙРОСЕТЕВЫХ МОДЕЛЕЙ И АЛГОРИТМОВ ИСКУССТВЕННОГО ИНТЕЛЛЕКТА

Специальность 2.3.6
Методы и системы защиты информации, информационная безопасность

ДИССЕРТАЦИЯ
на соискание ученой степени доктора технических наук

Научный консультант:
доктор технических наук, доцент
Ложников Павел Сергеевич

Омск – 2023

Оглавление

Введение

Актуальность темы исследования. Сегодня мировой рынок биометрии проходит фазу активного роста (по данным MarketsAndMarkets к 2025 г. его объем составит 68 млрд. $). Биометрические системы внедряются повсеместно: на объектах критической информационной инфраструктуры, в банковской сфере, государственном секторе (более 80 стран используют биометрические паспорта), в сфере управления транспортом и городом. Рост рынка биометрических систем обусловлен новыми тенденциями и вызовами, с которыми столкнулось общество и государство: увеличение объемов данных о действиях пользователей в сети Интернет, которые могут быть использованы в злоумышленных целях (обострились проблемы приватности, анонимности пользователей и защищенности биометрических шаблонов от компрометации); применение технологий искусственного интеллекта (ИИ) для реализации хакерских атак, дезинформации, мошенничества, фальсификации биометрических образов человека (например, при помощи deepfake, голосовых синтезаторов); замена традиционных биометрических образов отпечатка пальца на более удобные образы голоса, лица и др., пригодные для бесконтактной аутентификации, но в большей степени подверженные дрейфу (изменчивости). В связи с этим современная высоконадежная биометрическая система должна строиться на основе доверенного ИИ, устойчивого к деструктивным факторам (дрейф биометрических данных, компьютерные атаки) и обладающего поддержкой защищенного режима исполнения. Под «защищенным исполнением» понимается невозможность анализа логики работы ИИ, управления ИИ и извлечения знаний из памяти ИИ любым неавторизованным субъектом.

Настоящее диссертационное исследование посвящено решению **научно-технической проблемы**, которая заключается в повышении надежности многофакторной биометрической аутентификации и защищенности биометрических систем от компьютерных атак на основе технологии

защищенного исполнения нейросетевых моделей и алгоритмов искусственного интеллекта.

Степень разработанности темы исследования. На данный момент действует ряд международных стандартов, связанных с вопросами защиты биометрических систем от компьютерных атак (ISO/IEC 19792:2009, ISO/IEC 24761:2019, ISO/IEC 24745:2022, ISO/IEC 30107). Однако эти стандарты не позволяют устранить ряд актуальных угроз (извлечение знаний моделей ИИ, компрометация открытых биометрических образов, состязательные атаки). В России действует серия национальных стандартов ГОСТ Р 52633, не имеющих международных аналогов. Стандарты ГОСТ Р 52633 регламентируют особенности разработки, обучения и тестирования систем высоконадежной биометрической аутентификации, которые должны строиться на базе нейросетевых преобразователей биометрия-код (НПБК), позволяющих связать криптографический ключ или пароль пользователя с его биометрическим образом. Тем не менее, из-за наличия ряда недостатков применимость данных стандартов ограничена (высокая вероятность ошибок, малая длина ключа, подверженность атакам).

В мировой практике сложилось несколько подходов к повышению надежности биометрических систем аутентификации с обеспечением конфиденциальности биометрических данных, которые основаны на использовании нечетких экстракторов, искусственных нейронных сетей, искусственных иммунных систем, применении шифрования (в том числе, гомоморфного). Развитию аппарата искусственных нейронных сетей и искусственных иммунных систем, а также вопросам создания доверенного ИИ посвящены работы многих ведущих российских и зарубежных ученых. Среди них Брюхомицкий Ю.А., Вульфин А.М., Гарбук С.В., Галушкин А.И., Иванов А.И., Котенко И.В., Николенко С. И., Baker B., Bengio Y., De Castro L. N., Fung C., Greensmith J., Hinton G.E., Kurkova V., LeCun Y., Mishra P.K., Schapire R.E., Stanley K.O., Timmis J. и другие. Вопросам высоконадежной биометрической аутентификации, оценки изменчивости биометрических параметров, обеспечения

конфиденциальности биометрических данных, а также защите биометрических систем от компьютерных атак посвятили множество своих работ Ахметов Б.С., Бабенко Л.К., Безяев А.В., Васильев В.И, Волчихин В.И., Епифанцев Б.Н., Еременко А. В., Иванов А.И., Катасёв А.С., Ложников П.С., Маршалко Г.Б., Akkermans T.H., Catak F.O., Dodis Y., Hao F., Hafemann L.G., Jain A.K., Kumar A., Maiorana E., Mulionoa Y., Roy N.D., Wang L., Yuan L. и другие. Анализ этих работ позволил определиться с направлением диссертационного исследования и выявить перспективные подходы к решению обозначенной научно-технической проблемы. Эти подходы связаны с разработкой концепции защищенного исполнения нейросетевых алгоритмов ИИ, моделей искусственных нейронов и НПБК на их основе, изначально устойчивых к деструктивным воздействиям и атакам, адаптивных моделей ИИ, способных подстраиваться под изменяющиеся данные, снижая влияние концептуального дрейфа в задачах высоконадежной биометрической аутентификации, а также алгоритмов их обучения. Из проведенного анализа следует, что на основе предложенных концепции, моделей и алгоритмов необходимо разработать методы, технологию и программный комплекс для создания систем высоконадежной многофакторной биометрической аутентификации с обеспечением защиты биометрических данных от компрометации.

Объект исследования: системы биометрической аутентификации на основе методов, моделей и алгоритмов доверенного ИИ.

Предмет исследования: нейросетевые модели и алгоритмы машинного обучения на малых выборках для высоконадежной биометрической аутентификации и защиты биометрических данных от компрометации.

Цель диссертационной работы: повысить надежность многофакторной биометрической аутентификации на основе защищенного исполнения нейросетевых моделей доверенного ИИ и алгоритмов их автоматического синтеза и обучения на малых выборках биометрических данных.

Для достижения цели были выполнены следующие **задачи**:

1. Разработка концепции защищенного исполнения нейросетевых алгоритмов ИИ.
2. Разработка моделей искусственных нейронов и нейросетевого преобразователя биометрия-код, потенциально устойчивых к деструктивным воздействиям, и алгоритмов их робастного автоматического обучения на малых выборках.
3. Разработка адаптивной модели ИИ и алгоритмов ее обучения, позволяющих предупредить или снизить влияние концептуального дрейфа данных в системах биометрической аутентификации.
4. Разработка методов многофакторной аутентификации на базе тайных биометрических образов с обеспечением конфиденциальности биометрических данных.
5. Разработка технологии автоматического синтеза и обучения нейросетевых моделей для высоконадежной многофакторной биометрической аутентификации.

Основные результаты, выносимые на защиту:

1. Концепция защищенного исполнения нейросетевых алгоритмов ИИ, основанная на преобразовании корреляционных связей между признаками в мета-признаки, позволяющая снизить количество ошибок классификации образов и повысить защищенность систем ИИ от извлечения знаний.
2. Модель корреляционных нейронов и модель нейросетевого преобразователя биометрия-код на их основе, а также алгоритм их автоматического синтеза и обучения на малых выборках, которые позволяют повысить длину ключа, связываемого с биометрическими образами субъектов, и устойчивость биометрических систем к состязательным атакам и извлечению знаний.
3. Адаптивная нейро-иммунная модель ИИ и алгоритмы ее обучения с учителем и с подкреплением, позволяющие предупредить или снизить влияние концептуального дрейфа в системах биометрической аутентификации.

4. Методы и алгоритм высоконадежной многофакторной аутентификации на основе рукописных и голосовых образов, а также акустических образов уха с обеспечением защиты биометрических данных от компрометации, позволяющие повысить защищенность информации от неавторизованного доступа.
5. Технология автоматического синтеза и обучения нейросетевых моделей доверенного ИИ на малых выборках, а также программный комплекс на ее основе, позволяющие создавать системы высоконадежной биометрической аутентификации и другие ответственные приложения ИИ, обладающие повышенной устойчивостью к деструктивным воздействиям.

Научная новизна результатов:

1. Предложена концепция защищенного исполнения нейросетевых алгоритмов ИИ, *позволяющая* обеспечить устойчивость моделей и алгоритмов ИИ к извлечению знаний в задачах классификации образов, которая *в отличие* от существовавших ранее концепций основана на преобразовании корреляционных связей между признаками в высокоинформативные мета-признаки Байеса-Минковского с помощью предложенного для этой цели отображения. Экспериментально установлено, что корреляция между признаками увеличивает количество информации об образе (один мета-признак Байеса-Минковского может содержать в 2-3 раза больше информации, чем содержится суммарно в паре исходных признаков, от которых он порожден), что повышает надежность распознавания образов.
2. Разработаны модель корреляционных нейронов и модель НПБК на их основе, *отличающиеся* тем, что они анализируют корреляционные связи между признаками вместо признаков, а также робастной алгоритм автоматического синтеза и обучения этих моделей на малых выборках, что *позволяет* повысить защищенность биометрических данных от компрометации, длину ключа, связываемого с биометрическими образами субъектов, и устойчивость систем биометрической аутентификации к состязательным атакам.

3. Разработана адаптивная нейро-иммунная модель ИИ, *отличающаяся* от существовавших ранее использованием предложенной гибкой архитектуры искусственных иммунных детекторов (антител и клеток памяти), использованием в основе детекторов ядерных функций, сочетанием ансамблевых методов машинного обучения и метода обучения с подкреплением, что *позволяет* ей устойчиво обучаться на малых выборках и адаптироваться к изменению биометрических данных в процессе функционирования. Предложенные нейро-иммунная модель и алгоритмы ее обучения *в отличие* от существовавших ранее *позволяют* снизить влияние концептуального дрейфа и вероятность ошибок биометрической аутентификации, даже если исходная обучающая выборка недостаточно репрезентативна или незначительна в объеме.

4. Разработаны методы и алгоритм высоконадежной многофакторной биометрической аутентификации, *отличающиеся* использованием новых акустических биометрических параметров, характеризующих внутреннее строение ушного канала, комплексированием динамических и статических признаков с учетом их приоритезации, информативности и стабильности, а также совместным использованием НПБК и нейро-иммунной модели, что *позволяет* обеспечить более высокую надежность аутентификации, робастность дрейфующих характеристик, защиту биометрических образов от компрометации, снизить вероятность ошибок «ложного допуска» и «ложного отказа» по сравнению с известными аналогами.

5. Разработана технология синтеза нейросетевых моделей доверенного ИИ, которая *позволяет* снизить объем тренировочной выборки, повысить надежность и защищенность биометрических систем аутентификации и других приложений ИИ, *отличающаяся* наличием режимов автоматического обучения нейросетевых моделей ИИ, защищенного исполнения нейросетевых алгоритмов классификации образов и применением процедур автоматической оценки информативности признаков.

Теоретическая значимость работы заключается в предложенной концепции, моделях и алгоритмах обучения. В совокупности они образуют

математический аппарат, позволяющий создавать нейросетевой ИИ, который будет устойчив к различным деструктивным воздействиям на уровне архитектуры. Хотя в настоящей работе в качестве ключевой научной задачи выбрана задача высоконадежной многофакторной биометрической аутентификации, предложенный аппарат может применяться в других приложениях ИИ, для которых актуальны вопросы обеспечения защиты от компьютерных атак, извлечения знаний и обучения/дообучения на малых выборках. Решены важнейшие задачи автоматизации машинного обучения с использованием малых выборок биометрических данных и онлайн-обучения нейросетевых моделей (обучения модели в процессе ее исполнения в реальной практике). Это позволяет снизить негативное влияние таких факторов, как дрейф биометрических данных, а также в некоторых случаях успешно обучать модели, даже если биометрических данных мало, а тренировочная выборка недостаточна репрезентативна. Полученные результаты вносят значительный вклад в теорию машинного обучения, так как впервые предлагается использовать корреляционные связи между признаками в качестве новых мета-признаков и дается количественная оценка информативности этих мета-признаков.

Практическая значимость работы. На базе предложенной технологии синтеза нейросетевых моделей ИИ под руководством соискателя на базе ОмГТУ разработана первая редакция государственного национального стандарта ГОСТ Р «Искусственный интеллект. Нейросетевые алгоритмы в защищённом исполнении. Автоматическое обучение нейросетевых моделей на малых выборках в задачах классификации». Это первый стандарт, который регламентирует особенности создания и обучения нейросетевых моделей ИИ, исполняемых в защищенном от исследования режиме. Стандарт направлен на использование на объектах критической информационной инфраструктуры при разработке ответственных приложений ИИ. Стандарт прошел экспертизу технических комитетов Росстандарта и включен в программу стандартизации технического комитета «Искусственный интеллект» (ТК164).

Результаты работы легли в основу линейки программных продуктов AIConstructor (AIC), научным руководителем разработки которых является Сулавко А.Е. AIC desktop – программный комплекс для проведения научных исследований по машинному обучению. AIC ModelOps Platform – корпоративная среда управления жизненным циклом ИИ, может использоваться для автоматизации, отслеживания и контроля рабочих процессов на всех этапах: от исследования до внедрения в бизнес среду.

Практическую значимость представляют методы высоконадежной многофакторной биометрической аутентификации по особенностям ушного канала, рукописным и голосовым образам с показателями FRR=0,12 при $FAR<10^{-14}$ и FRR=0,03 при $FAR<10^{-10}$ и программные продукты на их основе.

Методы исследования. Применялись методы распознавания образов, машинного обучения, кодирования информации и защиты данных от компрометации, аппарат искусственных нейронных сетей (ИНС), ансамблевые методы, биоинспирированные алгоритмы и модели классификации образов, методы теории вероятностей и математической статистики, спектрального и корреляционного анализа, обеспечения дифференциальной конфиденциальности данных и знаний, идентификации и аутентификации.

Достоверность полученных результатов обусловлена корректным применением методов исследования, использованием признанных методик статистической обработки данных, математически строгим выполнением расчетов и подтверждается результатами практического использования и актами внедрения. Вводимые допущения мотивировались фактами, известными из практики. Предложенные в работе концепция, модели, методы и алгоритмы теоретически обоснованы и не противоречат известным достоверно подтвержденным результатам исследований других авторов.

Реализация и внедрение результатов работы. Результаты работы внедрены на предприятиях: ООО «Открытый код», ООО «Системы информационной безопасности», ООО «АИ ЗИОН», ООО «Джемс Девелопмент», БУЗОО «Медико-санитарная часть № 4», где они использовались в проектно-

конструкторской деятельности, и в учебный процесс ФГАОУ ВО СПбГЭТУ «ЛЭТИ» и ФГАОУ ВО «ОмГТУ». **Результаты применялись при разработке первой редакции национального стандарта** ГОСТ Р «Искусственный интеллект. Нейросетевые алгоритмы в защищённом исполнении. Автоматическое обучение нейросетевых моделей на малых выборках в задачах классификации» под руководством соискателя на базе ОмГТУ, который поставлен в план стандартизации техническим комитетом № 164 «Искусственный интеллект».

Результаты работы связаны с научными программами, руководителем которых являлся соискатель: государственное задание Минобрнауки России на 2023-2025 годы № FSGF-2023-0004, Грант ИБ №6 от МИРЭА и Минобрнауки РФ; Гранты РНФ 17-71-10094 «Разработка технологии широких нейронных сетей сверхбыстрого обучения и ее применение для надежной аутентификации субъектов на основе тайных биометрических образов», РФФИ 18-41-550002, РФФИ 16-37-50005; НИР «Защищенный режим исполнения искусственного интеллекта на базе автоматически обучаемых сетей автокорреляционных нейронов»; Грант Фонда Содействия Инновациям на проведение НИОКР по теме «Разработка ModelOps платформы для оптимизации процесса цифровой трансформации при создании и внедрении доверенного искусственного интеллекта с использованием сетей корреляционных нейронов»; НИР «Способы распознавания субъектов и их психофизиологического состояния по динамическим биометрическим признакам», НИР «Гибкие нейросетевые алгоритмы для анализа биометрических образов», НИР «Защита информационных и компьютерных систем на базе предиктивного анализа биометрических и поведенческих характеристик оператора».

Также соискатель участвовал в статусе исполнителя в Госзадании 2.9314.2017/БЧ и следующих проектах РФФИ: 13-07-00246, 15-07-09053, 16-07-01204, 18-37-00399, 15-37-50269, 16-37-50045.

Апробация результатов. Результаты работы регулярно докладывались и обсуждались на научных конференциях: Международная IEEE научно-техническая конференция «Динамика систем, механизмов и машин», г.Омск

(2014, 2016, 2017, 2018); Научно-практическая конференция «Безопасность информационных технологий», г.Пенза (2014, 2016, 2020); Международная IEEE Сибирская конференция по управлению и связи SIBCON, г.Омск, 2015, г.Москва, 2016, г.Астана, 2017; Международная конференция «Аппроксимация логических моделей, алгоритмов и задач», г.Омск, 2015; IEEE Международная конференция по использованию информационно-коммуникационных технологий г.Баку, Азербайджан, 2016; Международная научно-практическая конференция «Научно-технический прогресс: актуальные и перспективные направления будущего», г.Кемерово, 2016; Международная научно-практическая конференция «Инфографика и информационный дизайн: визуализация данных в науке», г.Омск, 2017; IFAC Conference on Technology, Culture and International Stability (TECIS), г.Баку, Азербайджан, 2018, г.Созополь, Болгария, 2019; Межвузовская научно-практическая конференция «Информационная безопасность: современная теория и практика», г.Омск (2018, 2019, 2020); Международная научно-техническая конференция «Актуальные проблемы электронного приборостроения (АПЭП)», г.Новосибирск, 2018; Всероссийская научно-практическая конференция с международным участием им. В.В.Губарева «Интеллектуальный анализ сигналов, данных и знаний: методы и средства», г.Новосибирск, 2018; Международная научно-техническая конференция «Проблемы машиноведения», г.Омск (2018, 2019, 2020); Международная научно-практическая конференция «Цифровизация и кибербезопасность: современная теория и практика», г.Омск, 2021.

Соответствие паспорту специальности. Результаты диссертационной работы соответствуют следующим пунктам паспорта научной специальности 2.3.6 – Методы и системы защиты информации, информационная безопасность: п. 9. «Модели противодействия угрозам нарушения информационной безопасности для любого вида информационных систем, позволяющие получать оценки показателей информационной безопасности»; п. 12. «Технологии идентификации и аутентификации пользователей и субъектов информационных процессов. Системы разграничения доступа»; п. 15. «Принципы и решения (технические,

математические, организационные и др.) по созданию новых и совершенствованию существующих средств защиты информации и обеспечения информационной безопасности».

Публикации. Соискателем опубликовано 80 работ, содержащих результаты диссертационного исследования, в том числе 38 статей в журналах из Перечня рецензируемых научных изданий, рекомендованных ВАК, либо в научных изданиях, индексируемых базой RSCI, 21 научная работа в изданиях, включенных в базы Web of Science и Scopus, 11 научных работ в других изданиях и 1 коллективная монография. Получен 1 патент на изобретение и 8 свидетельств о регистрации программ.

Структура и объём диссертации. Диссертация состоит из введения, 5 глав, заключения, списка сокращений, списка литературы и приложений. Диссертация содержит 391 страница машинописного текста, включая 108 рисунков, 28 таблиц, список литературы из 362 наименования.

Личный вклад автора состоит в постановке задач исследования, разработке экспериментальных и теоретических методов, разработке, тестировании и реализации предложенных концепции, моделей, методов, алгоритмов и компьютерных программ, анализе и обобщении полученных результатов и формулировке выводов. **Все результаты и положения, выносимые на защиту, а также научная новизна получены лично автором.** Подготовка к публикации некоторых результатов проводилась совместно с соавторами, но вклад диссертанта был определяющим. Участие научного консультанта заключалось в оказании методической и организационной помощи в формулировании задач, представлении результатов и оценке их корректности.

1 Аналитическое исследование проблемы защиты искусственного интеллекта от деструктивных воздействий. Задачи исследований

Обозначим ключевые термины исследования из ГОСТ Р 52633.0-2006 «Требования к средствам высоконадежной биометрической аутентификации»:

– высоконадежная биометрическая аутентификация – биометрическая аутентификация с приемлемой вероятностью ошибок первого рода и гарантированно малой вероятностью ошибок второго рода, сопоставимой по своему значению с вероятностью случайного подбора кода неизвестного криптографического ключа при малом числе попыток подбора. Также в ГОСТ Р 52633.0 прописано, что система биометрической аутентификации является высоконадежной, если показатель вероятности ошибки «ложного допуска» составляет менее 10^{-12}. Под надежностью же понимается способность биометрической системы сохранять эксплоатационные характеристики в изменяющихся условиях функционирования.

– биометрические данные – данные с выходов первичных измерительных преобразователей физических величин, совокупность которых образует *биометрический образ* конкретного человека.

– биометрический образ – образ человека, полученный с выходов первичных измерительных преобразователей физических величин, подвергающийся далее масштабированию и иной первичной обработке с целью извлечения из него контролируемых *биометрических параметров* человека.

– биометрический образ «Свой» – биометрический образ легального пользователя.

– биометрический образ «Чужой» – биометрический образ злоумышленника, пытающегося преодолеть биометрическую защиту.

– динамический биометрический образ – биометрический образ, изменяемый человеком по своему желанию, например рукописный образ слова-пароля.

– статический биометрический образ – образ, данный человеку от рождения, неизменяемый по воле человека, например рисунок отпечатка пальца.

– тайный биометрический образ – биометрический образ, сохраняемый пользователем в тайне.

– открытый биометрический образ – биометрический образ человека, общедоступный для наблюдения.

– биометрические параметры – параметры, полученные после предварительной обработки биометрических данных.

Отметим, что термин «признак» или «биометрический признак» в ГОСТ Р 52633.0 не фигурирует. Однако такое понятие фигурирует в ГОСТ Р ИСО/МЭК 19795-1 «Идентификация биометрическая. Эксплуатационные испытания и протоколы испытаний в биометрии. Часть 1», базирующегося на международном стандарте ISO/IEC 19795-1:2006:

– признаки (features) – цифровое представление информации, извлеченной из образца (подсистемой обработки сигналов) и используемой для создания шаблонов или сравнения с зарегистрированными в базе данных шаблонами;

– шаблон/модель (template/model) – информация, предназначенная для сохранения, полученная из биометрических характеристик пользователя на основе признаков, извлекаемых из образцов;

– образец (sample) – биометрическая характеристика пользователя, получаемая на выходе подсистемы сбора данных, предназначенных для регистрации.

Таким образом, понятия *биометрический параметр* и *признак* можно обозначить, как синонимы. Понятие «биометрический образ» дает более широкую трактовку, чем понятие «образец» и может пониматься, как вектор признаков, так и как данные, которые предшествовали первичной обработке (извлечению признаков). Отметим, что понятия образа и признака являются ключевыми в теории распознавания образов и также активно используются в специализированной литературе по биометрии вместо аналогичных понятий

«биометрический параметр» и «образец». Поэтому в настоящем исследовании мы будем преимущественно оперировать терминами «признак» и «образ» и реже терминами «биометрический параметр» и «образец».

Под «биометрическими» будем понимать любые характеристики человека, которые обладают уникальностью и могут быть использованы для его аутентификации или идентификации.

1.1 Актуальность проблемы повышения надежности и защищенности биометрических систем и данных

По данным McAfee за 2020 год совокупные потери мировой экономики от киберугроз, хищения и разрешения информации составили $945 млрд. (в 2018 году эта оценка была равной $522 млрд., а в 2014 – $475 млрд.) [323]. Согласно другой статистике, представленной Juniper Research, убытки от хищения и разрешения конфиденциальной информации в 2018 году в мире уже достигли $3 трлн., при этом сообщалось, что преступники активно используют технологии искусственного интеллекта (ИИ) [259]. К 2024 году прогнозируется, что эта цифра вырастет до $5 трлн. [259]. Аналитические исследования InfoWatch показали, что за последние три года возросло количество умышленных утечек персональных данных и коммерческой тайны, увеличилась доля сетевого канала, снизилась роль бумажных документов [82], что обострило проблему удаленной аутентификации.

Сделать аутентификаторы неотчуждаемыми от личности субъекта становится возможным с помощью систем высоконадежной биометрической аутентификации, которые во многих странах применяются в банковской сфере и государственном секторе (России, США, Канаде, Бразилии, Великобритании, Франции, Китае, Германии, Японии, Саудовской Аравии, Израиле, Египте, Африке и других). Производители мобильных устройств и операционных систем массово переводят свои продукты на использование биометрических методов контроля доступа, мотивируя это удобством и более высоким уровнем

защищенности от угроз, обусловленных человеческим фактором (по заявлению Microsoft, 70% пользователей Windows 10 уже используют биометрию, считая парольную защиту устаревшей технологией [109]).

В соответствии с российским законодательством биометрические образы являются персональными данными, которые нуждаются в надежной защите от компрометации согласно федеральным законам № 152 и № 572. Из-за уникальных свойств биометрические данные стали ценным товаром для мошенников. Овладев биометрией пользователя, хакер может получить доступ ко всем личным кабинетам, которые связаны с его скомпрометированным биометрическим шаблоном. В связи с этим доверие к биометрическим системам определяется не только количеством ошибок «ложного отказа» (False Reject Rate, FRR) и «ложного допуска» (False Access Rate, FAR), но и другими факторами:

– устойчивостью к предъявлению подделок (цифровых или физических «муляжей» биометрических образов) и состязательным атакам;
– возможностью обеспечения конфиденциальности биометрических данных пользователей при хранении, передаче по каналам связи;
– возможностью сокрытия биометрического образа от постороннего наблюдения.

Случаи хищения и взлома чужой биометрии все чаще становятся достоянием общественности. Печально известным примером массовой компрометации биометрических шаблонов является утечка в системе UIDAI в Индии (база биометрических данных UIDAI является самой крупной в мире и насчитывает более 1,1 млрд. пользователей, каждому из которых присевается идентификационный код AADHAAR). В мае 2017 г. было скомпрометировано более 135 млн. учетных записей [87], а в 2018 г. хранилище было скомпрометировано полностью [87]. В качестве другого примера можно привести утечку более миллиона отпечатков пальцев в августе 2019 года в Британии [164]. Эксперты «Лаборатории Касперского» прогнозируют, что в будущем число утечек биометрических персональных данных значительно возрастет, поскольку технологии биометрии активно внедряются в разные сферы деятельности [163].

Биометрические системы можно рассматривать как частный случай систем ИИ, которые сами по себе являются объектом для компьютерных атак. Утечки знаний ИИ создают не меньше проблемных ситуаций, чем утечки иной информации, так как знания ИИ могут содержать аналогичные сведения (в том числе, биометрические персональные данные). Оценить масштабы потенциальных потерь от утечек знаний ИИ и атак на ИИ численно пока сложно, тем не менее, однозначно можно сказать, что они внушительны. В соответствии с результатами исследований компании InfoWatch в 2021 году средняя стоимость одной утечки конфиденциальных данных в мире составила $4,24 млн. [137]. Актуальность защиты знаний ИИ высока, что подтверждается стремлением мирового сообщества разработать стандарты в этой области.

1.2 Проблемы и свойства доверенного искусственного интеллекта применительно к системам высоконадежной биометрической аутентификации

Доверенный искусственный интеллект – одно из важнейших понятий в области машинного обучения. Данный термин обычно понимается в достаточно широком смысле и используется в ряде международных и российских документов. В соответствии с ГОСТ Р 59276-2020 Системы искусственного интеллекта. Способы обеспечения доверия. Общие положения: «Доверенная система искусственного интеллекта: Система искусственного интеллекта, в отношении которой потребитель и, при необходимости, организации, ответственные за регулирование вопросов создания и применения систем искусственного интеллекта, проявляют доверие». Доверенный ИИ отвечает таким критериям, как объяснимость, прозрачность, робастность и безопасность. Для биометрических систем наиболее важными являются последние два свойства.

В ГОСТ Р 59276-2020 под объяснимостью понимается «свойство системы искусственного интеллекта, заключающееся в возможности представления

причин, приводящих к тому или иному решению системы, в виде, понятном человеку». Объяснимость позволяет ИИ принимать решения, кажущиеся для человека обоснованными и понятными. Выделяют три вида объяснимости:

– объяснимость причинно-следственных связей решения обеспечивается, если можно примерно проследить цепочку выводов, которые привели к решению;
– объяснимость на уровне данных обеспечивается, если можно заранее проверить, насколько выборка сбалансирована, чтобы понять, не приведет ли обучение или тестирование на такой выборке к предвзятым или неэтичным решениям и результатам;
– объяснимость путем оценки информативности признаков позволяет проследить, какой вклад дает каждый признак при принятии решения.

Близким по смыслу термином является прозрачность – свойство ИИ, согласно которому важная и необходимая информация о системе ИИ (параметрах работы ИИ, данных и решениях) передается заинтересованным лицам, не компрометируя информацию, к которой эти лица доступа не имеют. В ГОСТ Р 59276-2020 аналогом понятия прозрачности является понятность – «свойство системы искусственного интеллекта, заключающееся в возможности открытого, исчерпывающего, доступного, четкого и понятного представления информации».

Под робастностью понимается способность ИИ поддерживать уровень производительности при любых обстоятельствах. В ГОСТ Р 59276-2020 аналогом данного понятия является надежность – «свойство объекта сохранять во времени способность выполнять требуемые функции а заданных режимах и условиях применения, технического обслуживания, хранения и транспортирования». С понятием робастности тесно связано понятие устойчивости обучения, которое можно определить, как отсутствие склонности к переобучению.

ИИ сам по себе является объектом для компьютерных атак [38]. Любое несанкционированное вмешательство в работу ИИ может повлечь за собой нежелательные последствия – материальный ущерб, нарушение ИБ, угрозу жизни и здоровья граждан, технологический сбой или катастрофу и т.д. Все зависит от назначения конкретной реализации ИИ и возможностей, которыми данный

экземпляр обладает. Поэтому на объектах критической информационной инфраструктуры (КИИ) особенно важным является обеспечение функциональной и информационной безопасности ИИ. КИИ – это совокупность информационных систем и телекоммуникационных сетей, критически важных для работы ключевых сфер государства и общества: здравоохранения, промышленности, связи, транспорта, энергетики, финансового сектора и городского хозяйства [89].

Под функциональной безопасностью понимается безопасность и защита функций ИИ, а также функций объектов, связанных или управляемых ИИ. Информационная безопасность является частью функциональной безопасности и реализуется на разных уровнях – обучения, исполнения и тестирования. Обучение и тестирование биометрической системы может происходить в доверенной среде, однако исполнение всегда ведется в потенциально враждебной среде. Информационная безопасность ИИ связана с обеспечением защищенности знаний и моделей ИИ от компьютерных атак.

Доверие к ИИ во многом зависит от того, на каких данных и каким алгоритмом осуществлялось его обучение. В некоторых ответственных приложениях обучение ИИ должно выполняться в автоматическом режиме, без контроля со стороны человека (например, чтобы избежать таких угроз, как отравление данных, когда обучение выполняется в недоверенной среде). В таких случаях необходимо обеспечить робастность обучения. Однако сложные алгоритмы машинного обучения теряют устойчивость, если выборка незначительна в объеме или недостаточно репрезентативна. В наибольшей степени это касается аппарата многослойных (глубоких) нейронных сетей. Алгоритмы глубокого обучения сложно полностью автоматизировать, так как обладают высокой склонностью к переобучению, в особенности при малом количестве тренировочных примеров [63]. Поэтому обучение глубоких нейронных сетей ведется под контролем человека. Инженер-исследователь вынужден подбирать слишком много параметров, влияющих на структуру нейронной сети и алгоритм обучения, что создает большие трудозатраты. Кроме того, искусственные нейронные сети не пластичны. При функционировании

модели в реальной практике, данные могут меняться (смещаться, дрейфовать) со временем, что отразится на качестве решений. Крайне сложно быстро переобучить нейронную сеть или дообучать ее в процессе функционирования на малых выборках для предупреждения подобных ситуаций, так как методы глубокого обучения с подкреплением работают приемлемо только при пакетном обучении на больших выборках [127].

Настоящая работа направлена на решение следующих комплексных фундаментальных проблем доверенного ИИ, которые являются актуальными для ответственных приложений ИИ, и прежде всего для биометрических систем:

1. Защита от компьютерных атак для повышения доверия к ИИ. В настоящем исследовании рассматриваются следующие типы атак:

– анализ и интерпретация параметров обученного ИИ с целью извлечения конфиденциальной информации;
– состязательные атаки, направленные на принятие ИИ неверных или желательных для злоумышленника решений;
– зондирование моделей ИИ с целью извлечения знаний и параметров модели (например, весов и связей нейронов и восстановление по ним данных обучающей выборки полностью или частично);
– манипуляции с моделями для получения контроля над ИИ.

Для биометрических систем проблемы доверия и безопасности решены только в первом приближении (за счет серий стандартов ГОСТ Р 52633 и ISO/IEC 30107). Сегодня такие технологии как deepfake и нейросетевые голосовые синтезаторы [258, 339] позволяют генерировать реалистичные изображения лиц и голосовые образы субъектов, практически неотличимые от настоящих. Открытые биометрические образы (отпечаток пальца, радужка, лицо) находятся «на виду» и поэтому компрометируются в естественной среде. Злоумышленник может снять биометрические характеристики бесконтактно или скрыто от владельца (например, с ручки двери, фотографии). Для использования тайных биометрических образов (рукописных и голосовых паролей и других) требуется

надежная защита биометрических шаблонов при хранении и передаче по каналам связи при одновременном обеспечении высокой точности аутентификации, что пока не удается достичь на базе стандартов ГОСТ Р 52633.

2. Автоматизация машинного обучения с использованием незначительных объемов обучающих выборок. Важнейшим свойством для ИИ является возможность быстрого и устойчивого обучения на малом числе примеров, что означает способность ИИ обрабатывать большие объемы данных, а также формировать достоверные решения и делать высокоточные предсказания, даже если обучающая выборка ограничена в объеме и не в полной мере репрезентативна. Создание промышленных решений на базе ИИ не всегда возможно, если недостаточно данных для его обучения. Недостаток данных обучения может возникать по следующим причинам:

– собрать обучающую выборку технически сложно или процесс сбора данных связан с высокими материальными затратами. Например, в области медицины формирование выборки часто сопряжено с необходимостью верификации заболевания у пациента путем проведения инвазивных исследований. Обычно выборки достаточного объема собираются в течение многих лет;
– особенность задачи предполагает использовать только малые объемы обучающих данных. Наиболее остро проблема проявляется при разработке систем биометрической идентификации и аутентификации. Специфика этих задач заключается в том, что настройка биометрической системы должна выполняться быстро (нельзя требовать от пользователя повторять ввод биометрических данных множество раз, иначе система не будет востребована на практике). Проблема нехватки данных и низкой репрезентативности обучающей выборки в будущем никуда не исчезнет, независимо от того, какие объемы биометрических данных накоплены исследователями по всему миру. В реальной практике система будет обучаться на малом числе примеров (5-15).

3. Повышение предсказательной способности модели в процессе ее функционирования в условиях изменяющихся данных. Этот комплекс проблем принято разделять на две основные части:

– дрейф данных. К этой категории, как правило, относят предсказуемые или устранимые смещения данных в процессе функционирования модели. Например, сезонные изменения, повторяющиеся каждый календарный год, сбои датчиков, изменение единиц измерения, появление данных, неучтенных при обучении (хотя такие данные могли существовать ранее);
– дрейф концепций (концептуальный дрейф модели), связанный с непредсказуемыми изменениями. Различают постепенный и внезапный концептуальный дрейф. В первом случае свойства анализируемых образов меняются медленно, во втором – быстро и кардинально, что обычно связано с непредвиденными событиями, повлиявшими на саму концепцию решения задачи (например, эпидемия вызвала мировой кризис, и прогностические модели на рынке драгоценных металлов потеряли актуальность).

В биометрии дрейф данных или концепций возникает при порезах и травмах (актуально для *статических образов*, не меняющихся с течением жизни), устаревании биометрического эталона пользователя с течением времени, изменении психофизиологического состояния, например, опьянении (актуально для *динамических образов*, изменяющихся со временем – голос, почерк и т.д.).

ИИ, свободный от указанных проблем, удовлетворяет критериям робастности (надежности) и безопасности доверенных систем ИИ.

1.3 Уязвимости нейросетевых моделей искусственного интеллекта в задачах биометрической аутентификации

Традиционный подход к построению интерфейса взаимодействия с биометрической системой или системой управления на базе ИИ основан на том, что на вход ИИ поступает информация (в пакетном режиме или режиме реального времени), которая анализируется по некоторому алгоритму, после чего на выходе ИИ возникают управляющие воздействия. Каждое воздействие – это код команды (например, открыть/заблокировать доступ) из нескольких бит. В памяти (как

долговременной – на носителях информации, так и в кратковременной – оперативной) ИИ могут храниться конфиденциальные или персональные данные. Чтобы защитить эти данные от угрозы нарушения конфиденциальности, параметры решающих правил (например, таблицы весовых коэффициентов и связей нейронов), а также сами данные (биометрические, биомедицинские) принято шифровать [7, 8] на некотором криптографическом ключе. При таком варианте построения системы ИИ злоумышленники могут провести следующие атаки [151] (рисунок 1.1):

1. Атаки «на решающий бит» («один бит») [60]. Существует две ситуации, касающиеся такого рода атак. Первая связана с редактированием программного кода скомпилированного и обученного ИИ. Если на выходе ИИ возникают короткие команды, то злоумышленник может инвертировать логику программы, изменив решающее правило. Например, если на выходе нейронной сети располагается функция SoftMax, то достаточно поменять два ее выхода местами, чтобы заменить одно управляющее воздействие на другое (например, красный сигнал светофора на зеленый, рисунок 1.1 а). Вторая ситуация возникает, если хакер подключится к объекту управления или к каналу передачи данных с возможностью изменять сигналы на выходе ИИ. В этом случае он сможет имитировать определенные управляющие воздействия и изменять одну команду на другую. При этом ему не потребуется вникать в суть работы алгоритма анализа данных, достаточно лишь выявить ассоциации кода команды и связанного с ней действия. Для коротких управляющих команд выявить эти ассоциации несложно. Предсказание последствий исполнения управляющих команд за счет использования «цифровых двойников» объектов управления [108] далеко не всегда возможно даже в теории.

2. «Состязательные» атаки (спуфинг, атаки подбора), при которых хакер подает на вход ИИ сгенерированные, фальсифицированные или перехваченные данные [267] с целью получения на выходе ИИ желаемых управляющих воздействий (например, перед данными атаками уязвимы

сверточные нейронные сети в задачах классификации, когда на их входы подаются графические образы с наложением шумов). Это широкий класс атак, заключающихся в генерации синтетических или изменении естественных примеров данных с последующей подачей их на входы модели ИИ. В биометрических системах состязательные атаки могут быть реализованы путем синтеза и предъявления цифровых или физических «муляжей» (последнее названо атакой представления). Один из методов защиты сводится к обучению глубоких ИНС на состязательных примерах распознаванию этих атак [30, 31, 32], однако эффективность такого подхода в биометрии ограничена, так как невозможно учесть все вариации возможных подделок (в частности, для лицевой и голосовой биометрии показатели точности, полученные на открытых наборах данных, обычно не удается воспроизвести на практике).

3. Атака «извлечения знаний» [161]. Под этим термином подразумевается частичное или полное восстановление обучающей выборки путем зондирования нейронной сети и наблюдения статистики входов/выходов во время ее работы (рисунок 1.1 б), либо путем непосредственного анализа параметров обученной нейронной сети в незашифрованном виде. Конфиденциальная информация в памяти нейронной сети не должна быть извлечена злоумышленником, даже если ее параметры модели ИИ хранятся в незашифрованном виде. Хотя процедура обучения ИНС не подразумевает обратной разработки, восстановление знаний из таблиц нейросетевых функционалов в определенных случаях возможно. Федеративное обучение (заключение модели ИИ в защищенную среду и ее обучение без перемещения обучающей выборки куда-либо) не дает защиты от атаки «извлечения знаний», так как эта атака направлена на параметры уже обученного ИИ, при условии, что процесс обучения мог уже проходить в защищенной среде.

4. Атака «ключ под ковриком» [60]. Ключи шифрования параметров решающих правил ИИ должны где-то храниться. Чтобы алгоритмы ИИ исполнялись, требуется сначала дешифровать знания. Когда приложение

выполняет анализ входных данных, параметры решающих правил остаются незащищенными [60]. В теории гомоморфное шифрование [6] может быть использовано для защиты ИИ, но на практике имеются нерешенные проблемы с низкой производительностью [59] и с накоплением ошибок при шифровании даже небольших объемов данных [59, 67]. Начиная с некоторого размера гомоморфные шифртексты перестают расшифровываться правильно. Чем больше длина зашифрованного текста, тем больше вероятность, что верно дешифровать гомоморфное решение не удастся [59, 67]. На сегодняшний день для гомоморфного шифрования создан стандарт ISO/IEC 18033-6:2019 [252], но он не касается шифрования параметров нейросетевых решающих правил. Для защиты обученного ИИ и нейронных сетей с помощью гомоморфного шифрования следует разработать отдельные стандарты или рекомендации. Если не использовать гомоморфное шифрование, то ключи должны где-то храниться (в базе данных, коде ИИ и т.д.), что создает внешние и внутренние угрозы. Хакер может похитить ключ, используя уязвимости в защите или вступив в сговор с администратором (рисунок 1.1 в). Человек всегда является «узким местом» в системе безопасности [108], поэтому эти вопросы нельзя закрыть полностью. Создание инфраструктуры для безопасного хранения криптографических ключей – сложная задача, требующая значительных финансовых затрат.

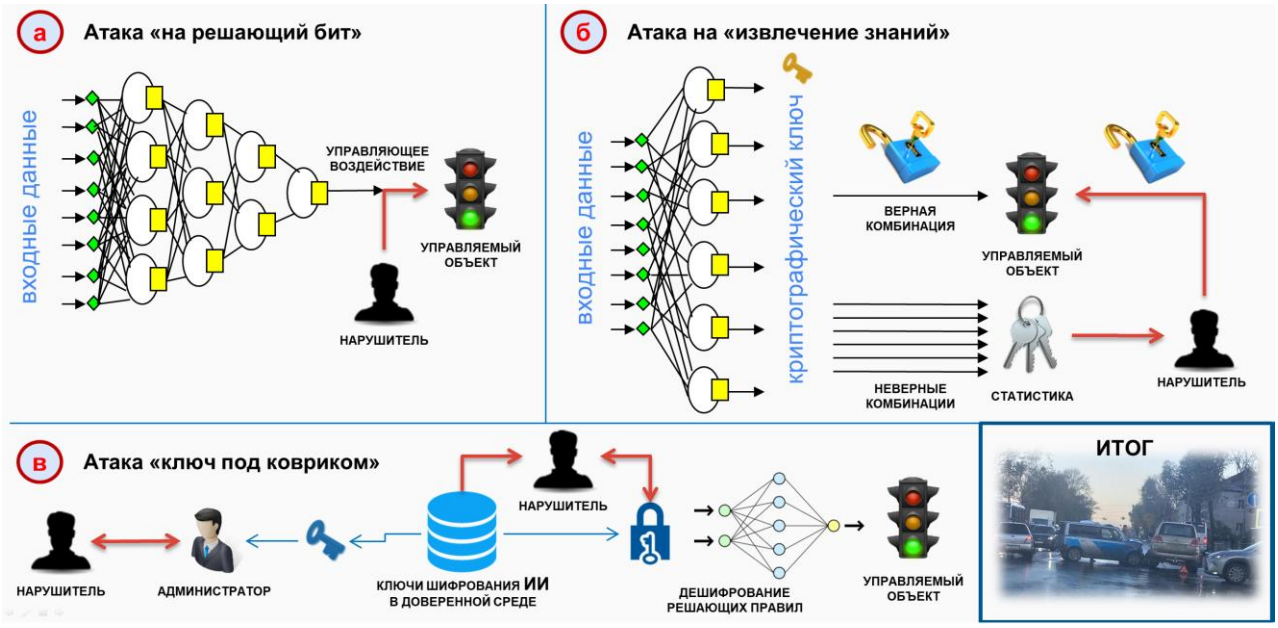


Рисунок 1.1 – Некоторые атаки на нейросетевой ИИ

Отметим, что сложность реализации указанных атак зависит от множества факторов. Однако при традиционном подходе к построению биометрических систем аутентификации (и других систем классификации образов) реализация этих атак в теории всегда возможна.

1.4 Федеративное обучение моделей искусственного интеллекта

Под федеративным (совместным) обучением сегодня понимается обучение на нескольких децентрализованных периферийных устройствах или серверах (узлах), содержащих локальные образцы обучающих выборок, без явного обмена этими выборками между узлами. Общий принцип состоит в обучении локальных моделей на выборках локальных данных и обмене параметрами (например, весами и смещениями нейронной сети) между этими локальными узлами с некоторой частотой для создания глобальной модели, совместно используемой всеми узлами. Федеративное обучения не стоит путать с распределенным обучением, которое направлено на распараллеливание вычислительной мощности и может выполняться в распределенной защищенной среде. Также отметим, что федеративное обучение нацелено на защиту данных от угроз нарушения конфиденциальности *на этапе обучения ИИ*, в том время, как рассматриваемые в предыдущем параграфе угрозы касаются обученного ИИ, который может исполняться в недоверенной среде.

Федеративное обучение имеет ряд нерешенных проблем и ограничений [176, 179], связанных с производительностью, передачей параметров обучаемой модели сторонним узлам (нужно обеспечить защиту этих параметров), различием в аппаратных платформах узлов, участвующих в процессе обучения (каждая платформа может иметь собственные ограничения) др. В репрезентативности обучающей выборки сложно убедиться без прямого (централизованного) доступа к тренировочным примерам. Обмен весовыми коэффициентами и смещениями нейронной сети (по сути это промежуточный результат обучения, т.е. параметры

обученного ИИ) может скомпрометировать знания ИИ. Сообщая обновления в процессе обучения (градиентную информацию), можно понять, не используют ли центральный сервер и сторонние серверы конфиденциальную информацию. Для защиты от этой угрозы используются методы обеспечения *дифференциальной конфиденциальности* [176, 179, 218]. Основной подход к защите строится на принципе «правдоподобного отрицания» (к ответам узлов добавляется шум, чем больше узлов участвует в объединении перед добавлением шума, тем меньше шума необходимо, чтобы скрыть индивидуальные ответы, в результате участники наблюдают общую статистику, но не видят чьих-либо прямых ответов). Такие методы дают меньший уровень защищенности, чем традиционные криптографические методы и снижают производительность обучения нейронной сети. Даже частичное совместное использование градиентов может вести к утечке конфиденциальности [312].

Для защиты градиентной информации многие исследователи предлагают использовать гомоморфное шифрование [176, 179, 189]. Однако, учитывая проблемы гомоморфного шифрования (накопление ошибок и низкая производительность), скорость и качество федеративного обучения, защищенного таким образом, оказывается гораздо ниже, чем при обычном обучении. На данный момент возможность применения федеративного обучения, полностью защищенного гомоморфным шифрованием, в реальных практических задачах вызывает сомнения, учитывая, что после обучения модели ее следует протестировать, причем следует сравнить результаты тестирования для двух случаев:

– после федеративного обучения с защитой гомоморфным шифрованием;
– после обычного обучения без гомоморфного шифрования.

Во втором случае точность работы модели должна быть существенно выше (вероятность ошибочных решений ИИ должна быть ниже). Это обусловлено, по крайней мере, двумя причинами. Во-первых, разбиение тренировочной выборки на пакеты (мини-батчи) при федеративном обучении может быть неравномерным относительно всего глобального набора обучения. Во-вторых, при исполнении

решающих правил классификатора, зашифрованных гомоморфным шифрованием, обычно повышается процент ошибочных решений (см. параграф 1.4.5), что свидетельствует об эффекте накопления ошибок (неоднозначном дешифровании результатов работы даже небольших решающих правил).

Наконец, федеративное обучение подвержено атакам Сивиллы [230], когда процесс обучения управляется злоумышленником с использованием вредоносных устройств (узлов). Предлагаются различные способы противодействия этой угрозе, что является отельным направлением исследований.

Подход федеративного обучения хоть и гипотетически позволяет сделать процесс обучения относительно безопасным, при этом создает угрозы конфиденциальности данных самой модели. Пока централизованное (обычное) обучение имеет преимущества перед федеративным обучением в скорости и обеспечении репрезентативности выборки. Для обеспечения конфиденциальности во многих случаях достаточно обезличивания тренировочных примеров (удаление из биометрических персональных данных информации об их владельце, замена этой информации на обезличенные идентификаторы). При должной проработке вопросов производительности и безопасности федеративное обучение может использоваться, но защиты от обозначенных атак оно не гарантирует.

1.5 Стандарты по защите искусственного интеллекта

На данный момент международным техническим комитетом ISO/IEC JTC 1/SC 42 «Artificial intelligence» опубликовано и введено 17 стандартов. Однако эти стандарты не постулируют методики защиты от обозначенных выше угроз. В рамках ISO/IEC TR 24028:2020 Information technology. Artificial intelligence. Overview of trustworthiness in artificial intelligence рассматриваются только общие подходы к оценке и достижению доступности, отказоустойчивости, надежности, точности, безопасности, защищенности и конфиденциальности систем ИИ. В ISO/IEC TR 24029-1:2021 Artificial Intelligence (AI). Assessment of the robustness of

neural networks. Part 1: Overview рассматриваются несколько иные вопросы, связанные с устойчивостью нейронных сетей – способностью поддерживать уровень производительности в различных условиях эксплуатации. В ISO/IEC TR 24368 Information technology — Artificial intelligence — Overview of ethical and societal concerns вопросы безопасности и приватности рассматриваются на высоком уровне абстракции и исключительно с этической точки зрения (затрагиваются также вопросы законодательного плана). В документе ISO/IEC 23894:2023 Information technology — Artificial intelligence — Guidance on risk management рассматриваются риски ИИ и управление рисками ИИ (идентификация, оценка, снижение и минимизация рисков ИИ). Однако методик для защиты ИИ от компьютерных атак и обеспечения конфиденциальности знаний ИИ не описывается.

Помимо указанных стандартов следует отметить другой международный стандарт, который касается вопросов защиты ИИ ISO/IEC 20547-4:2020 Information technology — Big data reference architecture — Part 4: Security and privacy. Данный стандарт разработан техническим комитетом ISO/IEC JTC 1/SC 27 «Information security, cybersecurity and privacy protection». Этот документ определяет аспекты безопасности и конфиденциальности, применимые к эталонной архитектуре больших данных (BDRA), и не дает рекомендаций относительно технических аспектов защиты систем ИИ, предназначенных для классификации образов, от описанных выше атак.

На текущий момент силами ТК 164 утвержден 51 стандарт, из них множество документов можно отнести к отраслевым (управление транспортом, медицина, анализ и синтез речи и др.). Проблемы защиты ИИ явно поднимаются лишь в нескольких из них. В ГОСТ Р 59276-2020 «Системы искусственного интеллекта. Способы обеспечения доверия. Общие положения» в таблице 4 прописаны следующие факторы снижения качества системы ИИ: недостаточная защищенность информации о модели данных (на этапе создания системы ИИ); недостаточная защищенность информации о модели данных, используемой в системе ИИ; недостаточная защищенность обрабатываемых персональных

данных; нарушение конфиденциальности персональных данных при выводе системы ИИ из эксплуатации. В ГОСТ Р 59276-2020 не приводится конкретных способов (методов) снижения влияния или нейтрализации данных факторов, так как это стандарт верхнего уровня. ГОСТ Р 59897-2021 «Данные для систем искусственного интеллекта в образовании. Требования к сбору, хранению, обработке, передаче и защите данных» рассматривает только проблему защиты данных, но не знаний ИИ, полученных после обучения, и только применительно к области образования. Раздел о защите данных содержит всего обобщенных 3 требования верхнего уровня абстракции. В ГОСТ Р 59898-2021 «Оценка качества систем искусственного интеллекта. Общие положения» в таблице 2 постулируются 5 существенных характеристик ИИ с точки зрения безопасности.

Наиболее детально вопросы защиты искусственного интеллекта проработаны для приложений биометрии, где предъявляются требования из реальной практики к защите биометрических эталонов от компрометации при их хранении и передаче по каналам связи. Силами международного комитета по стандартизации ISO/IEC JTC 1/SC 37 Biometrics введено 136 стандартов, многие из которых адаптированы для России. Международные биометрические стандарты, действующие на территории РФ, закреплены за ТК 098 «Биометрия и биомониторинг». Большинство из них касаются форматов биометрических данных и испытаний надежности биометрических систем. Серия из 4-х стандартов ISO/IEC 30107 «Information technology — Biometric presentation attack detection» посвящена защите от атак представления. Рассматриваемые атаки направлены на обман датчиков во время представления и сбора биометрических характеристик. Любые другие атаки выходят за рамки этих документов. Также стандарты не рассматривают конкретные механизмы противодействия таким атакам, а только основные принципы, классификацию атак и ряд других вопросов.

В основе международных стандартов (ГОСТ Р ИСО/МЭК 19784-1, 19784-2, 19784-4, 24708, 24709-1, 24709-2, 24709-3) лежит программный интерфейс BioAPI (БиоАПИ). Этот интерфейс подразумевает возможность защиты биометрических шаблонов с помощью криптографии. ГОСТ Р ИСО/МЭК 19785-4-2012

устанавливает требования к обеспечению целостности и шифрованию биометрических данных, в рамках стандарта описывается спецификация блока защиты информации для защиты биометрических данных в соответствии с требованиями ИСО/МЭК 19785-1. К сожалению BioAPI не гарантирует защищенность от описанных выше атак. Для биометрических систем, защищенных в соответствии со стандартами, основанными на BioAPI, проблема коротких управляющих команд и расшифровывания шаблона перед его применением остается актуальной. Соответственно использовать этот стандарт (или его адаптацию) для защиты других приложений искусственного интеллекта от обозначенных атак также пока не представляется возможным.

Продолжая обзор биометрических стандартов, следует также назвать три международных стандарта по защите биометрических эталонов, введенных техническим комитетом ISO/IEC JTC 27: ISO/IEC 24745:2022 Information technology -- Security techniques -- Biometric information protection; ISO/IEC 24761:2019 Information technology -- Security techniques -- Authentication context for biometrics; ISO/IEC 19792:2009 Information technology -- Security techniques -- Security evaluation of biometrics. Данные стандарты также не дают рекомендаций относительно реализации защиты нейросетевых решающих правил ИИ, предназначенного для классификации образов.

Национальные стандарты нейросетевой биометрии (таблица 1.1), закрепленные за ТК 362 «Защита информации», основаны на концепции преобразователя «биометрия-код». В соответствии с ГОСТ Р 52633.0-2006 [40] «преобразователь «биометрия-код»» (ПБК) – это преобразователь, способный преобразовывать вектор нечетких, неоднозначных биометрических параметров «Свой» в четкий однозначный код ключа (пароля). Преобразователь, откликающийся случайным выходным кодом на воздействие случайного входного вектора, не принадлежащего множеству образов «Свой».

ПБК можно сравнить с интеллектуальным «черным ящиком», который «знает» своего владельца и безопасно хранит его пароль или криптографический ключ. ПБК обучается формировать и отдавать пользователю его пароль (ключ)

при предъявлении биометрического образа «Свой» (рисунок 1.2). При предъявлении образа любого другого субъекта (образа «Чужой») ПБК должен формировать случайный бинарный код, близкий по информационной энтропии к “белому шуму” (рисунок 1.2). Предполагается, что сами пароли и ключи генерируются перед обучением ПБК в соответствии с принятыми нормами. Принципиального отличия в реализации ПБК при связывании биометрии с паролем, ключом шифрования или электронной подписи нет. В зависимости от применения ПБК на практике могут предъявляться требования к длине и информационной энтропии ключа, а также соответствующим свойствам ПБК. Данные обученного ПБК (ключ и биометрический эталон) должны быть защищены от компрометации при хранении и передаче по каналам связи [256]. Хакеры не должны иметь возможность извлечения знаний из обученного ПБК.

Таблица 1.1 – Семейство стандартов ГОСТ Р 52633, утвержденных ТК 362

Обозначение	Название стандарта
ГОСТ Р 52633.0-2006	Защита информации. Техника защиты информации. Требования к средствам высоконадежной биометрической аутентификации
ГОСТ Р 52633.1-2009	Защита информации. Техника защиты информации. Требования к формированию баз естественных биометрических образов, предназначенных для тестирования средств высоконадежной биометрической аутентификации
ГОСТ Р 52633.2-2010	Защита информации. Техника защиты информации. Требования к формированию синтетических биометрических образов, предназначенных для тестирования средств высоконадежной биометрической аутентификации
ГОСТ Р 52633.3-2011	Защита информации. Техника защиты информации. Тестирование стойкости средств высоконадежной биометрической защиты к атакам подбора
ГОСТ Р 52633.4-2011	Защита информации. Техника защиты информации. Интерфейсы взаимодействия с нейросетевыми преобразователями биометрия—код
ГОСТ Р 52633.5-2011	Защита информации. Техника защиты информации. Автоматическое обучение нейросетевых преобразователей биометрия—код доступа
ГОСТ Р 52633.6-2013	Защита информации. Техника защиты информации. Требования к индикации близости предъявленных биометрических данных образу «Свой»
Проект стандарта ГОСТ Р 52633.7	Защита информации. Техника защиты информации. Высоконадежная мультибиометрическая аутентификация
Проект стандарта ГОСТ Р 52633.xx-20xx	Защита информации. Техника защиты информации. Автоматическое обучение сетей квадратичных нейронов с многоуровневым квантованием биометрических данных

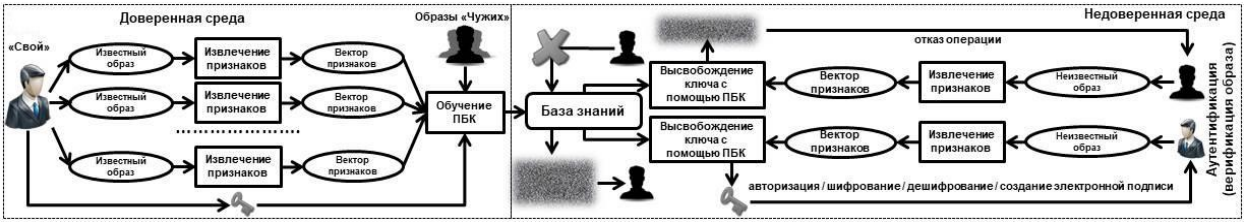


Рисунок 1.2 – Общая схема работы ПБК

Реализация концепции ПБК подразумевает под собой защиту биометрических приложений от атаки на «решающий бит», так как короткие управляющие команды заменяются на более длинные криптографические ключи. В России разработана серия стандартов ГОСТ Р 52633, которые описывают методики синтеза, автоматического обучения и тестирования ПБК на основе искусственных нейронных сетей для приложений высоконадежной биометрической аутентификации. Для защиты параметров обученных нейросетевых ПБК от атак «ключ под ковриком» и «извлечения знаний» разработано два документа:

– проект стандарта ГОСТ Р 52633.xx-20xx «Защита информации. Техника защиты информации. Автоматическое обучение сетей квадратичных нейронов с многоуровневым квантованием биометрических данных», первая редакция разработана ФБГОУ ВО «Пензенский государственный университет» в 2019 году, г. Пенза, находится в ТК 362 на этапе подготовки к публичному обсуждению [27];

– техническая спецификация «Системы обработки информации. Защита криптографическая. Защита нейросетевых биометрических контейнеров с использованием криптографических алгоритмов» документ принят большинством голосов на XXV заседании ТК 26 «Криптографическая защита информации» от 19.11.2020 [162].

На сегодняшний день вторая из указанных спецификаций является единственной (среди международных и российских стандартов), в которой изложена методика защиты искусственных нейронных сетей с помощью классических (не гомоморфных) криптографических методов применительно к приложениям высоконадежной биометрической аутентификации. Также отметим,

что особенности данной спецификации заключаются в том, что она позволяет реализовать эффективную защиту от угрозы «ключ под ковриком» без использования гомоморфного шифрования. Это становится возможным благодаря использованию специальной архитектуры нейронной сети, которая в совокупности с обычными криптографическими операциями позволяет устранить угрозу реализации данной атаки. При этом схема защиты не снижает точности сравнения биометрических образов (как это обычно бывает для схем защиты на базе гомоморфного шифрования), но накладывают ограничения на длину связываемого с биометрическим образом ключа (каждый признак может быть связан только с одним нейроном). Эффективность ГОСТ Р 52633 достаточна для приложений, использующих информативные биометрические образы с множеством признаков (например, отпечаток пальца, радужка), но если чи признаков ограничено либо образы малоинформативны (клавиатурный почерк, голос и др.), применять стандартизованный алгоритм и модель НПБК затруднительно (что следует из аналитического обзора, приведенного далее).

Техническая спецификация и серия стандартов ГОСТ Р 52633 рассчитаны на специфику применения в области биометрической аутентификации. Тем не менее, концепцию ПБК целесообразно распространить и на другие приложения искусственного интеллекта. Однако для этого следует определиться с граничными условиями для выбора архитектуры нейросетевого искусственного интеллекта и предложить методы синтеза и обучения универсальных преобразователей образов в код для задач классификации на основе нейронных сетей. Новые преобразователи могут быть основаны на иной архитектуре нейронов. После этого следует разработать (или заимствовать из других существующих стандартов) методы тестирования и криптографической защиты ИИ, построенного на основе концепции ПБК с использованием новой модели нейронов. Необходимость применения криптографической защиты по отношению к сетям нейронов, базирующихся на новой модели, следует оценивать криптографам в ходе публичного обсуждения, но уже после утверждения (или отклонения) новой архитектуры нейронов.

1.6 Методы защиты искусственного интеллекта на примере биометрических приложений

Рассмотрим существующие виды моделей ИИ, позволяющие реализовать концепцию ПБК. Преимущество этих моделей заключается в том, что архитектура ИИ изначально будет строиться с ориентацией на защиту от компьютерных атак. Однако требуется выявить, какие из существующих моделей могут быть устойчивы ко всем или большинству рассматриваемых атак, а также позволяют реализовать обучение в автоматическом режиме на малых выборках и онлайн-обучение.

Все известные типы моделей можно разделить на две основные категории:

1. Модели, основанные на методах помехоустойчивого кодирования. К этой категории относятся нечёткие экстракторы (fuzzy extractors, fuzzy vault, fuzzy commitment, fuzzy embedder рисунок 1.3а), а также гибриды нечеткого экстрактора с многослойной нейронной сетью (neural fuzzy extractors), нейронные сети в данном случае используются для извлечения признаков [105];
2. Модели, основанные на применении искусственных нейронных сетей. К этой категории относятся:
   - автоматически обучаемые нейросетевые ПБК (рисунок 1.3б), которые также могут быть комплексированы с глубокими нейронными сетями, извлекающими признаки [105];
   - нейросетевые ПБК на базе методов глубокого обучения (эта категория представлена небольшим количеством работ);
   - сети квадратичных нейронов и нейронов среднего гармонического, а также варианты их комплексирования с классическими нейронными сетями (далее гибридные сети).

Рисунок 1.3 – Иллюстрация принципов работы ПБК (слева — обучение ПБК, справа — высвобождение ключа): *а* — нечеткий экстрактор; *б* — нейросетевой ПБК

Указанные методы позволяют реализовать концепцию ПБК. Также рассмотрим методы защиты ИИ путем гомоморфного шифрования на примере реализации биометрических приложений.

1.7 Достигнутые результаты по защите приложений биометрической аутентификации на основе преобразователей биометрия-код

В основе нечеткого экстрактора (Fuzzy Extractor) [220] применяются методы помехоустойчивого кодирования, в частности, коды Адамара, БЧХ (например, Рида-Соломона) для исправления ошибок, возникающих из-за невозможности точного повторного воспроизведения биометрического образа. На ключ накладываются помехоустойчивый код, затем кодированная последовательность битов интегрируется с квантованными биометрическими характеристиками, которые рассчитываются с использованием данных обучающей выборки. Этот процесс формирует «открытую строку» (рисунок 1.3а). В процессе аутентификации субъект представляет биометрические данные, которые «вычисляются» из «открытой строки» с использованием операции исключающего «или» (XOR). Для получения исходного ключа в результирующей последовательности корректируются ошибочные позиции с помощью кодов, исправляющих ошибки.

Известны различные вариации данного подхода (Fuzzy Vault [66], Fuzzy Commitment [66]). Однако все они построены на едином основном принципе – применение классических корректирующих кодов. Далее объединим все указанные и аналогичные схемы общим названием – «нечеткий экстрактор».

Описанный подход имеет принципиальные недостатки. Прежде всего, эффективная длина криптографического ключа для классической схемы нечеткого экстрактора, связанного с биометрическим образом, жестко зависит от исправляющей способности помехоустойчивого кода. Это происходит потому, что все классические коды вносят избыточность. Чем больше исправляющая способность кода, тем больше избыточности и меньше длина генерируемого ключа, при этом избыточность растет экспоненциально. При исправлении 4% ошибок требуется 50% избыточности, при исправлении 12% накладывается 8 кратная избыточность (800%) [253]. Например, для того, чтобы исправить ошибки в биометрическом коде отпечатка пальца в работе [238] использовалась 15 кратная избыточность (только 128 информационных бит из 2048). По этой причине классические коды не могут исправить большое количество ошибок, и их затруднительно использовать по отношению к слабым биометрическим данным, таким как рукописная подпись, голос или клавиатурный почерк (образ легитимного пользователя может содержать более 50% неверных позиций, такой процент ошибок просто невозможно исправить при сохранении достаточной длины ключа).

В работе [106] приводятся уязвимости нечетких экстракторов, позволяющие при неизвестном ключе наблюдать статистики распределений близости к нему генерируемых ключей, другими словами приближенно определять число ошибочных бит. Это позволяет выполнять направленный перебор биометрических данных для взлома ключа (по аналогии с паролем). На данный момент методов полной нейтрализации приведенных в [106] атак на нечеткие экстракторы не предложено. Об утечке конфиденциальности и секретности в схемах нечетких экстракторов идет речь и в других работах, например [249]. Авторы показали, что для достижения достаточного уровня защиты эталона в

схеме Fuzzy Commitment дополнительно должны использоваться методы усиления конфиденциальности.

Наконец, нечеткие экстракторы не способны к полноценному обучению. Они квантуют «сырые» биометрические данные, при этом подавляются шумы оборудования, но не выполняется обогащения (не учитывается характер распределения значений признаков пользователей). По этой причине их можно применять, только если образы высокоинформативные (как отпечаток пальца [238] или радужка [231]). При работе с динамическими образами число ошибок оказывается значительным, а подход – несостоятельным [278, 281].

Приведем несколько работ за последние пять лет по данной тематике.

В [224] предлагается схема связывания ключа и биометрических параметров походки человека на основе схемы Fuzzy Commitment. Для извлечения признаков используется метод главных компонент, в качестве алгоритма помехоустойчивого кодирования – БЧХ-коды. Длина ключа при этом составляет 50 бит.

В работе [240] предлагается новая схема Fuzzy Embedder для связывания криптографического ключа и отпечатка пальца. Авторы утверждают, что схема абсолютно безопасна и не допускает «утечки конфиденциальности». Новая схема основана на адаптивном встраивании битов секретного ключа в биометрический шаблон. Зафиксирован достаточно низкий процент ошибок второго рода (ложного допуска) FAR = 0,0051% при вероятности ошибок первого рода (ложный отказ) 5%<FRR<20% в зависимости от длины ключа, которая составляла от 40 до 64 бит. Этот результат можно назвать одним из лучших для схем, подобных нечеткому экстрактору. Отметим, что образы отпечатка пальца являются одними из наиболее информативных, что позволяет достичь коэффициента равной вероятности ошибок EER=FAR=FRR=0,022% без защиты эталонов (данные соревнований на действующей платформе Fingerprint Verification Competition (FVC) [321]). По сравнению с этим показателем полученные показатели ошибок для схемы Fuzzy Embedder весьма велики.

В работе [232] предлагается использовать схему Fuzzy Vault по отношению к образам рукописной подписи, которые гораздо менее информативны, чем

отпечаток пальца. По заявлениям авторов им удалось связать криптографический ключ длиной 128 бит, при этом вероятности ошибок на различных базах данных составили: FAR=6,91% при FRR=7,85% (база MCYT), FAR=6,21% при FRR=4,86% (проприентарная база), FAR=6,16% при FRR=13,6% (база DS2 BioSecure database). При 0,02%≤FAR≤0,2% уровень ошибок «ложного отказа» составлял 76,53%≤FRR≤82,28%. Таким образом, фактически в реальной практике эту систему использовать невозможно.

В работе [348] предлагается модифицированная схема Fuzzy Vault, которая применяется для связывания 70 битного ключа с изображением радужной оболочки глаза. Информативность радужки обычно превышает информативность отпечатка пальца. Вероятность ошибок ложного отказа составляет от 3% до 5%, при вероятности ложного принятия близкой к нулю. Для получения этих показателей объединялись изображения радужки обоих глаз испытуемых.

Можно привести множество работ по данному направлению, однако вероятности ошибок и длины ключей будут сравнимы. Длина ключа для нечеткого экстрактора, применяемого по отношению к динамическим (тайным) биометрическим образам, в современных работах в лучшем случае составляет от 40 до 128 бит при очень высоком проценте ошибок по сравнению с технологиями обычной (классической) биометрической аутентификации.

Искусственные нейронные сети (ИНС) состоят из взаимосвязанных вычислительных элементов (нейронов), способных к обучению, приводящему к уличению качества решения задачи. ИНС кодируют данные об особенностях признаков пользователей весовыми коэффициентами, что не дает прямого наблюдения за биометрическими параметрами [161]. Нейросетевой ПБК (НПБК) строится персонально для каждого субъекта (т.е. он работает в режиме *верификации* – имеется 2 гипотезы «Свой» и «Чужой»), при этом формируется ИНС, количество входов которой равно числу признаков, а количество выходов – длине его личного ключа. Каждый нейрон последнего слоя генерирует один [161] или более [116] бит. Нейронная сеть обучается на биометрических образах пользователя и образах “Чужих”, чтобы вырабатывать ключ субъекта при

поступлении на вход его биометрического образа. Хорошо обученная нейронная сеть не нуждается в дополнительной корректировке выходов с помощью кодов, исправляющих ошибки (технически для дополнительной корректировки могут быть использованы специальные корректирующие коды, предложенные Безяевым [12], однако методика применения этих кодов выходит за рамки стандартов ГОСТ Р 52633). Обучение НПБК должно быть автоматическим, при этом объем обучающей выборки “Чужие” может быть сколь угодно большим. Разработчик биометрической системы может заготовить репрезентативную выборку “Чужие” заранее и использовались ее для обучения каждого ПБК. Однако число примеров образа “Свой” должно быть малым (в соответствии с ГОСТ Р 52633.5-2011 не менее 11 [41]), нельзя заставлять пользователя сотню раз вводить биометрический образ. Это обстоятельство накладывает ограничения на архитектуру ИНС, используемую в основе ПБК.

Для построения НПБК используются неглубокие архитектуры (shallow networks). В основу стандартов ГОСТ Р 52633 легли большие нейронные сети с малым числом скрытых слоев (одного или двух). Эти сети принято называть «широкими» [66]. Важным отличием «широких» сетей является процедура автоматического обучения (без использования алгоритма градиентного спуска). Обучение выполняется послойно, каждый нейрон обучается независимо от остальных нейронов сети, исходя из параметров закона распределения признаков, вычисляемых по данным обучающей выборки. Алгоритм обучения ГОСТ Р 52633.5 всегда остается устойчивым. Утверждение об устойчивости этого алгоритма к переобучению доказано (для биометрических приложений), что подтверждается результатами его публичного обсуждения. Добавление нейронов, увеличение числа входов и выходов сети не приводит к росту объемов обучающей выборки, а ведет к снижению количества ошибок (до определенного предела) [194]. Для обучения «широких» нейронных сетей [161], как правило, требуется значительно меньший объем выборки (в разы, десятки раз), чем для обучения многослойных нейронных сетей с помощью алгоритмов оптимизации, основанных на градиентном спуске.

Сегодня развитием НПБК занимаются преимущественно ученые из России и Казахстана [181, 190], совместными усилиями которых опубликована коллективная монография [161], где детально сравниваются методы на базе нечетких экстракторов и НПБК. По результатам сравнения НПБК значительно превосходят нечеткие экстракторы во всех отношениях, в том числе в длине ключа и точности распознавания образов.

Для сравнения с нечеткими экстракторами по уровню ошибок можно привести ряд работ, например [185]. В указанной работе НПБК используется для связывания ключей с образами рукописной подписи субъектов. Достигнутый уровень ошибок составил FRR=10%, FAR=10^{-7}% при длине ключа 256 бит, что гораздо лучше аналогичных показателей для нечеткого экстрактора [190].

НПБК не имеют недостатков, характерных для нечетких экстракторов [64]. Технически НПБК позволяет связать биометрический образ с ключом любой длины. Однако определенные ограничения все-таки существуют. Требования криптографии не позволяют использовать каждый биометрический признак дважды при нейросетевой обработке. Другими словами в соответствии с требованиями криптографического сообщества входы нейронов не должны дублироваться (каждый нейрон должен обрабатывать уникальное сочетание признаков). В противном случае НПБК могут оказаться подверженными атаке Маршалко [287], основанной на наблюдении одинаковых весовых коэффициентов в таблицах нейросетевых функционалов. С учетом этого требования длина ключа для НПБК будет снижена. Например, для 416 признаков, извлекаемых из рукописных образов, может получиться 26 нейронов, у которых имеется по 16 неповторяющихся входов [71]. Исследования показывают, что 16 входов достаточно для надежной аутентификации по динамике рукописной подписи. Для других биометрических модальностей достаточное количество входов может быть другим, оно зависит от информативности распознаваемых образов (для отпечатка пальца входов нейрона требуется меньше, для клавиатурного почерка – больше).

Еще одно ограничение связано с возможностью провидения атаки «извлечения знаний» из обученного НПБК путем статистического анализа стабильности выходов ПБК при поступлении на его входы естественных и синтетических образов «Чужих» [64, 161]. В приведенных работах описываются результаты исследования НПБК, в том числе касающиеся энтропии их откликов при поступлении на вход образов «Чужих». Для защиты от атаки «извлечения знаний» можно применить действенный метод криптографической защиты. Нейроны выстраиваются в цепочке путем создания перекрестных связей, после обучения таблицы каждого нейрона шифруются наложением гаммы, представляющей собой контрольную сумму выходов всех предыдущих нейронов в цепочке [64, 161]. Техническая спецификация [162], принятая большинством голосов на XXV заседании ТК 26 «Криптографическая защита информации» от 19.11.2020, регламентирует особенности реализации данного метода. Известны другие варианты атак «извлечения знаний» из классических НПБК [196].

Активные исследования в области неглубоких сетей ведутся и в других странах. Неглубокие сети способны к универсальной аппроксимации (для этого требуется потенциально неограниченное число скрытых нейронов, которое играет роль сложности модели). На данный момент проведена оценка ограничений малых сетей, сформулирован ряд теорем [273], получены нижние оценки их сложности в зависимости от соотношения между областью значений аппроксимируемой функции и размерностью входа [272]. Экспериментально установлено количество нейронов скрытом слое, больше которого увеличивать ИНС не имеет смысла, в зависимости от числа тренировочных образов [318].

Многие исследователи стараются использовать преимущества методов глубокого обучения многослойных нейронных сетей в задачах получения ключей из биометрических данных.

Впервые термин «глубокое обучение» употреблен в 1986 году после появления работы Рины Дехтер [214]. Но многослойные персептроны и алгоритм их обучения «обратным распространением ошибки» («градиентным спуском») предложены гораздо раньше Галушкиным А.И., который опубликовал подробную

монографию в 1974 году по данной теме [33] (хотя в литературе также встречаются отсылки к Дрейфусу (1973), Вербосу (1974) или Линненмаа (1970), как к независимым первооткрывателям данного базового алгоритма). В 1987 году Хехт–Нильсеном предложен вариант теоремы Колмогорова-Арнольда о представимости непрерывных функций нескольких переменных с помощью ИНС. Из теоремы Колмогорова–Арнольда–Хехт–Нильсена следует, что для любой функции многих переменных существует отображающая ее ИНС фиксированной размерности, при настройке которой можно изменять размерность входного вектора признаков, количества слоев и параметров активации нейронов [275]. Принципиальная сложность того времени заключалась в отсутствии необходимых для этого вычислительных ресурсов (обучение «градиентным спуском» имеет экспоненциальную вычислительную сложность, а объемы обучающей выборки достигали миллионов примеров). В 2002 году Джефри Хинтон усовершенствовал алгоритм обучения, применив машины Больцмана для обучения нижних слов нейронов [241]. Этот подход позволил сократить количество обучающих примеров в разы и лег в основу современных глубоких нейронных сетей. Развитие средств вычислительной техники дало толчок новым исследованиям в этом направлении. На сегодняшний день под глубоким обучением обычно подразумевается итерационная настройка многослойных нейронных сетей прямого распространения, при которой в том или ином виде используется алгоритм «обратного распространения ошибки» (он имеет 2 типа реализации: пакетного или стохастического градиентного спуска) [356].

Помимо классических ИНС все большее распространение получают сверточные нейронные сети (CNN), впервые предложение Яном Лекуном в 1988 году и не являющихся явным развитием идеи ограниченных машин Больцмана и подхода Хинтона. Сверточные сети существенно сокращают сложность нейросетевой обработки, так как снижают размерность задачи. Отметим, что в биометрических приложениях сверточные сети часто тестируются с точки зрения устойчивости к состязательным атакам. Наложение различного вида шума на изображение распознаваемых образов обычно существенно увеличивает FAR

[236] при использовании сверточных сетей. При необходимости можно привести множество работ, где упоминается схожая проблема.

Создать НПБК на основе многослойной нейронной сети с использованием итерационных алгоритмов обучения на сегодняшний день затруднительно [66]. Дело в том, что формирование и обучение ПБК, как правило, должно быть автоматическим (по крайней мере, в биометрии и многих других приложениях информационной безопасности), но практически все итерационные алгоритмы имеют существенную склонность к переобучению [68]. Как следствие, приходится постоянно следить за процессом обучения и периодически проводить проверку качества решений на валидационной выборке. Чем ниже информативность биометрического образа (т.е. уникальность и стабильность признаков), тем больший объем обучающей выборки нужен и тем выше склонность к переобучению. Например, для обучения сверточных нейронных сетей в задаче биометрической идентификации личности по лицу [44, 301] достаточно пяти примеров изображения лица от каждого идентифицируемого субъекта (в различных ракурсах) для того, чтобы процесс настройки был относительно робастным. Образ лица весьма информативен, гораздо информативней, чем, например, рукописный [46] или голосовой образ [91]. В задаче верификации подписей обучающая выборка возрастает до 15-30 примеров на подписанта [237], при этом точность распознавания разных подписантов уже сильно варьируется, а процесс обучения теряет робастность. При верификации диктора увеличение числа обучающих примеров может вызвать повышение EER [346], что как раз говорит о неустойчивости обучения. Можно привести достаточно много других примеров приложений, где требуется автоматизировать обучение, но на базе алгоритма градиентного спуска (и его модификаций) этого сделать не удается.

Тем не менее, известны работы, в которых сообщается об успешном опыте создания НПБК на основе многослойных сверточных нейронных сетей (пока что это касается приложений лицевой биометрии). Эти работы, в частности [216, 269], можно отнести в отдельную категорию (*НПБК на базе методов глубокого*

обучения). В исследовании [216] предлагается модель НПБК на основе сверточных нейронных сетей, которая рассчитана на обработку изображений лиц, имеющих размерность 64*64 точки. Архитектура сети включает два сверточных слоя (32 и 64 фильтра, соответственно, с размером ядра 7*7), слой MaxPooling (2*2), два полносвязных слоя и два слоя Dropout (с вероятностью пропуска 0,5), которые располагаются после полносвязных и чередуются с ними. На каждом слое используется функция активации ReLU, кроме последнего, где нейроны имеют сигмоидальные функции активации, к которым далее применяются пороговые функции (чтобы обеспечить бинарные выходы). При обучении каждому субъекту присваивается уникальный случайно генерируемый бинарный код – МЕВ (методика генерации позволяет получить высокую энтропию кода). Количество бинарных выходов нейронной сети равно длине МЕВ. При обучении нейронной сети на данных пользователя его МЕВ указывается в качестве желаемого выходного состояния. В исследовании длина МЕВ менялась от 256 до 1024 бит. Авторы заявляют, что обучение является робастным. При обработке образа лица бинарный код, возникающий на выходе ИНС хешируется с помощью криптографической хеш-функции. Показатели ошибок составили: FAR=1% при FRR=2,41% (на базе данных PIE).

В работе [269] рассматривается аналогичная задача, и представлено развитие описанного выше подхода. В отличие от [216] в работе [269] используется ансамбль (стек) из двух многослойных нейронных сетей. Первая сеть предварительно обучена на 2,6 млн. примеров и принимает на вход изображения лиц 224*224, а на выходе выдает вектор квантованных признаков, преобразуемых в 4096 битный бинарный код. Архитектура этой сети называется VGG-Face, сеть включает 13 сверточных и 2 полносвязных слоя. Вторая сеть переводит вектор признаков в ключ пользователя, который задается при ее обучении (по аналогии с МЕВ [216]) и имеет длину 256 или 1024 бита. Вторая сеть имеет 4 или 6 полносвязных слоя в зависимости от длины ключа. Для снижения эффекта переобучения используются слои Dropout. В качестве функций активации везде кроме последнего слоя используется ReLU, на последнем слое

используются сигмоидальные функции активации, к которым далее применяются пороговые функции для получения бинарного вектора (ключа). Для обучения сети использовался алгоритм оптимизации Adam. Для хеширования ключа используется функция SHA3-512. Фактически вторая сеть является НПБК, но работающим в режиме *идентификации*, т.к. ПБК запоминает сразу множество пользователей и ключей.

Авторы работы [269] заявляют, что отклики НПБК на образы «Чужих» (незарегистрированных пользователей) обладают достаточными показателями энтропии. Если обучать НПБК на данных базы PIE (образы «Свой»), а при тестировании в качестве образов «Чужих» брать данные из базы FIE, то расстояние Хэмминга между откликами на образы «Свой» и ключом (до его хеширования) будут близки к нулю, а для откликов на образы «Чужих» эти расстояния будут существенно больше. По результатам эксперимента (на базе PIE) показатели ошибок идентификации субъектов по одному кадру (изображению лица) составили EER=4% (256 бит), EER=3,6% (1024 бита), по 10 кадрам: EER= 0,15% (256 бит), EER=0,35% (1024 бита). С точки зрения точности идентификации эти показатели являются высокими. Однако далее проанализируем преимущества и недостатки схемы.

Фактически схема [269] заключается в извлечении признаков с помощью одной (предварительно обученной) многослойной ИНС, и их дальнейшем анализе с помощью НПБК на основе другой многослойной ИНС, которая обучается итерационным алгоритмом Adam. Этот алгоритм имеет склонность к переобучению, в отличие от алгоритма ГОСТ Р 52633.5. Хотя авторам работы [269] по их заявлениям удалось сделать обучение робастным, в другой задаче классификации при аналогичном подходе робастность обучения НПБК не гарантирована, так как архитектура НПБК будет совершенно другой. Синтезировать многослойный НПБК (слои, нейроны и связи), обучаемый алгоритмом, базирующемся на градиентном спуске, сложнее, чем синтезировать неглубокий НПБК, обучаемый по абсолютно устойчивому алгоритму. Синтез многослойной нейронной сети с оптимальной конфигурацией [138, 266] под

заданную обучающую выборку и ее автоматическое обучение – крайне сложная научная задача, которая пока имеет общего эффективного решения. По всей видимости, сложность этой задачи превышает сложность автоматизации обучения уже синтезированной нейронной сети. Имеющиеся теоремы «о полноте», доказывающие принципиальную возможность решения ряда задач с помощью ИНС, дают мало информации для построения их оптимальных конфигураций. В итоге практика сводится к проведению эмпирических исследований, позволяющих найти оптимальное решение для частных случаев при отсутствии общего правила формирования ИНС. Эти исследования и породили множество алгоритмов оптимизации ИНС.

Таким образом, если следовать общей схеме из работы [269] при разработке ИИ для других приложений (не лицевой биометрии), то архитектуры нейросети для извлечения признаков и НПБК будут жестко зависимы от задачи классификации. Если использовать НПБК на базе «широкой» ИНС, то зависимым от задачи будет только алгоритм извлечения признаков, а для неглубокого НПБК можно разработать алгоритм автоматического синтеза и обучения, независимо от задачи. Такой НПБК будет легче протестировать, так он будет структурно проще, чем в работе [269]. В целом метод из работы [269] является более прогрессивным, чем метод из работы [216], и показывает более низкий процент ошибок.

Известны другие работы по синтезу НПБК на основе образов лица и их обучению с помощью градиентного спуска [169, 297]. Авторы также отметили необходимость переобучения НПБК при регистрации нового пользователя.

Еще нужно обратить внимание на то, что подходы из работ [216, 269] подразумевают, что НПБК работает в режиме идентификации (одна сеть обучается распознавать множество классов образов). Для биометрических приложений это неудобно. В реальной практике при регистрации нового пользователя придется заново обучать всю сеть. Классическая схема системы аутентификации на базе НПБК не обладает подобным недостатком. При регистрации нового пользователя, для него создается отдельный НПБК, при этом нет необходимости заново обучать уже существующие НПБК других

пользователей. Задачу идентификации можно реализовать путем построения множества НПБК, а схему идентификации из [216, 269] свести к верификации образов затруднительно.

Известно работы, в которых глубокие нейронные сети с множеством выходов обучают извлекать признаки из биометрических образов. Например, в работе [319] на основе глубокой нейронной сети из образов сетчатки извлекаются признаки, после чего они квантуются и преобразуются в 96 битный бинарный код, который авторы называют ключом. Недостатки этих работ в том, что генерируемый из биометрических данных код на выходе нейронной сети коррелирован с входными биометрическими данными, что небезопасно (подобное свойство может использоваться для сокращения числа вариантов перебора при осуществлении состязательных атак).

Извлечение признаков с помощью многослойных нейронных сетей нельзя приравнивать к генерации ключа на основе биометрических данных, так как к криптографическим ключам и паролям предъявляются определенные требования, связанные с их длиной и энтропией. Признаки, непосредственно извлеченные из биометрического образа (или любого другого распознаваемого образа), не обладают соответствующими свойствами. Извлечение признаков – это лишь этап обработки образа в системе связывания предварительно сгенерированного ключа и биометрических параметров субъекта. Поэтому в работе [319] следовало использовать концепцию ПБК, чтобы связать 96 битный код, получаемый на выходе нейронной сети, с криптографическим ключом или паролем, заранее сгенерированным в соответствии с общепринятыми методиками. Даже если предположить, что получаемые непосредственно из сетчатки бинарные коды (или их контрольные хеш-суммы) обладают высокой уникальностью, стабильностью и энтропией (что нужно проверять на больших выборках), разработанный метод [319] и ему подобные будут жестко зависимы от типа распознаваемых образов.

Следует упомянуть также еще одну категорию методов, предназначенных для хеширования изображений [215]. Данные методы направлены на преобразование графических образов в бинарный код, однако их нельзя

использовать в качестве НПБК, так как они не подразумевают автоматического обучения и не используют в качестве целевого выхода нейронной сети заранее заданный криптографический ключ или пароль. Однако концепция нейросетевого хеширования имеет общие черты с концепций ПБК в части требований к энтропии генерируемых кодов.

Из-за низкой робастности алгоритмов обучения ИНС при решении рассматриваемых задач многослойные нейронные сети обычно комплексируются со схемами связывания ключа, подобным нечеткому экстрактору. Результат такого объединения не стоит относить к принципиально новым схемам, так как суть при этом не меняется – новая схема наследует все недостатки нечеткого экстрактора. Единственным значительным улучшением является более эффективный блок извлечения признаков, представляющий собой глубокую предварительно обученную нейронную сеть. Примером подобной работы является [338]. В работе предлагается метод распознавания личности на основе анализа параметров походки. Сигналы от датчиков мобильного устройства (акселерометра и гироскопа) обрабатываются нейронной сетью, которая извлекает признаки, которые квантуются (представляются в бинарном виде). Далее вектор признаков поступает на вход блока классификации, который реализован на основе схемы Fuzzy Commitment с использованием кодов БЧХ. Длина связываемого таким образом ключа составила 128 бит при EER=5,5%. Как можно видеть глубокие сети позволяют повысить длину ключа по сравнению с простым применением схемы Fuzzy Commitment (128 против 50 бит в [224]) за счет извлечения более информативных признаков.

Также рассмотрим вопрос глубокого обучения нейросетвого ИИ в процессе его функционирования (обучение без явного учителя). Когда выполнить обучение с учителем практически невозможно находят применение методы нейроэволюции [326]. На сегодняшний день при сравнительно небольших обучающих выборках эволюционный подход успешно применяется для подбора топологий и весов в ИНС с одним скрытым слоем. Применение этого подхода к глубоким сверточным нейронным сетям возможно на больших выборках [191, 268, 305, 350].

Эволюционные подходы сложны в реализации и могут оказаться неприменимы непосредственно к построению НПБК. Однако их можно использовать для синтеза и обучения с подкреплением автокодировщика, который является составной частью ИИ, находящейся вне НПБК. Другими словами защищенный нейросетевой ИИ может обучаться с подкреплением без перенастройки НПБК, но путем обучения блока извлечения признаков.

Активное развитие «широких» нейронных сетей в настоящее время идет по направлению комплексирования нейронов, имеющих принципиально разную архитектуру, в рамках одной гибридной сети [229]. Предложено несколько вариантов новых моделей нейронов, не компрометирующих биометрический эталон. Функция активации при этом, как правило, остается пороговой (чтобы обеспечить генерацию бинарного кода нейронной сетью).

К таким моделям относятся квадратичные нейроны, которые легли в основу проекта стандарта [27], на данный момент находящегося на этапе публичного обсуждения в ТК 362. Сети квадратичных нейронов обучаются в полностью автоматическом режиме. Они имеют общие черты с сетями радиально-базисных функций, но в отличие от последних не компрометируют биометрический эталон. Квадратичные нейроны имеют иную архитектуру, что позволяет скрывать параметры распределения признаков, вычисляемые при обучении. Это свойство и робастность обучения квадратичных нейронов дают возможность создавать на их основе НПБК [61].

Проведенные исследования указывают на то, что при 4 входах квадратичный нейрон имеет производительность, сравнимую с классическим нейроном (с настройкой по ГОСТ Р 52633.5), имеющим 16 входов [71]. Это позволяет создавать на базе квадратичных нейронов НПБК, которые:

– дают меньший процент ошибочных решений, чем классический НПБК;
– дают на выходе бинарный код большей длины, чем классический НПБК. Так как для защиты от атаки Маршалко [287] требуется, чтобы входы нейронов не повторялись, то длина ключа на выходе такого НПБК будет выше примерно в 4 раза (например, в задаче верификации образов подписи

с использованием 416 признаков [70], длина кода составит 104 бита, вместо 26 бит).

Другой новой моделью, не компрометирующей параметры обученного ИИ, является нейрон среднего гармонического [65, 95, 96], где основной операцией является умножение (в то время как в классическом нейроне используется взвешенное суммирование входов). Автором работы [96] утверждается, что у нейронов среднего гармонического допустимо появление малого числа общих связей, поэтому длина криптографического ключа повышается до 38% по сравнению с классическим НПБК.

В таблице 1.2 представлены обобщенные сведения о наиболее высоких из известных результатов по биометрической аутентификации с использованием принципов концепции ПБК. Как можно видеть, все существующие методы либо имеют принципиальные недостатки, либо дают высокий процент ошибок или низкую длину ключа. Также проведенный обзор показал, что статические биометрические образы (отпечаток пальца, радужка, лицо и т.д.) сложно сохранить в тайне (мы не можем всегда быть в перчатках, маске и очках, не компрометируя открытые биометрические образы). Это означает, что их проще подделать (можно снять данные со стакана, ручки двери, фотографии и т.д.). При этом появляются большие возможности по компрометации систем, так как открытый образ имеет мало возможных вариантов замены (10 пальцев, 2 глаза и одно лицо, технологии изготовления электронных и цифровых муляжей постоянно совершенствуются). Секрет, содержащийся в биометрическом образе, многократно усиливает его защитные свойства. Однако простое сочетание пароля и отпечатка пальца хоть и несколько усложняет взлом системы, но не меняет ситуацию кардинально и не устраняет корень проблем. Требуется, чтобы биометрический образ был тайным (секрет должен быть внутри образа), тогда компрометация пароля не приводит к компрометации системы, так как пароль еще нужно правильно воспроизвести (сказать, написать или напечатать). Но вероятность ошибки «ложного допуска» для рукописных, голосовых образов гораздо выше, чем для статических образов.

Таблица 1.2 – Основные методы биометрической аутентификации, полностью или частично реализующие концепцию ПБК

AER – Average error rate, средний уровень ошибок (FRR≠FAR)

EER – Equal Error Rate, коэффициент равной вероятности ошибок (FRR=FAR)

Подход	Тип образа	Особенности метода и работы	Длина ключа, бит	Показатели ошибок, %	Год
нечеткий экстрактор [238]	радужка	Использование двухуровневой схемы исправления ошибок на базе кодов Адамара и Рида-Соломона. Затруднительно использовать тайные биометрические образы из-за недостатков нечетких экстракторов	140	AER=0,5	2006
НПБК [185]	руко-писный пароль	Однослойная нейронная сеть, обучаемая по ГОСТ Р 52633.5-2011. Возможно реализовать защиту от атак [196, 287] путем использования спецификации [162]. Каждый признак должен быть связан только с одним нейроном, что налагает ограничения на длину ключа, связываемого с биометрическим образом	256	FRR=10, FAR=10-7 (расчет вероятностей дан для одного испвытуемого)	2014
нечеткий экстрактор [281]	подписи	Используются коды Адамара. Энтропия ключа недостаточна (использовалась упрощенная процедура квантования признаков). Высокий процент ошибок. Метод подвержен недостаткам нечетких экстракторов	304	FRR=14,8 FAR=5	2016
НПБК [216]	лицо	Модель основана на использовании глубоких нейронных сетей. Обучение оптимизатором Adam. Метод не универсален. При регистрации нового пользователя нужно полностью переобучать нейронную сеть. Подход затруднительно перенести на другие модальности. Открытый биометрический образ лица легко скомпрометировать	от 256 до 1024	FRR=2,41, FAR=1	2016
нечеткий экстрактор [348]	радужка	Модификация схемы Fuzzy Vault. Затруднительно использовать тайные биометрические образы из-за недостатков нечетких экстракторов	70	3≤FRR≤5, FAR≈0	2016
нечеткий экстрактор [231]	радужка	Извлечение высокоинформативных признаков. Невозможно использовать тайные биометрические образы из-за недостатков нечетких экстракторов. Недостаточная достоверность результата	200-400	1,26≤FRR≤ 3,75, FAR≈0	2017
нечеткий экстрактор [240]	отпеча-ток пальца	Схема fuzzy embedder. Базируется на нечетком экстракторе. Низкая длина ключа. Невозможно использовать тайные биометрические образы	40-60	FAR=0,0051 5<FRR<20 EER=0,022	2017
гибрид-ный НПБК [91]	лицо, клавиа-турный почерк	Двухфакторная непрерывная биометрическая аутентификация в системах смешанного документооборота. Комплексирование классических нейронов, с квадратичными нейронами и многомерными нейронами Байеса. Без дополнительной криптографической защиты гибридный НПБК компрометирует биометрические данные пользователей. Предлагается адаптировать спецификацию [162] для защиты знаний. Однако этот вопрос не проработан и оставлен для дальнейших исследований (предлагается только концепт). Предложенная модель НПБК может быть также потенциально уязвима перед рядом атак.	200	FRR=0,2, FAR=0,09 (60 секунд мониторин-га)	2017

Продолжение таблицы 1.2

нечеткий экстрактор [224]	походка	Схема Fuzzy Commitment. Использовались БЧХ-коды, метод главных компонент. Для извлечения признаков анализировались видео данные. Низкая длина ключа. Недостаточная достоверность результата – проверка результатов на базах CMU MoBo (25 человек) и CASIA (20 человек), объем которых не дает оснований утверждать о нулевых показателях ошибок. Метод подвержен недостаткам нечетких экстракторов	50	0<FRR≤ 4, FAR≈0	2019
Преобразование образа в вектор бинарных признаков [319]	сетчатка	Не используется концепция ПБК в полной мере. Модель реализует блок извлечения бинарных признаков на основе глубоких нейронных сетей. Вектор признаков представляет собой бинарный код, используемый для аутентификации. Нельзя настраивать модель на генерацию ключа или пароля. Вопросы обеспечения высокой энтропии кода, извлекаемого из образов Чужих, не проработаны.	96	4,83≤AER ≤4,85	2020
Гибрид нечеткого экстрактора и нейронной сети [338]	походка	Сигналы акселерометра и гироскопа обрабатываются нейронной сетью, которая извлекает признаки, которые квантуются. Далее вектор признаков поступает на вход блока классификации. Схема Fuzzy Commitment с использованием кодов БЧХ. Метод подвержен недостаткам нечетких экстракторов	128	EER=5,5	2020
нечеткий экстрактор [232]	подпись	Схема Fuzzy Vault. Высокое количество ошибок, недостаточная длина ключа. Метод подвержен недостаткам нечетких экстракторов	128	FAR=6,91, FRR=7,85; FAR=6,21, FRR=4,86; FAR=6,16, FRR=13,6%	2020

Объединение различных нейронов (классических, квадратичных, гармонических и др.) в гибридную сеть потенциально позволяет значительно снизить вероятность ошибочных решений. Гибридную сеть можно рассматривать, как ансамбль слабых классификаторов [139]. Снижение вероятности ошибок гибридной сети объясняется теоремой Кондорсе, которая утверждает: если мнения экспертов независимы, и вероятность правильного решения каждого из них больше 0,5, то с увеличением количества экспертов вероятность правильного решения комитета экспертов возрастает и стремится к единице. Причем, чем выше вероятность верного решения для каждого эксперта в отдельности, тем выше вероятность верного решения комитета. Отметим, что решение любого нейрона с бинарным выходом можно инвертировать, чтобы преодолеть барьер Кондорсе в 0,5 (известны также доказательства других теорем [315], позволяющих обойти барьер Кондорсе).

Однако на практике решения нейронов, играющих роль экспертов, в той или иной мере коррелированы, чем ниже коррелированность решающих правил, тем более ощутим положительный эффект при их комплексировании. Таким образом, имеются следующие параметры, которые влияют на эффективность гибридной нейронной сети: количество нейронов, матрица коэффициентов корреляции Пирсона между решениями всех возможных пар нейронов, мощность нейронов (их способность давать верные решения).

Для повышения энтропии откликов НПБК на образы «Чужих» также предлагаются модели нейронов с множеством бинарных выходов [27, 61, 116]. Увеличение количества выходов нейрона можно рассматривать как потенциальный способ защиты от ряда атак на НПБК [196], так как этот прием затрудняет рекурсивный анализ алгоритма работы НПБК. Математических конструкций, на базе которых можно сформировать новую модель нейрона, потенциально существует множество [139]. Однако предложенные модели нейронов и НПБК на их основе пока не дают значительных преимуществ перед классическими нейронами.

1.8 Достигнутые результаты по защите приложений биометрической аутентификации на основе гомоморфного шифрования

Гомоморфное шифрование — форма шифрования, позволяющая производить определённые математические действия с зашифрованным текстом и получать зашифрованный результат, который соответствует результату операций, выполненных с открытым текстом. Идея гомоморфного шифрования впервые высказана в 1978 году авторами алгоритма ассиметричного шифрования RSA. Одна из первых схем шифрования, гомоморфная относительно умножения и способная зашифровать всего лишь один бит, предложена в 1982 году. В 2005 году предложена криптосистема, которая основывалась на использовании билинейных спариваний на эллиптических кривых, позволяющая выполнять

неограниченное количество операций сложения и одну операцию умножения [197]. Первая конструкция полностью гомоморфной криптосистемы предложена Крейгом Джентри в 2009 году [208].

За последние годы опубликовано немало работ по гомоморфному шифрования и всего один стандарт ISO/IEC 18033-6:2019 [252], но этот стандарт не касается шифрования параметров нейросетевых решающих правил. Проблемы методов гомоморфного шифрования заключается в низкой производительности и накоплении ошибок (чем длиннее гомоморфный шифротекст, тем больше вероятность, что он не сможет быть расшифрован на гомоморфном ключе). Из-за накопления ошибок под каждую архитектуру нейросетевого ИИ обычно приходится подбирать соответствующие операции гомоморфного шифрования. Поэтому метод защиты нейросетевых решающих правил гомоморфным шифрованием, как правило, сложно переносится на другие архитектуры нейронной сети. Рассмотрим несколько последних работ в этой области.

В работе [200] речь идет о полном гомоморфном шифровании данных отпечатка пальца для систем пограничного контроля с учетом требований Общего регламента защиты персональных данных (GDPR, постановление Европейского Союза). В статье описывается система аутентификации с использованием гомоморфной защиты биометрических шаблонов. Хотя для ускорения сравнения предъявляемых образов с эталонами отпечатка пальца применяются параллельные вычисления, авторы отмечают, что процедура распознавания личности требует много времени и значительных вычислительных ресурсов.

В статье [175] предлагается метод гомоморфной защиты параметров лиц, извлекаемых из изображений с помощью глубоких нейронных сетей. Извлекаемые параметры шифруются с помощью алгоритма Пайе. Структура всей системы включает три стороны: клиент, сервер данных и сервер проверки. Сервер данных сохраняет зашифрованные пользовательские функции и идентификатор пользователя, сервер проверки выполняет проверку, а клиент отвечает за сбор информации о запрашивающей стороне и ее отправку на серверы. Информация передается между сторонами в виде зашифрованного текста, ни одна из сторон не

знает закрытых ключей, кроме сервера проверки. Предложенная схема протестирована с двумя архитектурами глубоких сверточных нейронных сетей на наборах данных Wild и Faces94. По заявлению авторов предложенная схема шифрования снижает точность сравнения верификации образов лиц. Защите образов с лиц также посвящена работа [195].

В исследовании [242] предлагается две архитектуры для блоков классификации, обеспечивающих защиту шаблонов голоса на основе гомоморфного шифрования, соответствующие устаревшему стандарту ISO/IEC 24745:2011 (Information technology -- Security techniques -- Biometric information protection).

В статье [296] по заявлению авторов предложена первая общая структура для защиты мультибиометрических шаблонов, основанная на гомоморфном шифровании, которая позволяет обрабатывать только зашифрованные данные с учетом выполнения всех требований стандарта ISO/IEC 24745:2011 без снижения точности распознавания. Однако авторы все же отмечают, что главный недостаток схемы – низкое быстродействие, что затруднит использование схемы при осуществлении операций с зашифрованными данными, если они будут иметь больший объем.

В статье [344] предлагается новая система биометрической верификации и защиты шаблонов THRIVE. Система включает новые протоколы регистрации и аутентификации, основанные на пороговой гомоморфной криптосистеме, в которой закрытый ключ используется совместно пользователем и проверяющим. В системе THRIVE в базе данных хранятся только зашифрованные двоичные биометрические шаблоны, а проверка выполняется с помощью гомоморфно рандомизированных шаблонов. Система THRIVE разработана для случаев, при которых сторона-мошенник может произвольно отклоняться от спецификации протокола. Пользователь, и проверяющий должны сотрудничать, чтобы расшифровать зашифрованные двоичные шаблоны. Предлагаемая система может использоваться в приложениях, где пользователь не желает раскрывать свои

биометрические данные верификатору, хотя ему необходимо подтвердить свое физическое присутствие с помощью биометрических данных.

В работе [306] предлагается эффективный метод вычисления расстояния Хэмминга для зашифрованных данных с использованием гомоморфного шифрования на основе идеальных решеток. В предложенной реализации безопасного расстояния Хэмминга для 2048-битных двоичных векторов с размерностью решетки 4096, шифрование вектора, безопасное расстояние Хэмминга и дешифрование занимают соответственно около 19,89, 18,10 и 9,08 миллисекунд на Intel Xeon X3480 при 3,07 ГГц.

В [347] предлагается протокол аутентификации на основе радужки с использованием гомоморфного шифрования. Среди недостатков можно отметить низкую производительность.

В некоторых публикациях встречалась идея замены прямых «гомоморфных» операций на предсказание их результатов с помощью глубоких нейронных сетей. На вход нейронной сети поступают данные в зашифрованном виде, и она обучается предсказывать результат вычислений (в зашифрованном или дешифрованном виде). Обучение нейронной сети требует выборки очень большого объема, а результат предсказания имеет вероятностный характер (всегда существует вероятность ошибки предсказания). Также подобный подход зависим от операций, которые эмитирует нейронная сеть, так как для каждой последовательности операций с зашифрованным текстом следует использовать отдельную обучающую выборку и топологию нейронной сети.

В работе [67] предлагается раздельно выполнять гомоморфное шифрование параметров каждого нейрона в гибридной нейронной сети, заранее определив число гомоморфных операций для искусственных нейронов каждого типа. Например, для классического нейрона с 16 входами требуется выполнить 16 операций сложения и столько же операций умножения. Таким образом, можно сократить количество одновременно шифруемых операций и снизить (или устранить) эффект накопления ошибок. Проще всего раздельное шифрование реализовать для неглубоких НПБК.

1.9 Выводы и задачи исследований

Вопросы защиты систем высоконадежной биометрической аутентификации и других ответственных приложений ИИ, неправильная работа которых может повлечь за собой материальный ущерб, нарушение информационной безопасности, угрозу жизни и здоровья граждан, крайне актуальны. Разработкой биометрических систем контроля доступа и рабочего времени занимается множество крупных организаций, в частности, Biolink Solutions, Центр речевых технологий (ЦРТ), Anviz, Сонда Технолоджи и др. Подавляющее число технологий основано на открытых биометрических образах.

К основным научным организациям (институтам и лабораториям), занимающимся разработкой и исследованием биометрических методов распознавания личности относятся:

- The Biometric System Laboratory [http://biolab.csr.unibo.it/Home.asp]
- Международная научная лаборатория Многомодальных биометрических и речевых систем при Санкт-Петербургском национальном исследовательском университете информационных технологий, механики и оптики. Создана на кафедре речевых информационных систем (РИС), организатором которой является ЦРТ.
- Norwegian Biometrics Laboratory [http://nislab.no/biometrics_lab].
- Научно-исследовательский и испытательный центр биометрической техники МГТУ им. Н.Э. Баумана [http://biometric.bmstu.ru/].
- Biometrics Lab of Federal University of Viçosa [http://www.biometria.ufv.br/eng/?page_id=11]. Задачи лаборатории связаны с интеллектуальным анализом и обработкой данных с акцентом на селекцию растений.
- Пензенский научно-исследовательский электротехнический институт (АО "ПНИЭИ") [http://пниэи.рф/activity/science.htm]. Научный коллектив из АО "ПНИЭИ" под руководством Иванова А.И. является одним из ведущих в

России по рассматриваемому направлению, так как под его руководством разработана серия национальных стандартов ГОСТ Р 52633.

– Омская научная школа на базе ОмГТУ.

Идеологически и концептуально наиболее близкой к Омской научной школе расположена научная школа Пензы. Данные научные направления объединяет использование НПБК, изначально предложенных в Пензе. Однако отличительной особенность Омских направлений исследований является акцентирование внимания не только на биометрии, но и на вопросах защиты и обучения искусственного интеллекта в целом, а также построении биометрических нейроинтерфейсов на базе электроэнцефалографии (ЭЭГ). Коллектив ОмГТУ концентрирует усилия на развитии методов построения неглубоких искусственных нейронных сетей, ансамблевых методов, искусственных иммунных сетей, байесовских методов, учете корреляционной зависимости между признаками и обучении ИИ на малых выборках.

Кроме того, в России существует шесть университетов, на базе которых функционируют научно-исследовательские центры ИИ мирового уровня – это Сколковский институт науки и технологий, Московский физико-технический институт (МФТИ), Институт системного программирования им. В.П. Иванникова, Университет Иннополис, Национальный исследовательский университет ИТМО и Высшая школа экономики (ВШЭ). Все эти университеты в той или иной степени занимаются исследованиями в области доверенного искусственного интеллекта, что частично пересекается с направлениями исследований ОмГТУ. Отличительной особенностью Омской научной школы является проведение исследований именно в области защищенного исполнения алгоритмов ИИ без применения криптографических средств, а также в области обучения ИИ на малых выборках.

На сегодняшний день нет международных стандартов, которые бы регламентировали технические аспекты защиты ИИ от угроз реализации атак «на решающий бит», «ключ под ковриком», состязательных атак и атак «извлечения знаний», а также особенности построения архитектуры ИИ, изначально

устойчивого к этим атакам. Серия отечественных стандартов ГОСТ Р 52633 ориентирована на биометрические технологии с высокой информативностью и низкой изменчивостью признаков. Недостатками стандартов ГОСТ Р 52633 является недостаточная длина ключа, необходимость применения дополнительных средств криптографической защиты (в соответствии со спецификацией [162]), более высокий уровень ошибок (в сравнении с некоторыми классическими методами аутентификации), а также невозможность дообучения НПБК в процессе функционирования. Длина ключа для НПБК, обучаемого по ГОСТ Р 52633.5-2011, жестко ограничена из-за подверженности атаке Маршалко (для 416 признаков длина ключа составляет 26 бит). Для сетей квадратичных нейронов атака Маршалко также актуальна, однако длина ключа может быть примерно в 4 раза выше, чем для НПБК. Тем не менее, ограничения длины ключа не позволяют использовать такие нейроны для широкого спектра классификационных задач. Аналогичные выводы можно сделать относительно нейронов среднего гармонического. Нечеткие экстракторы с учетом известных недостатков [55, 56, 184] не удовлетворяют требованиям надежности. Аналитико-синтетический обзор литературы показал, что на данный момент не предложено моделей НПБК и алгоритмов их обучения, способных решить рассматриваемые в работе проблемы.

Гомоморфное шифрование открывает значительные перспективы для защиты от обозначенных угроз ИБ, однако на сегодняшний день разработать и стандартизовать метод защиты искусственных нейронных сетей для биометрических систем и других приложений ИИ пока не удалось. Имеются нерешенные научно-технические проблемы: низкая производительность, накопление ошибок (вероятностный характер дешифрования), зависимость предложенных схем защиты от архитектуры ИИ. Результаты исследования документального потока указывают на то, что разработать метод защиты на базе гомоморфного шифрования, отвечающий всем требованиям в скором времени вряд ли удастся.

Применение технологий глубокого обучения и аппарата многослойных ИНС в системах высоконадежной биометрической аутентификации, по всей видимости, ограничивается задачами извлечения признаков. Классификатор должен иметь простую архитектуру, такую, чтобы его синтез и обучение могли выполняться в автоматическом режиме, что практически невозможно, если в его основе лежат глубокие нейронные сети.

Кроме того, в системах высоконадежной биометрической аутентификация крайне важно обеспечить актуальность эталона пользователя. С течением времени возрастает количество сбоев и ошибок для таких систем [279]. Изменчивость динамических признаков зависит от психофизиологического состояния субъекта [246]. Это актуально для систем на базе клавиатурного почерка [145], рукописного почерка [144], голоса [23], а также методов распознавания личности по электроэнцефалограмме (ЭЭГ) [110], термограмме лица и другим типам образов [337]. Изменчивость статических признаков лица, отпечатка пальца, радужки может быть связана с травмами, протезами, макияжем, использованием глазных линз и т.д. Данная проблема является частным случаем более общей проблемы дрейфа данных и концепций [199]. Концептуальный дрейф значительно снижает робастность (надежность) модели ИИ и со временем делает ее непригодной.

Известно много методов снижения влияния концептуального дрейфа [199]. Как правило, для его обнаружения оцениваются ошибки или точность классификации, что затруднительно получить, не имея информации об истинных метках обрабатываемых объектов [177, 260, 199, 299]. Поэтому смещение в данных обычно обнаруживается на основе их статистических характеристик [88, 170, 226, 322]. Однако при обнаружении дрейфа не всегда возможно переобучить модель, так как данных для этого может не быть. Поэтому наилучшим подходом противодействия дрейфу являются алгоритмы онлайн-обучения [127, 304]. Существующие модели НПБК и алгоритмы их обучения имеют узкое применение и не способны компенсировать снижение надежности ИИ при наличии дрейфующих характеристик моделей.

Для решения обозначенных проблем требуется разработать концепцию защищенного исполнения нейросетевых алгоритмов ИИ, модели искусственных нейронов и НПБК, изначально устойчивые к деструктивным воздействиям и атакам, адаптивные модели ИИ, способные подстраиваться под изменяющиеся данные, снижая влияние концептуального дрейфа в задачах высоконадежной биометрической аутентификации, а также алгоритмы их автоматического обучения и онлайн-обучения. Следует разработать методы, технологию и программный комплекс для высоконадежной многофакторной биометрической аутентификации с обеспечением защиты биометрических данных от компрометации. Длина ключа, связываемого с биометрическим образом, должна соответствовать или быть выше требований стандартов (например, ГОСТ Р 34.10-2012). При этом нужно использовать биометрические образы, которые не компрометируются в естественной среде, т.е. которые пользователь может скрыть от непосредственного наблюдения.

Цель диссертационной работы: повысить надежность многофакторной биометрической аутентификации на основе технологии автоматического синтеза и обучения нейросетевых моделей доверенного ИИ.

Для достижения цели выполнены следующие **задачи по разработке**:

1. Концепции защищенного исполнения нейросетевых алгоритмов искусственного интеллекта.
2. Моделей искусственных нейронов и нейросетевого преобразователя биометрия-код, потенциально устойчивых к деструктивным воздействиям, и алгоритмов их робастного автоматического обучения на малых выборках.
3. Адаптивной модели ИИ и алгоритмов ее обучения, позволяющих предупредить или снизить влияние концептуального дрейфа данных в системах биометрической аутентификации.
4. Методов и алгоритмов многофакторной биометрической аутентификации с обеспечением защиты биометрических данных от компрометации.
5. Технологии автоматического синтеза и обучения нейросетевых моделей для высоконадежной многофакторной биометрической аутентификации.

2 Сети корреляционных нейронов для защищенного исполнения процедур высоконадежной биометрической аутентификации

Прежде всего, требуется разработать математические и концептуальные основы, на которых будут базироваться разрабатываемая технология автоматического синтеза и обучения нейросетевых моделей ИИ и системы высоконадежной многофакторной биометрической аутентификации.

В настоящей главе предлагаются и описываются:

– концепция защищенного исполнения нейросетевых алгоритмов искусственного интеллекта, которая позволяет сформировать устойчивость модели к извлечению знаний;
– модель корреляционных нейронов и модель нейросетевого преобразователя биометрия-код на их основе, а также робастной алгоритм их автоматического синтеза и обучения.

Корреляционные нейроны – это новый класс нейронов, анализирующих корреляционные связи между признаками вместо значений признаков в задачах классификации образов. Анализ внутренних корреляционных связей образов и принятие классификационных решений происходит без необходимости хранения информации о корреляционных связях или значениях признаков, характерных для биометрических образов пользователей. Другими словами эталонная информация о классах образов не компрометируется при хранении. Процесс обучения корреляционных нейронов и НПБК полностью автоматический и сохраняет робастность даже на малых обучающих выборках.

Также в главе речь пойдет об архитектурных принципах построения ИИ, исполняемого в защищенном режиме.

Чтобы ИИ можно было считать защищенным от рассматриваемых атак, ИИ должен работать, как преобразователь входных воздействий (поступающей информации) в длинный криптографический ключ или длинный случайный пароль, который можно ассоциировать с определенным управляющим воздействием. Таким образом, вместо коротких кодовых команд на выходе ИИ, нужно использовать длинные криптографические ключи. Каждый ключ должен

быть ассоциирован с отдельным действием, и только объект управления должен иметь представление о том, какая реакция соответствует принятому коду, и игнорировать любой неизвестный код (последовательность управляющих команд может быть зашифрована на данном ключе). Другими словами – *ИИ должен проходить процедуру аутентификации с использованием неотчуждаемого от ИИ ключа или пароля, который должен храниться безопасно в защищенной памяти (ключ должен «помнить» только ИИ)*. *Должна быть предусмотрена возможность изменения ключа (пароля), используемого для авторизации ИИ, в любой момент*. Для этого нейросетевую модель необходимо обучить продуцировать на своих выходах ключи или пароли при поступлении на его входы аутентичных образов, а также продуцировать случайный код при поступлении на входы состязательных примеров. В биометрических приложениях эти принципы реализуются в рамках концепции ПБК.

2.1 Защищенное исполнение нейросетевых алгоритмов и архитектурная безопасность искусственного интеллекта

Введем следующие определения, касающиеся процедуры защищенного исполнения алгоритмов искусственного интеллекта и процедур высоконадежной биометрической аутентификации.

Защищенное исполнение нейросетевых алгоритмов доверенного ИИ – концепция, при которой модель или система ИИ строится на основе нейросетевых алгоритмов, обладающих повышенной устойчивостью к следующим деструктивным воздействиям: анализ логики алгоритма классификации (как по принципу «белого», так и «черного» ящика) любым неавторизованным лицом, извлечение и интерпретация знаний ИИ (например, персональных данных), зондирование моделей машинного обучения, реализация состязательных атак. Защищенное исполнение нейросетевых алгоритмов ИИ может быть реализовано с использованием нейросетевых преобразователей образов в код.

Защищенное исполнение процедуры высоконадежной биометрической аутентификации – это защищенное исполнение нейросетевых алгоритмов доверенного ИИ в задачах бинарной классификации (верификации) биометрических образов с использованием НПБК при низком проценте ошибочных решений (в том числе, при $FAR \leq 10^{-12}$).

Параметры обученного ИИ (в биометрии – параметры шаблонов биометрических образов), необходимо хранить в специальном виде, защищенном от извлечения знаний даже при отсутствии стороннего шифрования (конечно, шифрование можно применять для усиления защиты).

Jain A.K. и другие изложили 4 свойства [256], которыми должна обладать схема защиты биометрического эталона:

- обеспечение конфиденциальности биометрического эталона;
- возможность замены эталона, если ключевой биометрический образ субъекта был скомпрометирован;
- получение исходного биометрического образа (эталона) из образа (эталона), представленного в защищенном виде, должно быть вычислительно трудно;
- схема защиты биометрического шаблона не должна ухудшать FAR (вероятность ошибки ложного допуска) и FRR (вероятность ошибки ложного отказа) биометрической системы.

Схожие требования можно сформулировать в более общем виде применительно к системам доверенного ИИ, функционирующим на объектах КИИ:

- необходимо обеспечить конфиденциальность знаний ИИ (невозможность частичного или полного восстановления фрагментов обучающей выборки, содержащих конфиденциальные данные);
- защищенный режим исполнения не должен блокировать возможность развития (дообучения) ИИ в процессе функционирования, если такие функции изначально были предусмотрены. В особо ответственных приложениях *должна быть предусмотрена возможность смены ключа*

(пароля), используемого для авторизации ИИ. Для подобной перенастройки нужна процедура *быстрого автоматического обучения* блока классификации, отвечающего за связывание ключа с другими знаниями ИИ (контроль над возникновением переобучения должен быть автоматическим). Это требуется и в тех случаях, когда ИИ необходимо адаптировать к новым условиям работы, или если злоумышленник мог перехватить входные данные для реализации состязательных атак (чтобы можно было дообучить ИИ не реагировать на предъявление скомпрометированных данных);

- извлечение параметров обученного ИИ в незащищенном виде из базы знаний, представленных в защищенном виде, должно быть вычислительно трудно (или невозможно);
- схема защиты нейросетевого ИИ не должна приводить к значительному снижению производительности ИИ и точности его решений, в защищенном режиме исполнения нейросетевой ИИ должен работать с низкими показателями ошибочных решений.

Для реализации концепции защищенного исполнения нейросетевых алгоритмов ИИ в задачах классификации образов предлагается разделить функционал ИИ на блок выделения признаков и блок классификации (рисунок 2.1). Блок извлечения признаков преобразуют образ в вектор фиксированной длины (эту операцию можно назвать ортогонализацией образа). Этих блоков может быть множество (в зависимости от приложения могут быть предусмотрены блоки предварительной обработки, нормировки и т.д.). Далее вектор признаков подается на вход в блок классификации, который в системах высоконадежной биометрической аутентификации представляет собой НПБК.

На этапе извлечения признаков образ нормируется, и из него удаляется незначимая информация. Блоки извлечения признаков могут быть реализованы на основе практически любых подходов (нейронные сети, классические методы спектрального и корреляционного анализа и др.). Эти блоки могут быть обучены на больших выборках практически любым алгоритмом, их задача – извлекать из

классифицируемых образов наиболее значимую информацию. В общем случае блок извлечения признаков является зависимым от предметной области, так как для разных приложений входные данные могут кардинально отличаться, как и характер извлекаемой из образа информации (вектора признаков). Например, для обработки звука часто используются методы x-vector, d-vector, i-vector, быстрое преобразование Фурье, вычисление мел-кепстральных коэффициентов или вейвлет преобразование. Разные подходы могут комбинироваться.

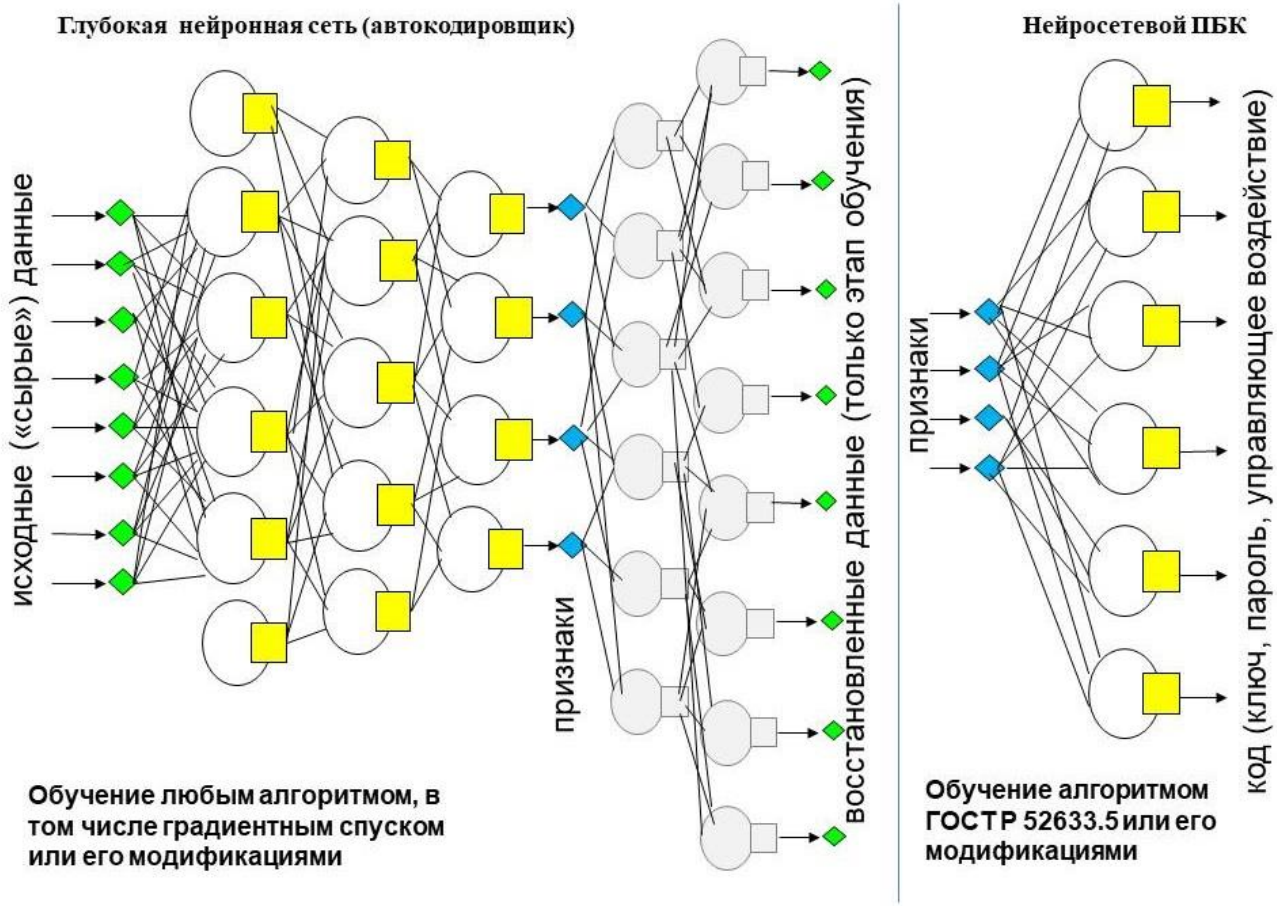


Рисунок 2.1 – Схема связывания ключа и распознаваемого образа, объединяющая «глубокую» и «широкую» сети (блок извлечения признаков и блок классификации на базе НПБК)

Блок извлечения признаков может быть реализован на основе автокодировщика – предварительно обученной глубокой нейронной сети со специальной архитектурой, которая способна сжимать размерность входных данных, кодируя их набором информативных признаков, а также восстанавливать входные данные из вектора признаков [105]. Автокодировщик обучается как единая сеть, однако для извлечения признаков используется только кодировщик –

сеть, отвечающая за сжатие пространства признаков. Декодировщик – сеть, отвечающая за восстановление данных, необходима только на этапе обучения (без нее невозможно обучить кодировщик). Кодировщик может принимать на вход данные, предварительно обработанные другим алгоритмом ортогонализации. Слои автокодировщика могут быть любыми: сверточными, рекуррентными, полносвязными, частично связными и др. Обучение автокодировщика гипотетически может вестись любым алгоритмом на выборках большого объема (в случае биометрических приложений на выборке обезличенных биометрических данных), в том числе алгоритмом градиентного спуска или его модификациями.

Отметим, что блоки извлечения признаков могут оставаться в незащищенном виде, так как они не продуцируют классификационных решений на своих выходах, а также не хранят персональных данных и другой конфиденциальной информации (биометрических данных, ключей или паролей пользователей). Выбор архитектуры блока извлечения признаков можно почти полностью возложить на разработчика системы ИИ. Требования к блокам извлечения признаков могут касаться только свойств вектора признаков (например, требования к закону распределения признаков).

Блок классификации должен строиться по стандартизованной технологии с предъявлением жестких требований к архитектуре. Обучение блока классификации должно быть робастным. Этот блок будет функционировать в потенциально враждебной среде и должен работать в режиме защищенного исполнения, обеспечивая конфиденциальность знаний, а также высокую точность решений. Должна быть предусмотрена возможность закрепить ассоциацию между классом образов и заранее генерируемым криптографическим ключом.

Безопасность блока классификации может быть дополнительно усилена криптографическими методами, если в этом будет необходимость. Для НПБК, обучаемого по ГОСТ Р 52633.5, уже разработана спецификация по криптографической защите решающих правил. Но не любые знания можно защитить при помощи данной спецификации. Для новой модели НПБК может потребоваться адаптация данной спецификации, разработка новой или вообще

целесообразности в криптографической защите не будет. Эти вопросы могут рассматриваться техническим комитетом №26.

В системах высоконадежной биометрической аутентификации выходы блока извлечения признаков должны быть связаны с НПБК (как это показано на рисунке 2.1). Обучение НПБК может выполняться в доверенной среде, но обученный НПБК размещается в потенциально враждебной среде. Кодировщик может быть размещен в «облаке», чтобы функции извлечения признаков были доступны удаленно. В некоторых биометрических приложениях декодировщик не следует размещать в недоверенной среде, в противном случае злоумышленник может использовать его для восстановления исходных данных из компактного описания.

При правильной реализации предлагаемой архитектуры и концепции защищенного исполнения нейросетевых алгоритмов ИИ можно получить преимущества, которые дают методы глубокого обучения многослойных нейронных сетей (выявление сложных закономерностей в данных) и НПБК (защита от рассмотренных компьютерных атак и быстрое автоматическое обучение). НПБК можно повторно обучить/дообучить в автоматическом режиме на малой выборке, в то время как автокодировщик может обучаться как заранее на больших объемах данных, так и в процессе функционирования системы (в том числе, с использованием эволюционных подходов и генетических алгоритмов).

Отметим, что таким образом нужно строить архитектуру ИИ не во всех приложениях. Речь идет только о системах высоконадежной биометрической аутентификации и наиболее ответственных приложениях на объектах КИИ, вмешательство в работу которых может повлечь за собой серьезные последствия – значительный материальный ущерб, нарушение информационной безопасности, угрозу жизни и здоровья граждан, технологический сбой или катастрофу и т.д.

Опишем математические основы концепции защищенного исполнения нейросетевых алгоритмов ИИ для задач классификации образов.

2.2 Искривление пространства признаков с учетом их информативности и коррелированности. Мера Минковского и ее свойства

В качестве решающих правил, защищенных гомоморфным шифрованием, находят применение меры близости Евклида, Пирсона и мера «городских кварталов» [306, 344]. Эти меры близости обобщаются в виде меры Минковского [330]:

$$y = \sqrt[g]{\sum_{j=1}^{n} \left| \frac{(m_j - a_j)}{\sigma_j} \right|^g},$$

где a_j – значение j-го признака из вектора $\bar{a}$, представляющего собой распознаваемый образ; n – количество признаков; m_j и σ_j – математическое ожидание и среднеквадратичное отклонение значений j-го признака для класса «Свой», с которым сравнивается образ $\bar{a}$ (класс «Свой» представляет биометрические образы одного из легитимных пользователей); g – степенной коэффициент, определяющий уровень «искривления» пространства. При g=1 мы получаем меру хи-модуль, при g=2 – меру Пирсона, при $g \to \infty$ мера Минковского стремится к мере Чебышева. В зависимости от значения степенного коэффициента g расстояние Минковского оказывается различным. Рисунок 2.2 иллюстрирует, как может выглядеть окружность в двумерном пространстве Минковского. В отличие от примера на рисунке 2.2 пространство признаков многомерно (каждый признак «создает» одно измерение). Однако для наблюдателя, находящегося в евклидовом пространстве, все многомерные сферы, построенные при $g \neq 2$, будут иметь аналогичные с рисунком 2.2 деформации.

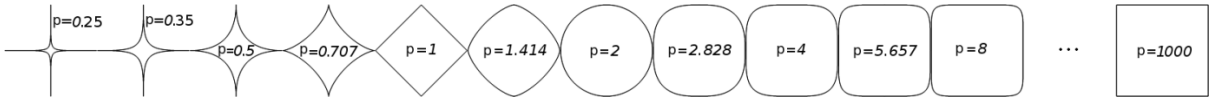


Рисунок 2.2 – Окружность на плоскости при различных значениях степенного коэффициента g

Искривление пространства признаков возникает из-за наличия корреляционных связей между измерениями (рисунок 2.3). Как правило,

пространство признаков не является ни плоским, ни в равной степени искривленным. Скорее уровень искривления пространства признаков меняется относительно наблюдателя. Все классы образов имеют отличающиеся матрицы коэффициентов корреляции $C_{j,t}$ (2.1) между признаками (биометрический образ каждого человека имеет уникальную корреляционную матрицу). Поэтому относительно различных классов пространство признаков искривлено по-разному.

$$C_{j,t} = \frac{\sum_{k=1}^{K_G}(a_{t,k} - m_t)(a_{j,k} - m_j)}{\sqrt{\sum_{k=1}^{K_G}(a_{t,k} - m_t)^2 \sum_{k=1}^{K_G}(a_{j,k} - m_j)^2}} \approx \frac{K_G \sum_{k=1}^{K_G}(a_{t,k} - a_{j,k}) - (\sum_{k=1}^{K_G} a_{t,k} \cdot \sum_{k=1}^{K_G} a_{j,k})}{\sqrt{\left(K_G \sum_{k=1}^{K_G} a^2{}_{t,k} - (\sum_{k=1}^{K_G} a_{t,k})^2\right)\left(K_G \sum_{k=1}^{K_G} a^2{}_{j,k} - (\sum_{k=1}^{K_G} a_{j,k})^2\right)}}, \quad (2.1)$$

где K_G – количество обучающих примеров образа «Свой» (далее K_I – количество обучающих примеров образа «Чужие»), k – порядковый номер примера в обучающей выборке «Свой».

Чтобы учесть неоднородность искривления пространства признаков также можно использовать меру Махалонобиса:

$$y = (\overline{m} - \overline{a})^T \cdot [R]^{-1} \cdot (\overline{m} - \overline{a}),$$

где $[R]^{-1}$ – матрица парных коэффициентов корреляции между признаками. Использование данной квадратичной формы усложняется тем, что требуется выполнить обращение корреляционной матрицы [R], что на практике не представляется возможным, так как данная операция является плохо обусловленной и выполняется со значительными ошибками [91].

Помимо уровня корреляции важным показателем является уровень информативности признаков [193]. Количество собственной информации j-го признака для определенного класса образов определяется по формуле:

$$I_j = -\log_2(AUC(\Phi_G(a_j), \Phi_I(a_j))),$$

где AUC — площадь (area under curve), ограниченная функциями плотности вероятности (ФПВ) «Свой» $\Phi_G(a_j)$ и «Чужие» $\Phi_I(a_j)$, а также осью абсцисс (рисунок 2.4). $\Phi_G(a_j)$ характеризует значения признака строго для определенного класса образов, $\Phi_I(a_j)$ характеризует значения этого же признака для всех классов образов в целом [193]. Чем выше I в среднем, тем дальше разнесены собственные области классов в пространстве признаков.

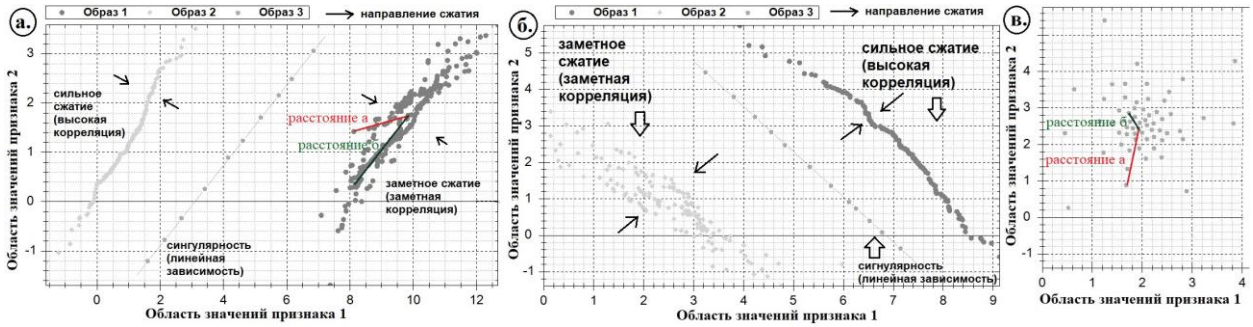


Рисунок 2.3 – Направление сжатия пространства двух признаков:

а. при положительной корреляции между признаками (расстояние «а» на самом деле больше, чем расстояние «б», так как пространство признаков не является «плоским», а искривлено из-за корреляции),

б. при отрицательной корреляции,

в. при независимости признаков (расстояние «а» больше, чем «б»)

Изменяя параметр g можно добиться снижения количества ошибок классификации [330]. Чтобы это продемонстрировать проведен вычислительный эксперимент по распознаванию образов в пространстве 200 абстрактных (имитированных) признаков. Все признаки имели нормальное распределение значений (наиболее распространённый случай для биометрии). На каждом этапе эксперимента генерировалось два пространства признаков – независимых (C≈0, рисунок 2.5а) и зависимых (C>0, рисунок 2.5б). Отличия этапов заключалось в уровне коррелированности зависимых признаков и информативности признаков в целом (рисунок 2.4). Эксперимент состоял из 3-х этапов:

1. Малоинформативные признаки (I≈0,5 бит – почти предельно низкий показатель количества информации для биометрического параметра [159]) со значительной корреляцией (C≈0,55).
2. Почти неинформативные признаки (I≈0,15) с очень высокой корреляционной зависимостью (средний коэффициент парной корреляции составил C≈0,9).
3. Весьма информативные признаки (I≈1,75) с низкой корреляционной зависимостью (C≈0,1).

Генерируемые классы образов отличались между собой параметрами распределения значений признаков (класс описывался двумя векторами параметров – математических ожиданий и среднеквадратичных отклонений). Значения независимых признаков генерировались методом Монте-Карло под соответствующие параметры классов. Для набора классов с зависимыми признаками перед формированием соответствующих образов $\bar{a}$ значения каждого признака внутри класса были отсортированы по возрастанию. Так в первом наборе признаки оставались независимыми случайными величинами, а во втором – между ними появлялась корреляционная зависимость. Таким образом, в эксперименте смоделировано 6 вариантов пространства признаков (зависимых и коррелированных с учетом 3 уровней информативности). Для каждого случая сгенерировано 500 классов по 125 примеров образа на класс.

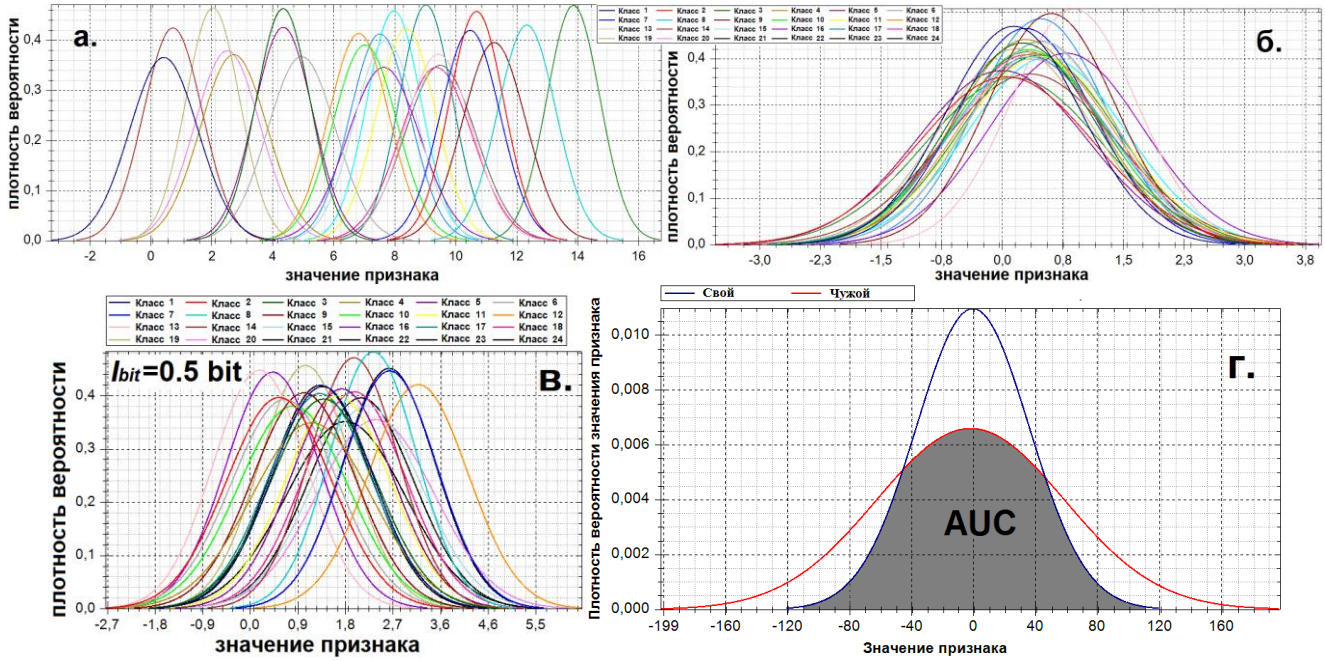


Рисунок 2.4 – Примеры ФПВ признака.

ФПВ «Свой» для 24 классов: а. I_{bit}≈1,75, б. I_{bit}≈0,15, в. I_{bit}≈0,5;

г. расчет AUC через построение ФПВ «Свой» и «Чужие»

С каждым набором данных проведена отдельная сессия по распознаванию образов в режиме верификации методом перекрестного сравнения. Каждый классификатор обучался на 25 случайных сгенерированных примерах, остальные 100 примеров использовались для тестирования. Исходя из порогового значения

для меры Минковского, принималось решение об отнесении данных к категории «Свой» или «Чужой» (рисунок 2.6). По окончанию сессии рассчитывались относительные частоты возникновения ошибок 1-го и 2-го рода. В биометрических системах эти показатели принято называть вероятностями (или процентом) ошибок «ложного отказа» (***FRR***) и «ложного допуска» (***FAR***). Сравнение биометрических систем часто выполняется по коэффициенту равной вероятности ошибок (***EER***=***FRR***=***FAR***, рисунок 2.6). Обобщенные результаты эксперимента приведены на рисунке 2.7 и 2.8 (все вероятности ошибок представлены в логарифмической шкале).

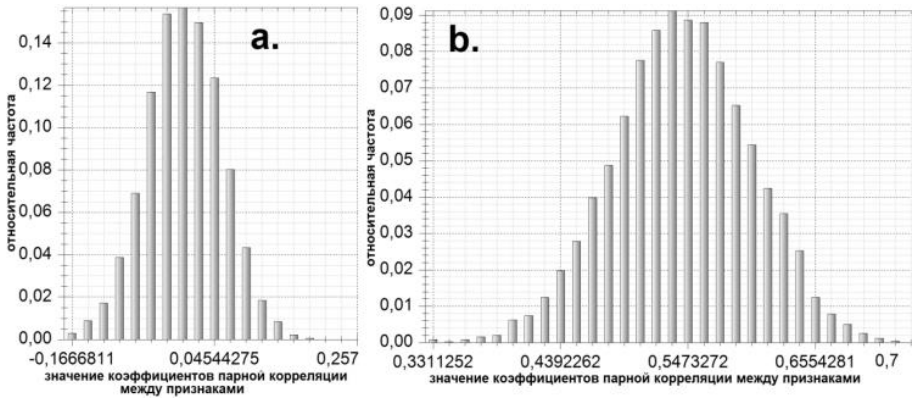


Рисунок 2.5 – Гистограммы относительных частот коэффициентов корреляции между всеми парами признаков при I_{bit}≈0,5: a. признаки независимы (C≈0,0), b. зависимы (C≈0,55)

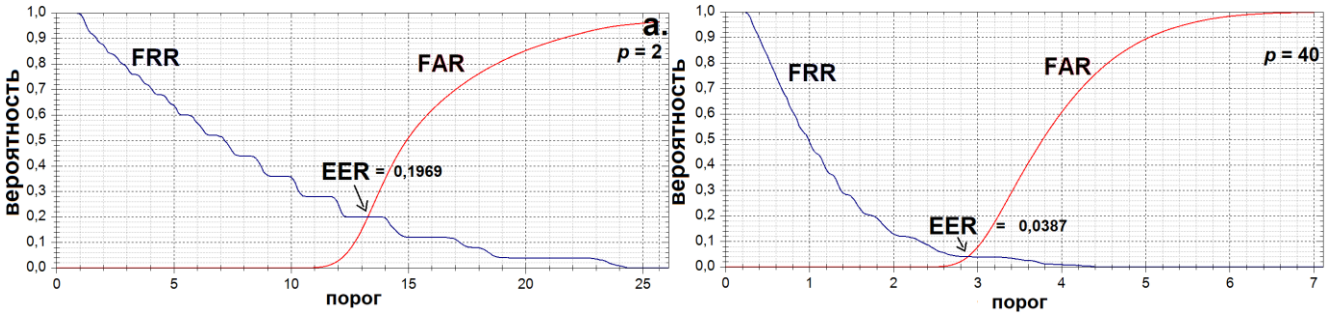


Рисунок 2.6 – Характеристические кривые, иллюстрирующие зависимость FRR и FAR от порога меры Минковского, когда признаки коррелированы (при I_{bit} ≈ 0,5, n = 100): а. g=2; б. g=40

Как можно видеть, всегда существует оптимум параметра g, при котором достигается наименьший показатель EER. Если признаки независимы, то оптимум g принадлежит интервалу [1,6; 2,4] (рисунки 2,7а, 2,8а), в этом случае

пространство признаков можно считать Евклидовым и мера близости Пирсона дает хороший результат. Чем сильней корреляция между признаками, тем больше возрастает оптимальное значение g и шире интервал, на котором лежит оптимум (C≈0,1 – [10;20], C≈0,55 – [40;50], C≈0,9 – [30;100], рисунки 2.7b, 2.8b, 2.8c).

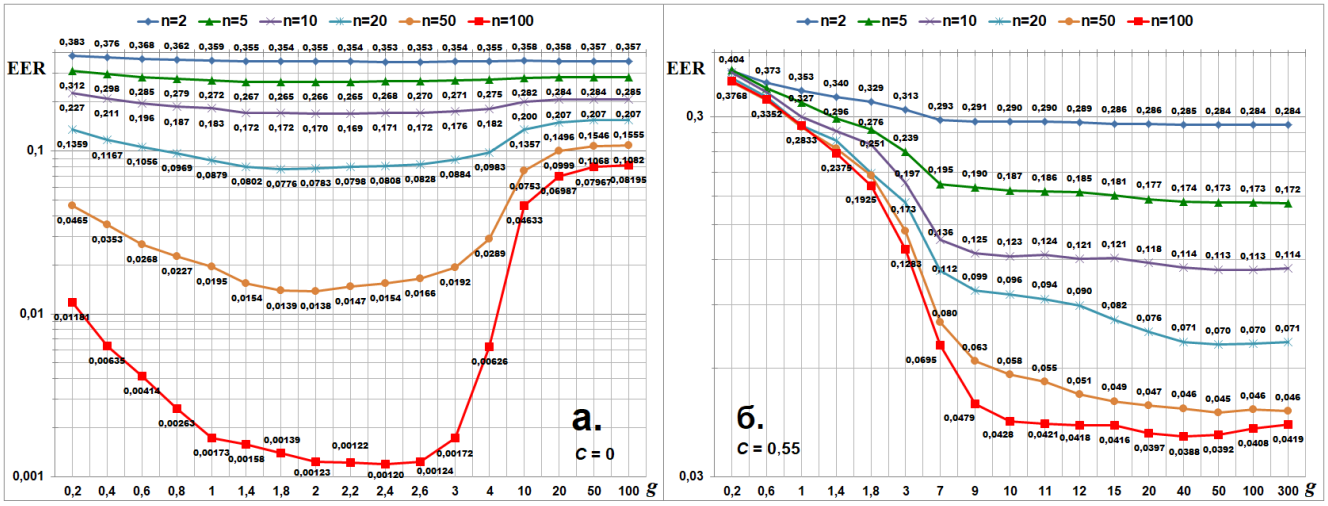


Рисунок 2.7 – Влияние коэффициента g и размерности пространства признаков n на EER (при I≈0,5): а. признаки независимы (C≈0), б. коррелированы (C≈0,55)

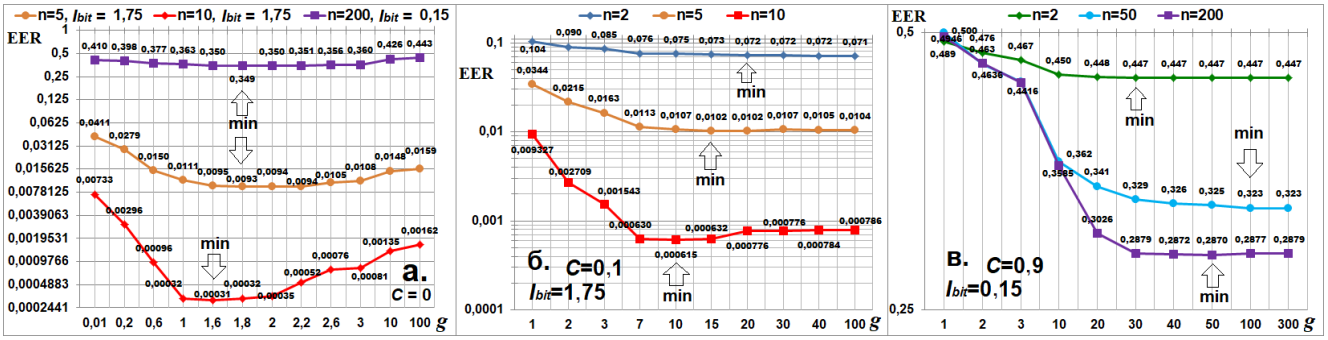


Рисунок 2.8 – Влияние g и n на EER:

а. Признаки независимы, различная информативность;

б. Весьма информативны (I_{bit}≈1,75), слабо зависимы (C≈0,1);

в. Почти неинформативны (I_{bit}≈0,15), сильно зависимы (C≈0,9)

Таким образом, если признаки зависимы, то увеличение g может привести к снижению количества ошибок распознавания в разы. *Этот важный аспект необходимо учитывать при построении любых классификаторов (нейросетевых, иммунологических, ансамблевых), где в том или ином виде используется какая-либо мера расстояний в пространстве признаков.*

Однако описанный «трюк» на практике не сработает, если «смешивать» независимые и зависимые признаки. Любое пространство признаков искривлено в различных направлениях с разной силой (корреляционная зависимость между признаками различна). В описанных экспериментах мы попытались воссоздать одинаково зависимые признаки, тем не менее, даже в этом случае имеется существенный разброс оценок коэффициентов корреляции (рисунок 2.5а). Так каждый признак помимо информации о классе образов несет в себе шум, обусловленный несоответствием его корреляционных связей значению g. Потенциальный уровень шума пропорционален объему полезной информации признака (с учетом его коррелированности с остальными), реальный уровень шума тем меньше, чем правильнее подобран параметр g. В результате эффект снижения EER наблюдается до момента информационного «насыщения», после которого шумы уже не дают уменьшить число ошибок. Например, если признаки малоинформативны (I_{bit}≈0,5), то насыщение наблюдается при n=100 ($EER_{C=0,55}$=0,038>$EER_{C=0}$=0,0012, рисунок 2.7). При более высокой информативности (I_{bit}≈1,75) сравнимый уровень надежности для зависимых и независимых признаков не обеспечивается уже при n=10 ($EER_{C=0,1}$=0,0006>$EER_{C=0}$=0,0003, рисунок 2.8). Для сильно зависимых и почти неинформативных признаков (I_{bit}≈0,15) наоборот наблюдается прирост производительности даже при n=200 ($EER_{C=0}$=0,349<$EER_{C=0,9}$=0,287, рисунок 2.8). Когда полезной информации слишком мало, независимые признаки дают больше шумов «несоответствия», чем зависимые.

Всегда существует оптимум параметра g, при котором достигается наименьший показатель ошибок, который зависит от средних показателей информативности и внутриклассовой корреляции между признаками (I и C, соответственно). *Мера Минковского позволяет точнее определять расстояния в искривленном пространстве признаков, что дает снижение количества ошибок классификации образов почти до уровня, соответствующего «плоскому» пространству.* Тем не менее, часть информации все-таки теряется и мера Минковского дает хорошие результаты, только если корреляционная зависимость

между признаками не значительная. При более сильном искривлении пространства признаков вероятность ошибочных решений становится высокой.

2.3 Извлечение мета-признаков из внутренних корреляционных связей образа. Пространства мета-признаков Байеса-Минковского

Далее впервые показано, что корреляция не только искривляет пространство признаков, но и переносит часть информации об образах в «скрытые» измерения. Речь идет об информации, касающейся уровней искривления пространства признаков в направлении каждого измерения. Чтобы извлечь данную информацию введем несколько вариаций меры Байеса-Минковского (2.2)-(2.4), которая оперирует разностями между признаками и таким образом осуществляет анализ данных, которые заключены между измерениями исходного пространства признаков. Эти метрики принимают тем меньшие значения, чем выше $C_{j,t}$ (рисунок 2.9). Если же t-й и j-й признаки линейно зависимы ($C_{j,t}$=1) для образов определенного класса, то t-е и j-е измерения относительно этого класса становятся «сингулярными» (объединяются в одно измерение, рисунок 2.3), и соответствующая разность под знаком модуля всегда принимает нулевое значение (будто j-го измерения нет). Но если признаки имеют слабую зависимость, то разность по модулю увеличивается. Чем выше корреляция между признаками, тем меньший процент неверных решений будет получен. На рисунке 2.9 видно: $AUC_{|C|>0,95}(\Phi_G(y),\Phi_I(y))<\mathrm{AUC}_{|C|<0,3}(\Phi_G(y),\Phi_I(y))$.

$$y=\sum_{j=1}^{n}\left|\left|\frac{(m_t-a_t)}{\sigma_t}\right|^g-\left|\frac{(m_j-a_j)}{\sigma_j}\right|^g\right|,\ j\neq t,\qquad y=\sqrt[g]{\sum_{j=1}^{n}\left|\left|\frac{(m_t-a_t)}{\sigma_t}\right|^g-\left|\frac{(m_j-a_j)}{\sigma_j}\right|^g\right|},\ j\neq t, \tag{2.2}$$

$$y=\sum_{j=1}^{n}\left|\left|\frac{\mu_t-a_t}{\delta_t}\right|^g-\left|\frac{\mu_j-a_j}{\delta_j}\right|^g\right|,\ j\neq t,\qquad y=\sqrt[g]{\sum_{j=1}^{n}\left|\left|\frac{\mu_t-a_t}{\delta_t}\right|^g-\left|\frac{\mu_j-a_j}{\delta_j}\right|^g\right|},\ j\neq t, \tag{2.3}$$

$$y=\sum_{j=1}^{n}\left|\left|\frac{a_t}{\delta_t}\right|^g-\left|\frac{a_j}{\delta_j}\right|^g\right|,\ j\neq t,\qquad y=\sqrt[g]{\sum_{j=1}^{n}\left|\left|\frac{a_t}{\delta_t}\right|^g-\left|\frac{a_j}{\delta_j}\right|^g\right|},\ j\neq t, \tag{2.4}$$

где μ_j и δ_j – это нормирующие коэффициенты, вычисляемые как математическое ожидание и среднеквадратичное отклонение значений признака для класса «Чужие». Смысл коэффициентов μ_j и δ_j заключается в приведении всех признаков примерно к единому масштабу (хотя для образов разных субъектов области значений признаков все равно будут отличаться), μ_j и δ_j не компрометируют данные какого-либо пользователя, так как представляют параметры распределения значений признака для множества обезличенных образов. Таким образом, *при использовании метрик* (2.3) *и* (2.4) *обеспечивается дифференциальная конфиденциальность*. Мера (2.4) дает наиболее высокий уровень конфиденциальности, так как она оперирует только нормирующими коэффициентами разброса δ_j, поэтому в рассматриваемых задачах ее применение предпочтительнее. Для усиления конфиденциальности меры (2.3) можно добавить к μ_j шум (случайное смещение значения).

Размерность пространства мета-признаков Байеса-Минковского составляет (2.5):

$$n' = 0{,}5(n(n\text{-}1)) = 0{,}5n^2 - 0{,}5n \qquad (2.5)$$

Под мета-признаками подразумеваются разности вида (2.6)-(2.8):

$$a'_{j^*} = a'_{t,j} = f(a_t, a_j) = \left| \left| \frac{(m_t - a_t)}{\sigma_t} \right|^g - \left| \frac{(m_j - a_j)}{\sigma_j} \right|^g \right|, \qquad (2.6)$$

$$a'_{j^*} = a'_{t,j} = f(a_t, a_j) = \left| \left| \frac{\mu_t - a_t}{\delta_t} \right|^g - \left| \frac{\mu_j - a_j}{\delta_j} \right|^g \right|, \qquad (2.7)$$

$$a'_{j^*} = a'_{t,j} = f(a_t, a_j) = \left| \left| \frac{a_t}{\delta_t} \right|^g - \left| \frac{a_j}{\delta_j} \right|^g \right|, \qquad (2.8)$$

$$j > t,\ j^* = \sum_{t^*=1}^{t-1} (n - t^*) + j - t$$

которые фактически являются грубыми (точечными) оценками корреляционной зависимости между двумя исходными признаками под номерами j и t (чем меньше по модулю a', тем выше внутриклассовая корреляция между соответствующими признаками, если $C_{j,t}$=1, то $a'_{t,j} \approx 0$). Под точечной оценкой понимается оценка, сделанная всего по одному примеру тестовой выборки, но при наличии некоторых

априорных знаний (m_j σ_j, δ_j, μ_j), полученных в процессе обучения на выборке небольшого объема.

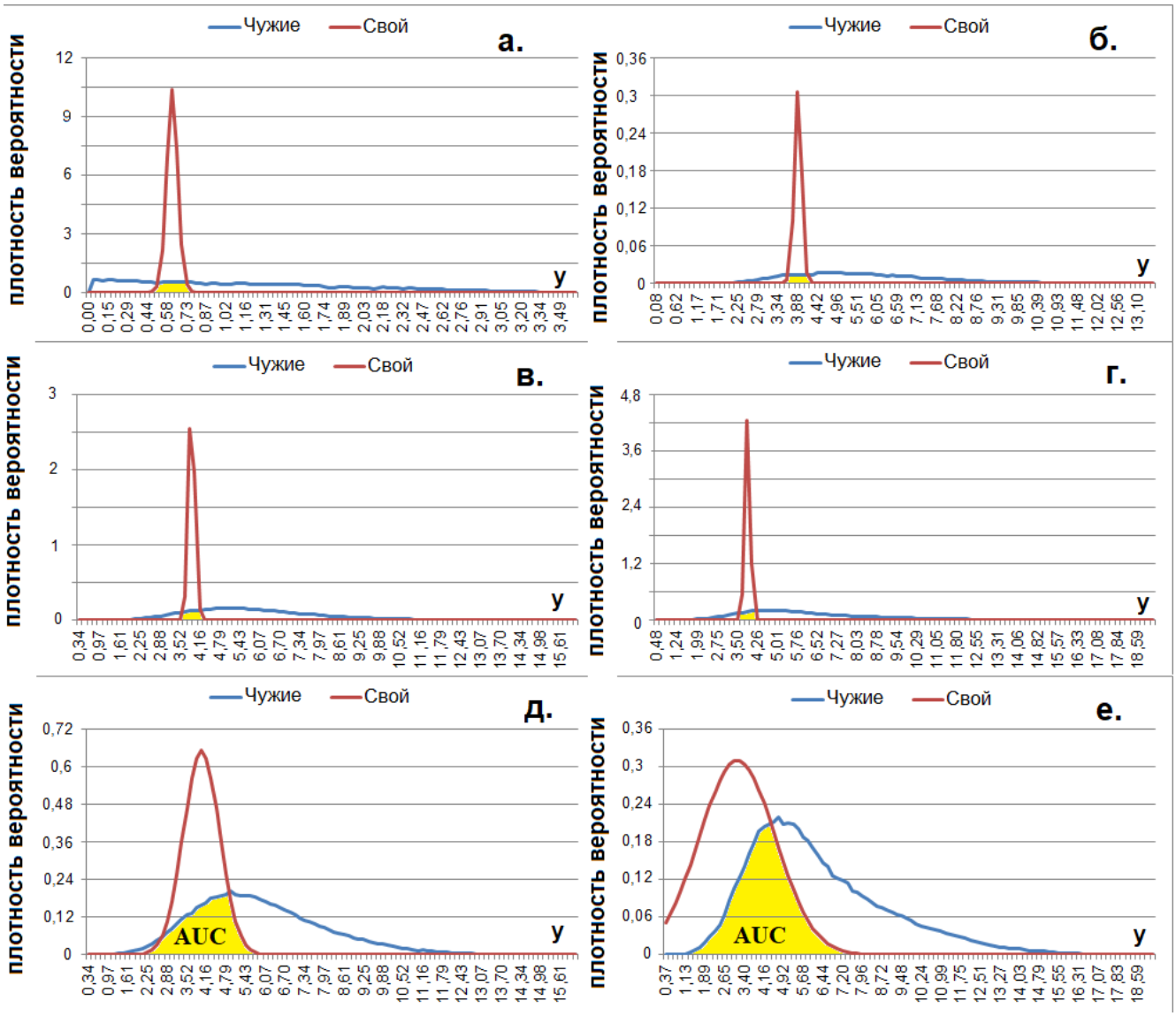


Рисунок 2.9 – Графики плотностей вероятности значений меры (4) (без квадратного корня) при g=1, I≈1,75 бит:

а. для всех классов 1> $C_{j,t}$ >0,95, n'=1;

б. для всех классов 1> $C_{j,t}$ >0,95, n'=5;

в. для класса «Свой» 1> $C_{j,t}$ >0,95, для класса «Чужие» $|C_{j,t}|$ < 0,3, n'=5;

г. для класса «Свой» 1> $C_{j,t}$ >0,95, для класса «Чужие» -1< $C_{j,t}$ < -0,95, n'=5;

д. для всех классов $|C_{j,t}|$ < 0,3, n'=5;

е. для всех классов -1< $C_{j,t}$ < -0,95, n'=5;

Как можно видеть, меры (2.2)-(2.4) являются линейными классификаторами в пространстве мета-признаков Байеса-Минковского, однако в нем можно строить

классификаторы любой сложности, например, искусственные нейронные сети. Для этого нужно применить одно из возможных отображений $a'_{j*}=f(a_t,\ a_j)$ исходного признакового пространства в спрямляющее пространство Байеса-Минковского. Удобнее всего изобразить трехмерное пространство (рисунок 2.10), так как $n'=n=3$. Пространство мета-признаков может содержать значительно больше информации о классах образов, чем исходное пространство. Из примера на рисунке 2.10 можно видеть, что в исходном пространстве классы линейно неразделимы и информативность признаков очень низкая ($0,1<I_j<0,2$), но мета-признаки оказываются гораздо информативнее ($0,35\leq I'_{j*}\leq 2,95$). Кроме того, исходные признаки сильно коррелированы ($0,94\leq C_{j,t}\leq 0,96$), в то время как корреляция между мета-признаками незначительна ($0,1\leq C'_{j*,t*}\leq 0,22$).

Интересным свойством исследуемого пространства является то, что отрицательно коррелированные пары признаков образуют в нем, как правило, либо положительно либо отрицательно коррелированные мета-признаки. На рисунке 2.11 иллюстрируется, что два положительно коррелированных признака (нормированных по δ_j) образуют мета-признак a'_2 с хаотичной динамикой (который не имеет заметной корреляции с другими мета-признаками). Отрицательно коррелированные признаки образуют мета-признаки a'_1 и a'_3, которые являются положительно коррелированными (относительно друг друга) для класса 1 и отрицательно коррелированными для класса 3. Для класса 2 a'_1 и a'_3 имеют неявную корреляционную связь – в определенный момент положительная корреляция меняется на отрицательную.

Чтобы избавиться от отрицательной корреляции следует неоднократно использовать одно из отображений (2.6)-(2.8) – сначала по отношению к парам отрицательно коррелированных признаков (мета-признаков), потом по отношению к парам положительно коррелированных мета-признаков. Пространство мета-признаков второго порядка (после повторного «перехода») имеет еще большую размерность:

$$n'' = 0,5(n'(n'-1)) = 0,5n'^2 - 0,5n' = 0,5(0,5n^2 - 0,5n)^2 - 0,5(0,5n^2 - 0,5n)$$

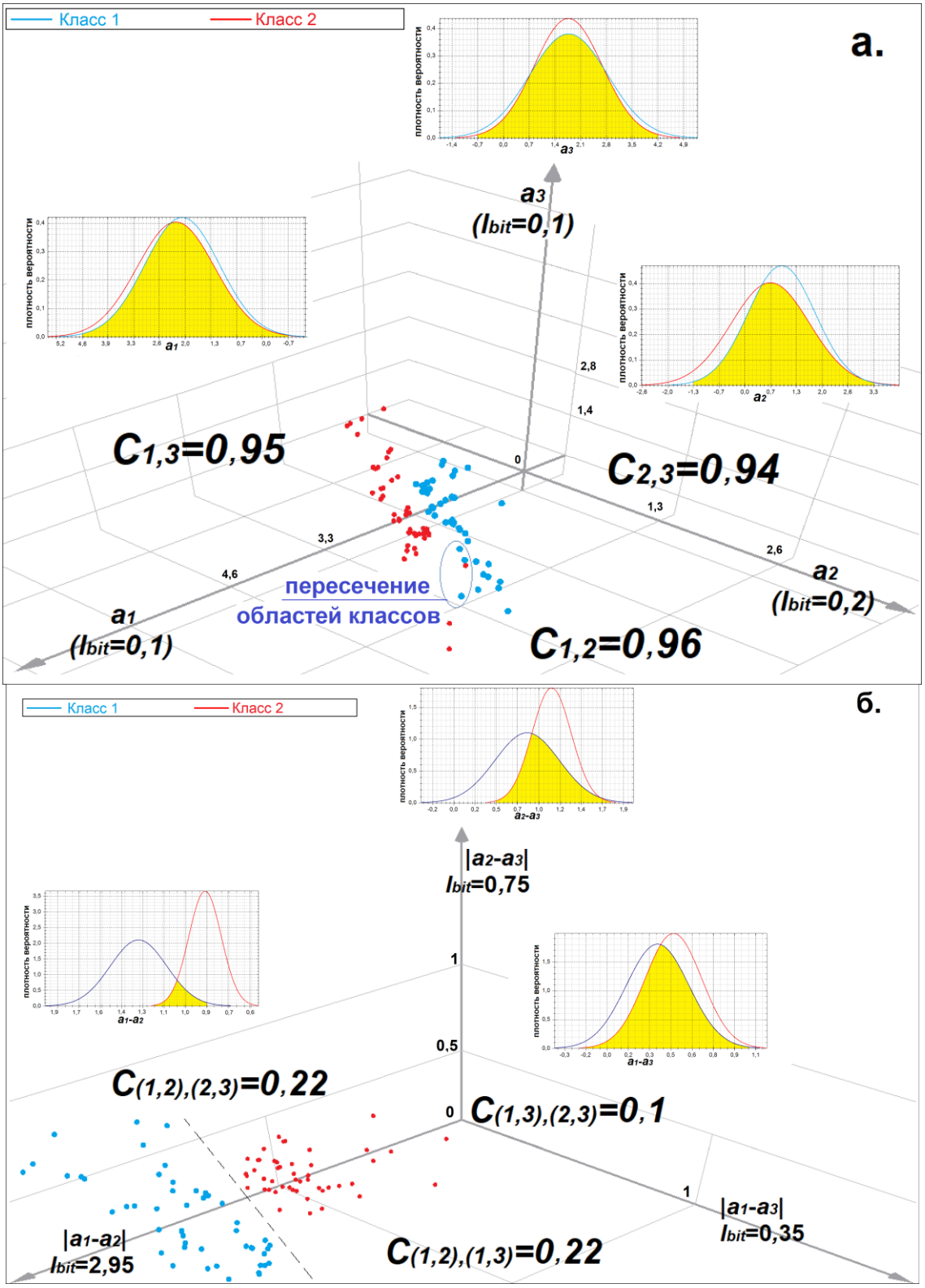


Рисунок 2.10 – Пространства трех признаков и графики плотностей вероятности их значений:

а. исходное пространство положительно коррелированных признаков,

б. пространство мета-признаков Байеса-Минковского, производное от исходного путем применения отображения (8) при g=1.

По этой причине информативность отрицательных корреляционных связей исходных признаков может быть значительно выше, чем положительных. Дальнейший «переход» (построение пространств мета-признаков третьего, четвертого и т.д. порядков) имеет смысл, пока остаются коррелированные пары мета-признаков (обычно после двух-трех «переходов» корреляция между всеми парами мета-признаков является почти нулевой или слабой по шкале Чеддока).

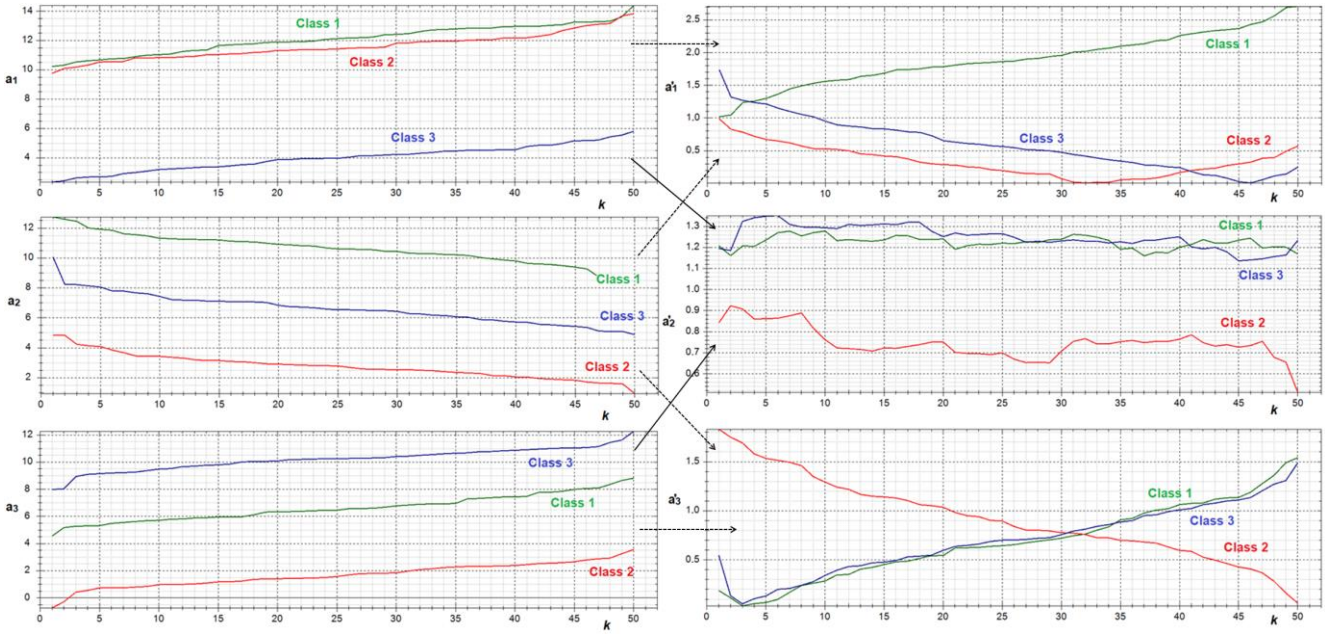


Рисунок 2.11 – Исходные признаки (слева) и мета-признаки (справа), порожденные путем отображения (2.8) при g=0,9

Таким образом, при построении классификаторов можно использовать мета-признаки, полученные из пар положительно и отрицательно коррелированных признаков. При этом следует избегать использования мета-признаков, порожденных парами независимых (слабо коррелированных) признаков (либо мета-признаков), так как такие порождения могут представлять собой шум. Независимые (слабо коррелированные) признаки следует обрабатывать отдельно, не производя над ними преобразований (2.6)-(2.8).

Считается, что «наивная» схема классификации Байеса является полностью корректной, когда признаки независимы, т.е. пространство признаков абсолютно не имеет кривизны. Мера Минковского наоборот измеряет расстояние в искривленном пространстве. Новые метрики преобразуют пространство

коррелированных признаков в пространство независимых мета-признаков, поэтому они названы метриками Байеса-Минковского.

2.4 Симметрия корреляционных связей

Остановимся также на таком важном аспекте, как симметрия и асимметрия корреляционных связей. Под симметрией понимается, когда при обработке нейроном группы признаков, уровни их взаимной корреляционной зависимости оказываются примерно одинаковыми. Близкие корреляционные связи также наблюдаются между группами признаков, обрабатываемых другими нейронами. Асимметрия корреляционных связей напротив означат, что признаки, извлекаемые из образа и поступающие на вход нейрону, имеют существенно отличающиеся уровни взаимной корреляционной зависимости.

Полная симметрия корреляционных связей неблагоприятно сказывается на эффективности большинства мер близости (в частности, квадратичных [136]), однако является крайне желательным свойством для мер близости Байеса-Минковского. Следует отметить, что классический коэффициент парной корреляции Пирсона (2.1) можно рассматривать, как одну из форм записи правила Байеса [14, 136]:

$$P(a_2/a_1)\cdot P(a_1) = P(a_1/a_2)\cdot P(a_2)$$

В этом легко убедиться, рассматривая предельные значения коэффициентов корреляции:

$$\begin{cases} C(a_1,a_2)=1 \Rightarrow P(a_1)=P(a_2) \vee P(a_1/a_2)=P(a_2/a_1)=1; \\ C(a_1,a_2)=0 \Rightarrow P(a_1,a_2)=P(a_1)\cdot P(a_2). \end{cases}$$

При $C > 0{,}7$ можно записать следующее приближенное равенство:

$$C(a_2,a_1)\cdot P(a_1) \approx C(a_1,a_2)\cdot P(a_2)$$

Данное приближенное равенство становится точным равенством при высоких значениях коэффициентов корреляции. Это приближенное равенство указывает на связь между коэффициентами парных корреляций и правила Байеса.

Используя эту связь, было предложено множество так называемых многомерных метрик Байеса, в частности корреляционные двумерные, трехмерные и многомерные функционалы:

$$y(a_1, a_2, a_3) = \\ = \frac{C(a_1, a_2) + C(a_1, a_3) + rC(a_2, a_3)}{3}$$

$$y(a_1, a_2, a_3, a_4) = \\ = \frac{C(a_1, a_2) + C(a_1, a_3) + C(a_1, a_4)}{6} + \\ + \frac{C(a_2, a_3) + C(a_2, a_4)}{6} + \frac{C(a_3, a_4)}{6}$$

$$y(a_1, a_2, ..., a_n) = \\ = \{C(a_1, a_2) + C(a_1, a_2) + ... + C(a_1, a_n) + \\ + C(a_2, a_3) + ... + C(a_2, a_n) + \\ ::::::::::: \quad ::::::::::: \quad ::::::::::: \quad ::::::::::: \quad :::::::: \\ + C(a_{n-1}, a_n)\} \cdot \frac{1}{(n-1) + (n-2) + ... + (n-n)},$$

а также разностные и гиперболические многомерные функционалы:

$$y = \sum_{j=1}^{n} \sum_{i=1}^{n} \left| \frac{|m_i - a_i|}{\sigma_i} - \frac{|m_j - a_j|}{\sigma_j} \right|$$

$$y = \sum_{t=1}^{n} \left| \frac{|m_t - a_t|}{\sigma_t} - \frac{|m_j - a_j|}{\sigma_j} \right|$$

$$y = \sum_{t=1}^{n} \left(\frac{(m_t - a_t)^2}{\sigma^2_t} - \frac{(m_j - a_j)^2}{\sigma^2_j} \right)^2$$

Очевидно, что многомерные разностные и гиперболические функционалы Байеса являются частным случаем метрик Байеса-Минковского, компрометирующих знания. Установлено [14, 66, 97, 136], что симметризация корреляционных связей является желательным свойством для многомерных сетей Байеса-Хэмминга. Исходя из этого, можно сделать аналогичные выводы и о корреляционных нейронах Байеса-Минковского. Таким образом, предложенный алгоритм обучения НПБК будет работать эффективнее, если использовать множество интервалов коррелированности признаков при настройке нейронов.

Важнейшим преимуществом метрик, лежащих в основе корреляционных нейронов, является возможность эффективной обработки отрицательно коррелированных признаков, чего не удается достичь с помощью разностных и гиперболических многомерных функционалов Байеса. Кроме того, предложенные метрики Байеса-Минковского не компрометируют знания ИИ, в отличие от многомерных функционалов Байеса.

2.5 Оценка информативности мета-признаков с использованием синтетических наборов данных. Свойства пространств мета-признаков Байеса-Минковского

Проведен аналогичный эксперимент по распознаванию образов мерой Байеса-Минковского (2.2). Из представленных данных (рисунок 2.12) видно, что оптимум g меняется в каждом рассмотренном случае. Если признаки независимы, то минимум по EER достигается при $g>1$, если существенно коррелированы – при $0,5<g<1$, при слабой корреляции между признаками наблюдается 2 экстремума EER – первый при $g\epsilon[0,7;0,9]$, второй при $g\geq 100$.

Динамика изменения EER(g) для меры Байеса-Минковского имеет обратную тенденцию по сравнению с показателями EER(g) для меры Минковского. Если признаки коррелированы, мера Байеса-Минковского дает более высокий результат, чем, если признаки независимы. Причем, *вероятности ошибок распознавания в пространстве коррелированных признаков для меры Байеса-Минковского ниже, чем уровень ошибок для меры Минковского в случае независимости признаков* (рисунки 2.7, 2.8, 2.12). Для меры Байеса-Минковского также характерно следующее: *чем выше уровень корреляции, тем больше снижается уровень ошибок* по сравнению со случаем независимости признаков. Таким образом, мера Байеса-Минковского является «антагонистом» по отношению к мере Минковского, так как обладает противоположными свойствами.

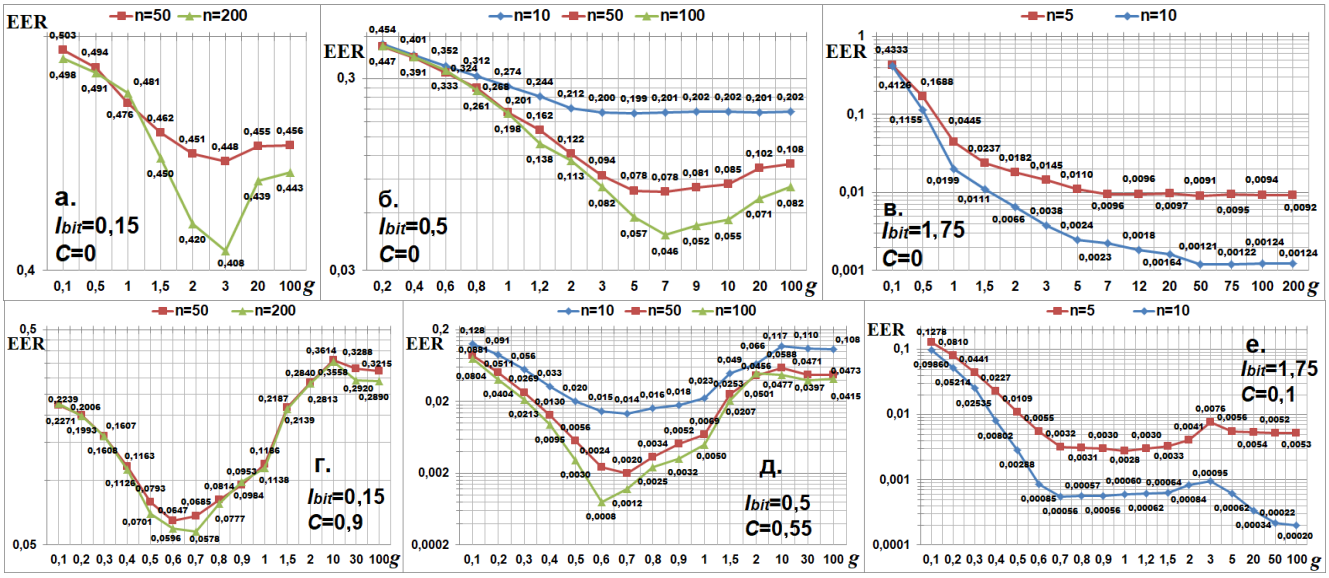


Рисунок 2.12 – Влияние g и n на EER:

а. Признаки независимы и почти не информативны;

б. Независимы и малоинформативны;

в. Независимы и весьма информативны;

г. Сильно коррелированы и почти неинформативны;

д. Значительно коррелированы и малоинформативны;

е. Весьма информативны и слабо коррелированы

Проведен еще один вычислительный эксперимент по распознаванию образов в пространстве абстрактных (имитированных) признаков. Все признаки имели нормальное распределение значений. Генерировалось по 65 классов образов в пространствах независимых и зависимых признаков с различными показателями I. Генерируемые классы образов отличались между собой параметрами распределения значений признаков. Методика генерации признаков и образов была аналогичной (основана на методе Монте-Карло и описана в [330]).

С использованием «наивного» байесовского классификатора проведен вычислительный эксперимент по идентификации сгенерированных образов на закрытом множестве из 65 классов в различных пространствах признаков (и соответствующих им пространствах мета-признаков). Для обучения байесовского классификатора вычислялись параметры распределения нормального закона (мат. ожидание и среднеквадратичное отклонение) каждого признака или мета-

признака на основе данных тренировочной выборки (по 10 случайных примеров на класс). Вместо условных вероятностей использовались условные плотности вероятности (что является общепринятым подходом [246]), которые вычислялись в соответствии с нормальным законом распределения. Для тестирования использовалось по 100 примеров от каждого класса, не вошедших в тренировочный набор. Решение принималось в пользу гипотезы с наивысшей апостериорной вероятностью. Далее вычислялась точность первого порядка (rank-1 accuracy): подсчитывалось количество верных классификационных решений, которое делилось на общее количество опытов. Результаты тестирования представлены на рисунке 2.13. Этот эксперимент наглядно демонстрирует следующее:

- корреляция между признаками может нести больше информации, чем сами признаки. Если исходные признаки более информативны (I≈0,5) и независимы ($|C_{j,t}| < 0,3$), то в пространстве мета-признаков точность идентификации образов ниже, чем в случае, когда исходные признаки менее информативны (I≈0,15), но сильно коррелированы ($1 > C_{j,t} > 0,95$);
- если признаки независимы, то мета-признаки могут вносить шум (точность идентификации выше, если использовать только исходные независимые признаки, чем, если объединить независимые признаки с мета-признаками);
- «переход» в пространство мета-признаков не ведет к проявлению проблемы «проклятья размерности», если признаки сильно коррелированы. Проклятие размерности — это проблема, связанная с экспоненциальным ростом объема обучающей выборки и связанных с этим вычислений из-за линейного роста размерности пространства признаков. Мы видим (рисунок 2.13), что при использовании аналогичной обучающей выборки (10 примеров) удается достичь более высокой точности, если перейти в пространство большей размерности (n'=435) по сравнению с исходным (n=30). При этом количество вычислений при расчете апостериорных («байесовских») вероятностей растет линейно по отношению к увеличению размерности пространства признаков,

а количество вычислений при расчете корреляционной матрицы и количество признаков растет не по экспоненте, а по степенному закону (2.5).

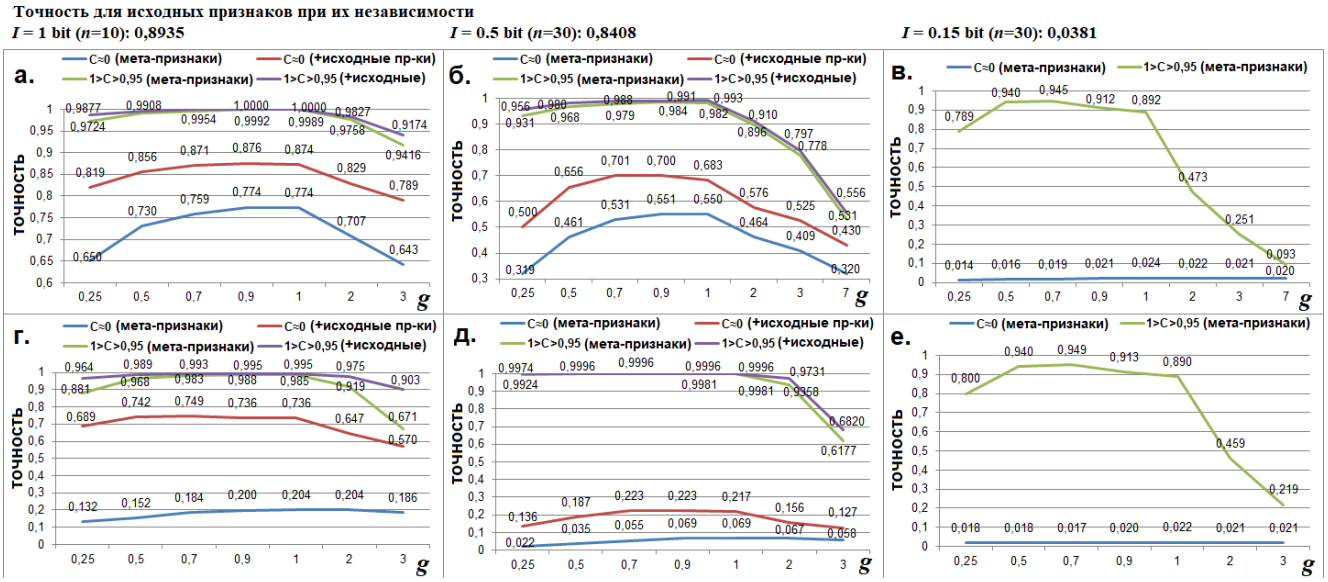


Рисунок 2.13 – Изменение точности идентификации образов на основе «наивного» классификатора Байеса при различной средней информативности мета-признаков:

- при использовании отображения (2.8):

а. I≈1 бит, n=10 и n'=45,

б. I≈0,5 бит, n=30 и n'=435,

в. I≈0,15 бит, n=30 и n'=435,

- при использовании отображения (2.7):

г. I≈1 бит, n=10 и n'=45,

д. I≈0,5 бит, n=30 и n'=435,

е. I≈0,15 бит, n=30 и n'=435

Из результатов моделирования видно, что оптимум по точности распознавания образов достигается при $0{,}7 \le g \le 1$ (в зависимости от уровня информативности признаков). В дальнейших экспериментах решено использовать g=0,9 при переходе в пространство мета-признаков первого порядка, что в большинстве случаев должно давать более высокий результат, кроме того при $g \ne 1$ создается нелинейность преобразования признаков в мета-признаки (что усложняет анализ логики работы ИИ злоумышленником). Значительного выигрыша по точности распознавания при использовании отображения (2.7) не наблюдалось.

Также проведена оценка информативности мета-признаков в зависимости от информативности и коррелированности исходных признаков (рисунок 2.14). Из приведенных данных можно видеть, что экстремум информативности мета-признаков также наблюдается при $g \approx 0{,}9$. Для независимых признаков $I' < I$ (при $g > 1$ $I' \to I$). Однако если признаки коррелированы, то $I' > I$ (в 2-3 раза). Мы видим, что корреляционная связь между двумя признаками, может быть информативнее, чем пара признаков, породивших ее.

Описанные закономерности справедливы, если признаки имеют нормальное распределение. Для других законов распределения оценка не проводилась.

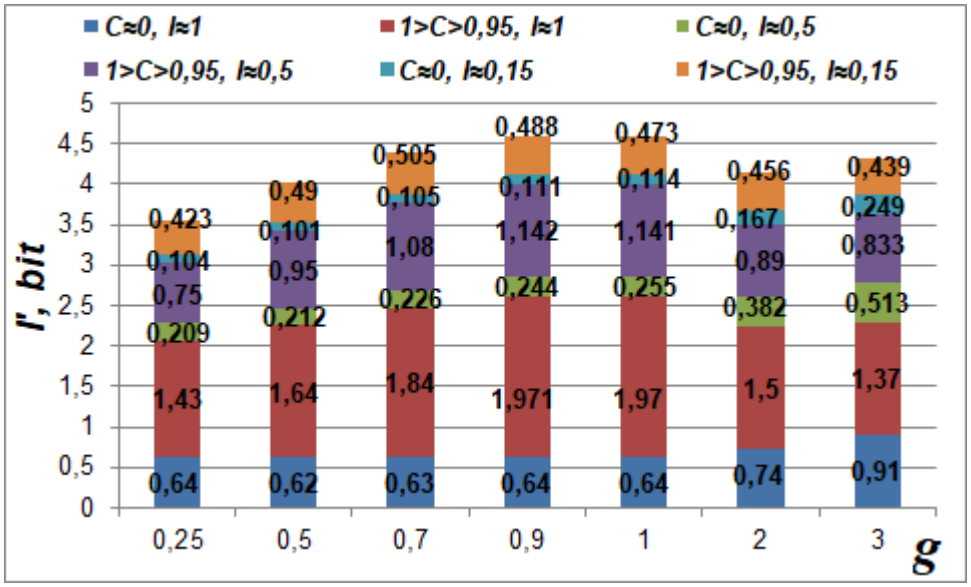


Рисунок 2.14 – Информативность мета-признаков I', порождённых отображением (2.8)

Таким образом, *чем выше уровень корреляции, тем информативнее мета-признаки Байеса-Минковского*. На первый взгляд это противоречит классической теории математической статистики, которая утверждает: корреляция между признаками указывает на то, что часть информации в признаках повторяется. Но учитывая полученные результаты, данное утверждение стоит уточнить: *пространство признаков искривляется из-за корреляционных связей, причем относительно каждого класса образов и каждого измерения характер искривления различен, что влечет за собой уменьшение количества информации об отличии классов, наблюдаемой в пространстве признаков, и появление новой информации в пространстве Байеса-Минковского, но в большем объеме.*

Приведенные данные свидетельствуют о перспективности использования пространств мета-признаков Байеса-Минковского, по крайней мере, в задачах классификации образов.

2.6 Модель разностного корреляционного нейрона Байеса-Минковского

Мы исходим из того, что каждый нейрон должен разделять входные данные по уровню коррелированности. Для этого нейрон соединяется с мета-признаками, которые были порождены парами признаков с близким уровнем взаимной корреляции. Введем два уровня коррелированности признаков: $C_{-}>C_{j,t}$ ($C_{-}\in$[-0,95;-0,3]) и $C_{+}<C_{j,t}$ ($C_{+}\in$[0,3; 0,95]). При $|C_{-+}| \geq 0{,}3$ корреляционный нейрон будет работать неверно и количество ошибок будет значительным. Условие $|C_{-}|=|C_{+}|$ не обязательно должно выполняться, чем больше отрицательно и положительно коррелированных пар признаков, тем выше по модулю следует задавать пороговые коэффициенты C_{-} и C_{+}, соответственно. Необходимо учитывать, что один мета-признак может быть связан только с одним корреляционным нейроном во избежание реализации атак, основанных на поиске общих связей нейронов [196, 287]. Таким образом, корреляционные нейроны являются частично связными.

Метрика (2.9) отлично справляется с выделением положительно коррелированных данных на фоне данных с любой корреляцией, но не может определить отрицательно коррелированные данные (рисунок 2.9). Метрика среднеквадратичного отклонения значений мета-признаков (2.10) позволяет отделить как положительно коррелированные, так и отрицательно коррелированные данные (рисунок 2.15). Это происходит потому, что при сильной корреляции между исходными признаками (как положительной, так и отрицательной) значения модулей отклонений $|a'_{j*} - m'|$ имеют тенденцию к снижению.

$$y = \sum_{j^*=1}^{n'} a'_{j^*}, \tag{2.9}$$

$$y = \sqrt{\frac{1}{n'} \sum_{j^*=1}^{n'} \left(a'_{j^*} - m'\right)^2}, m' = \frac{1}{n'} \sum_{t^*=1}^{n'} a'_{t^*} \tag{2.10}$$

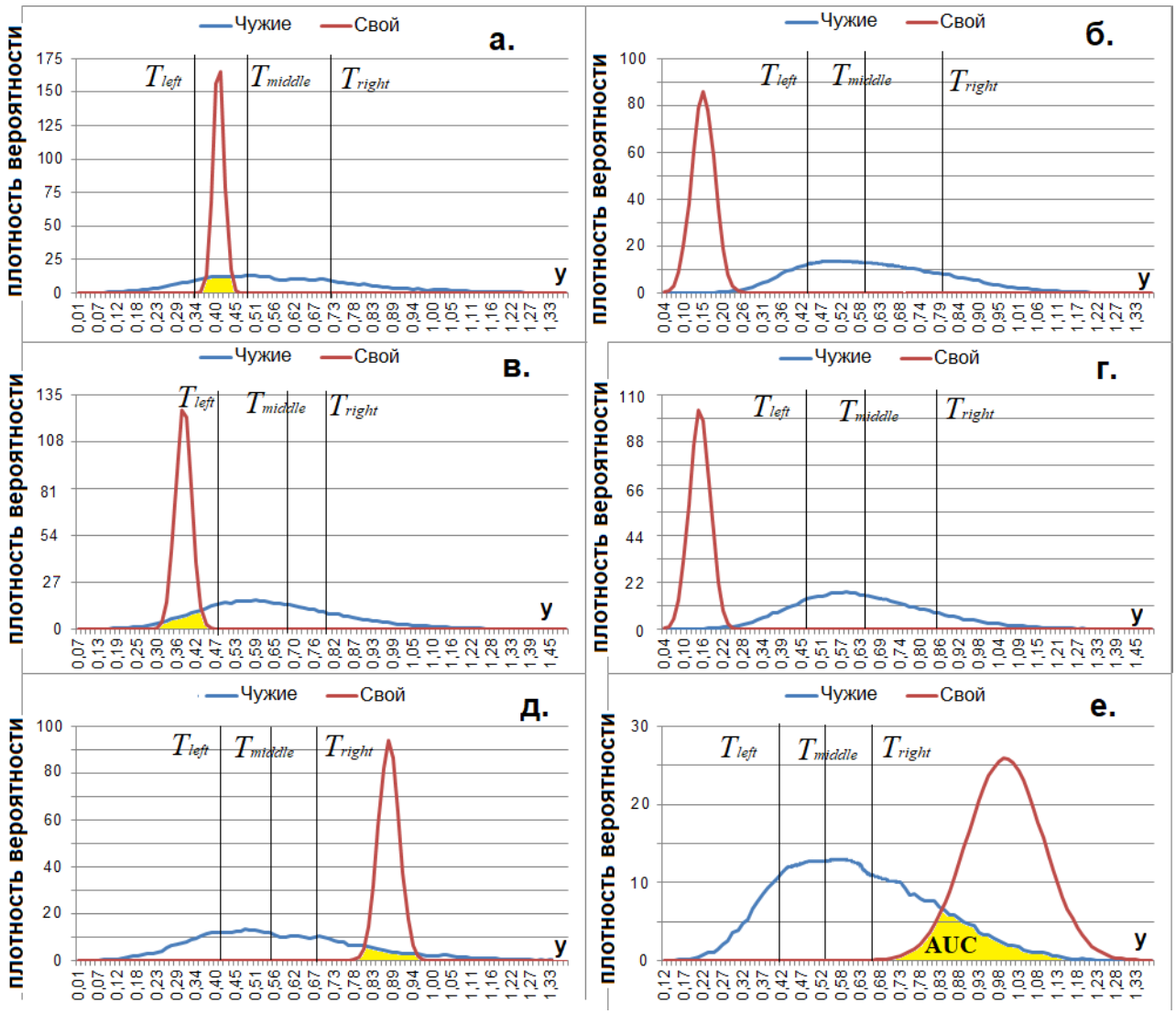


Рисунок 2.15 – Графики плотностей вероятности значений меры (2.10) после отображения (2.8) при g=1, I≈1,75 бит:

а. для всех классов 1> $C_{j,t}$ >0,95, n'=10;

б. для всех классов -1< $C_{j,t}$ < -0,95, n'=10;

в. для классов «Свой» 1> $C_{j,t}$ >0,95, для класса «Чужие» $|C_{j,t}|$ < 0,3, n'=10;

г. для классов «Свой» -1< $C_{j,t}$ < -0,95, для класса «Чужие» $|C_{j,t}|$ < 0,3, n'=10;

д. для всех классов 1> $C_{j,t}$ >0,95, n'=10 (класс «Свой» расположен справа);

е. для всех классов -1< $C_{j,t}$ < -0,95, n'=10 (класс «Свой» расположен справа).

Таким образом, разностный корреляционный нейрон Байеса-Минковского может быть основан на метрике взвешенного среднеквадратичного отклонения (2.11):

$$y = \sqrt{\frac{1}{\eta}\sum_{j^*=1}^{n'} w_{j^*}\left(a'_{j^*} - m'\right)^2} = \sqrt{\frac{1}{\eta}\sum_{\iota=1}^{\eta} w_\iota\left(a'_\iota - m'\right)^2}, m' = \frac{1}{\eta}\sum_{\iota=1}^{\eta} a'_\iota \qquad (2.11)$$

где η – количество входов нейрона, w_{j^*} – вес синапса под номером j^* ($w_{j^*} \geq 0$, если $w_{j^*}=0$, то j^*-й мета-признак не влияет на сумму, т.е. не соединяется с нейроном), ι – номер мета-признака без учета синапсов с нулевым весом (для их сквозной нумерации внутри нейрона). Метрика (2.11) на самом деле осуществляет неявный «переход» в одно из возможных пространств мета-признаков Байеса-Минковского второго порядка ($a''_\iota = (a'_\iota - m')^2$), но только для связанных с нейроном мета-признаков. Вес синапса рассчитывается по формуле (2.12):

$$w_\iota = \frac{|m''_{(G),\iota} - m''_{(I),\iota}|}{\sigma''_{(G),\iota} \cdot \sigma''_{(I),\iota}}, \qquad (2.12)$$

где $m_{(G),\iota}''$, $m_{(I),\iota}''$ – математические ожидания, а $\sigma_{(G),\iota}''$, $\sigma_{(I),\iota}''$ – среднеквадратичные отклонения значений ι-го мета-признака второго порядка ($a''_\iota = (a'_\iota - m')^2$) для образов «Свой» и «Чужие», соответственно, рассчитанные по данным обучающей выборки. По сути, вес w_ι является упрощенной, более быстрой оценкой информативности соответствующего мета-признака. После обучения нейрона параметры $m_{(G),\iota}''$, $m_{(I),\iota}''$, $\sigma_{(G),\iota}''$, $\sigma_{(I),\iota}''$ должны быть незамедлительно удалены.

В качестве функции активации в настоящей работе предлагается использовать многоуровневую пороговую функцию квантования (2.13):

$$\phi(y) = \begin{cases} 3, y < T_{left} \\ 2, T_{left} \leq y < T_{middle} \\ 1, T_{middle} \leq y < T_{right} \\ 0, y \geq T_{right} \end{cases}, \qquad (2.13)$$

где T_{left}, T_{middle} и T_{right} – левый, средний и правый пороговые значения активации нейрона (рисунок 2.15). В соответствии с предлагаемой моделью нейрон имеет четыре варианта активации {0, 1, 2, 3} и только одно из них соответствует гипотезе «Свой», остальные соответствуют гипотезе «Чужие». *О том, какое именно состояние активации соответствует гипотезе «Свой» (далее ϕ_G),*

известно только на этапе синтеза и обучения НПБК, злоумышленник не обладает этой информацией, так как она не сохраняется после настройки нейрона.

Идеальное решение задачи обучения нейрона состоит в том, чтобы при поступлении образа «Свой» на выходе нейрона почти всегда возникало определенное состояние, а в других случаях состояния {0, 1, 2, 3} на выходе нейрона стали равновероятны: P(0) ≈ P(1) ≈ P(2) ≈ P(3) ≈ 0,25, где P($\phi(y)$) – это относительная частота появления $\phi(y)$ при поступлении на вход образа «Чужой». Однако на практике достичь такого соотношения очень сложно. Для обеспечения высокой энтропии выходов нейронов в ответ на образы «Чужие» достаточно придерживаться следующего соотношения: 0,1<P($\phi(y)$)<0,4.

При вычислении порогов сначала рассчитываются вероятные граничные значения откликов нейрона y на обучающие примеры «Свой» (y_{Gmin}, y_{Gmax}) и «Чужие» (y_{Imin}, y_{Imax}) по формуле (2.14), а также значения соответствующих функций распределения $F_G(y)$ и $F_I(y)$ (2.15). В первом приближении закон распределения случайной величины y (2.11) близок к нормальному (2.14), что подтверждено методом хи-квадрат на больших выборках сгенерированных данных. Исходя из гипотезы о нормальном распределении y, каждый нейрон будет давать ложный отказ пользователям «Свой» с вероятностью в среднем не превышающей 0,002. Однако в силу наличия корреляции между откликами различных нейронов показатели FRR и FAR невозможно просчитать заранее без проведения численного эксперимента.

$$y_{min} = \xi - 4\,\varsigma, \quad y_{max} = \xi + 4\,\varsigma \qquad (2.14)$$

$$F(y) = \int_{-\infty}^{y} \Phi(\zeta) d\zeta, \quad \Phi(y) = \frac{1}{\varsigma\sqrt{2\pi}} e^{-\frac{(y-\xi)^2}{2\varsigma^2}} \qquad (2.15)$$

где ξ и ς – математическое ожидание и среднеквадратичное отклонение величины y при поступлении на входы нейрона обучающих примеров «Свой» или «Чужие».

Далее настройка порогов выполнялась в соответствии с предложенным алгоритмом, который иллюстрируется на рисунке 2.16.

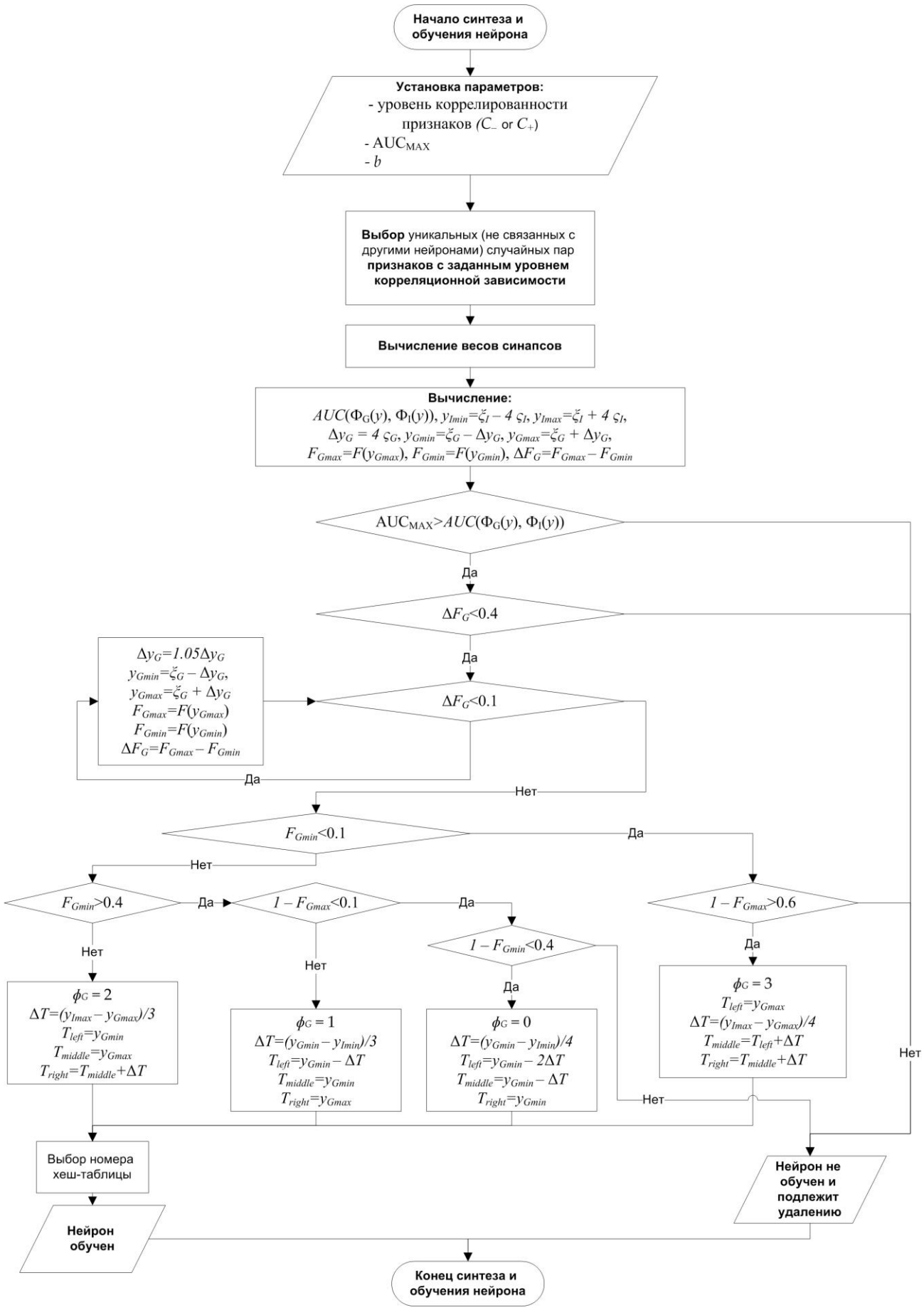


Рисунок 2.16 – Схема алгоритма синтеза и обучения корреляционного нейрона

Также введем коэффициент AUC_{MAX}, равный максимально допустимому показателю $AUC(\Phi_G(y), \Phi_I(y))$ для нейрона, чтобы исключить «слабые» нейроны, которые дают близкие отклики на образы «Свой» и «Чужие» (рисунок 2.16).

К значению функции активации применяется одна из таблиц перевода состояний {0, 1, 2, 3} в двухбитный код (далее хеш-таблицы, таблица 2.1). При обучении нейрона хеш-таблица выбирается случайно, но с учетом того, на какие два ключевых бита (далее b) настраивается нейрон. Например, если $\phi_G = 1$ и b = «10», то номер хеш-таблицы выбирается из множества {5, 6, 9, 10, 15, 16, 21, 22} (таблица 2.1).

Таблица 2.1. Варианты хеширующих преобразований отклика нейрона в двоичный код

Хеш-таблица	$\phi(y)$								
№	0	1	2	3	№	0	1	2	3
1	«11»	«00»	«01»	«10»	13	«01»	«00»	«11»	«10»
2	«11»	«00»	«10»	«01»	14	«01»	«00»	«10»	«11»
3	«11»	«01»	«00»	«10»	15	«01»	«10»	«00»	«11»
4	«11»	«01»	«10»	«00»	16	«01»	«10»	«11»	«00»
5	«11»	«10»	«00»	«01»	17	«01»	«11»	«10»	«00»
6	«11»	«10»	«01»	«00»	18	«01»	«11»	«00»	«10»
7	«00»	«01»	«11»	«10»	19	«10»	«00»	«01»	«11»
8	«00»	«01»	«10»	«11»	20	«10»	«00»	«11»	«01»
9	«00»	«10»	«01»	«11»	21	«10»	«01»	«11»	«00»
10	«00»	«10»	«11»	«01»	22	«10»	«01»	«00»	«11»
11	«00»	«11»	«10»	«01»	23	«10»	«11»	«00»	«01»
12	«00»	«11»	«01»	«10»	24	«10»	«11»	«01»	«00»

Таким образом, для обучения корреляционного нейрона Байеса-Минковского достаточно определить связанные мета-признаки, вычислить веса и пороги, а также задать хеш-таблицу (рисунок 2.16).

Даже если хакеру удалось определить пары признаков с близким уровнем корреляции (что затруднительно), это не даст оснований утверждать, что биометрический шаблон пользователя может быть скомпрометирован. Так как нормировка по δ_j (2.8) не приводит признаки к единой области значений для всех классов образов, а только корректирует масштаб, то выражение (2.11) зависит не только от корреляции между признаками, но и от значений признаков. Поэтому сгенерировать данные с определенной корреляционной матрицей недостаточно для создания успешного состязательного примера.

Возможны иные более сложные конструкции корреляционных нейронов с большим количеством уровней коррелированности и множеством вентилей, пропускающих данные с разной коррелированностью (например, вентили отрицательной и положительной корреляции), а также оперирующие произведением коррелированных признаков вместо разностей (гармонические корреляционные нейроны). В незащищенном режиме исполнения можно применять иные функции активации. Эти нейроны требуют проведения отдельных исследований и в настоящей работе не рассматриваются.

2.7 Множественные квантователи в активационных функциях корреляционных нейронов

Важнейшим свойством НПБК является следующее: чем больше квантователей имеет функция активации нейрона, тем выше хеширующие свойства корреляционного нейрона [97, 116]. Эти свойства справедливы и для корреляционных нейронов. Трехуровневое квантование соответствует функциям активации:

$$f(y)=\begin{cases}"01", if\ y<\mu_0\\"10", if\ \mu_0<y<\mu_1\\"11", if\ y>\mu_1\end{cases} \qquad f(y)=\begin{cases}"00", if\ y<\mu_0\\"10", if\ \mu_0<y<\mu_1\\"11", if\ y>\mu_1\end{cases}$$

$$f(y)=\begin{cases}"00", if\ y<\mu_0\\"01", if\ \mu_0<y<\mu_1\\"11", if\ y>\mu_1\end{cases} \qquad f(y)=\begin{cases}"00", if\ y<\mu_0\\"10", if\ \mu_0<y<\mu_1\\"01", if\ y>\mu_1\end{cases}$$

Существует всего 4 варианта ее реализации (к разным нейронам можно применить различные активационные функции в рамках одного слоя сети). На выходе этих функций будет два бинарных значения, при этом для каждой из них существует только 3 возможных двухбитных состояния. Четырехуровневое квантование определяется функцией активации с тремя порогами:

$$f(y)=\begin{cases}"00", if\ \ y<\mu_0\\ "10", if\ \ \mu_0<y<\mu_1\\ "01", if\ \ \mu_1<y<\mu_2\\ "11", if\ \ y>\mu_2\end{cases}$$

Чтобы повысить хеширующие свойства НПБК на базе корреляционных нейронов, желательно, чтобы состояния каждого нейрона были равновероятны (по аналогии с соответствующим требованием для двухуровневого линейного нейрона в ГОСТ Р 52633.5-2011): для трехуровневых нейронов – 0,333, для четырехуровневых – 0,25 и т.д. [62, 141, 155, 158].

Нейрон с множественными квантователями в определенном смысле лучше, чем множество нейронов с одним квантователем и двумя бинарными выходными состояниями, так как:

1. Решения всех нейронов в той или иной степени коррелированы. Даже, если на входах двух нейронов различные признаки, их решения все равно могут коррелировать, так как сами признаки могут быть коррелированными.
2. Чем больше нейронов, тем выше вычислительная нагрузка.

Таким образом, один «многоуровневый» нейрон работает как несколько нейронов с одним квантователем. Совершенствование моделей нейронов может идти по пути увеличения количества квантователей функции активации. Предложенная модель использует 4 уровня (3 квантователя), но их может быть больше.

2.8 Синтез и автоматическое обучение нейросетевых преобразователей биометрия-код на основе разностных корреляционных нейронов Байеса-Минковского

Предлагаемая модель НПБК представляет собой малую (неглубокую) нейронную сеть, состоящую из одного скрытого слоя корреляционных нейронов Байеса-Минковского (рисунок 2.17). НПБК работает с векторами признаков,

поэтому «сырые» образы обучающей выборки «Чужие» и «Свой» сначала должны быть обработаны блоком извлечения признаков. К блоку извлечения признаков предъявляется всего одно требование: каждый извлекаемый из образа признак должен подчиняться нормальному закону распределения или близкому к нему (функция плотности вероятности должна быть симметричной и одномодальной). Это требование несложно выполнить, если для извлечения признаков использовать вариационный автокодировщик.

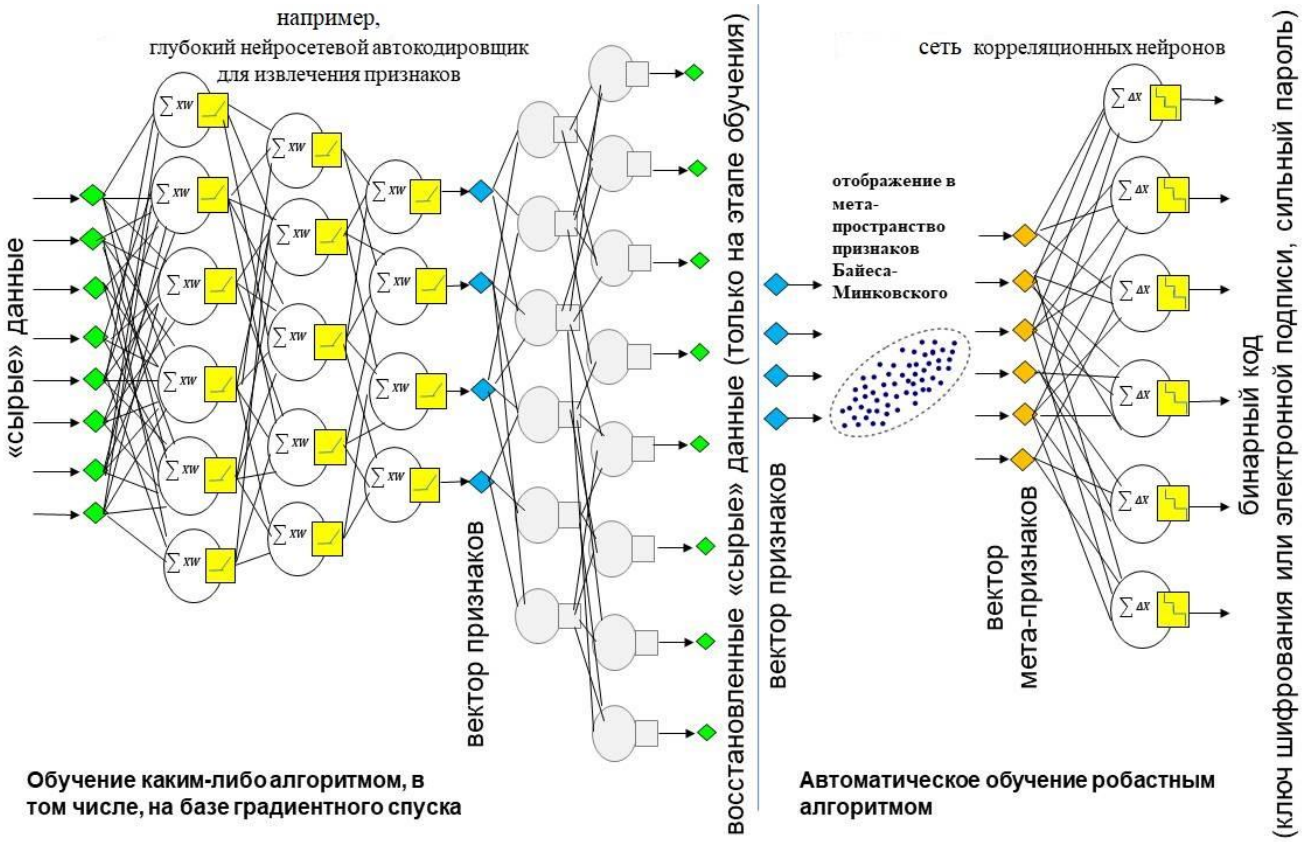


Рисунок 2.17 – Структурная схема связывания ключа и биометрического образа: слева экстрактор признаков, справа НПБК

Выходы кодировщика должны быть связаны с НПБК, однако предварительно извлекаемые признаки должны быть преобразованы в мета-признаки Байеса-Минковского с помощью специального отображения (рисунок 2.17). При регистрации нового пользователя в системе высоконадежной биометрической аутентификации для него создается отдельный НПБК, который обучается на примерах «Свой» и «Чужие» в доверенной среде (в более общем случае, для каждого класса образов создается отдельный НПБК, который

обучается независимо от остальных на выборке относительно небольшого размера). НПБК играет роль последнего слоя (вместо слоя SoftMax), но более интеллектуального, способного генерировать почти случайный выход в ответ на неоднозначный образ «Чужого», поступивший на вход кодировщика. После обучения НПБК может размещаться в открытом виде в недоверенной среде.

Пусть переход в пространство мета-признаков осуществляется с использованием отображения (2.8) при g=0,9. Нормирующие коэффициенты δ_j для перехода в пространство мета-признаков должны быть вычислены на основании выборки «Чужие» до построения и обучения НПБК.

Далее рассматривается случай, когда количество входов η для всех корреляционных нейронов должно быть равным. При синтезе НПБК для конкретного пользователя необходимо убедиться, что имеется достаточное количество пар признаков с уровнями взаимной корреляции $C_{j,t}<C_-$ и $C_{j,t}>C_+$. Для этого следует рассчитать корреляционную матрицу по данным выборки «Свой». Любая пара коррелированных признаков потенциально порождает один мета-признак. Пусть N_- и N_+ – количества нейронов, ориентированных на обработку уровней коррелированности данных $C_{j,t}<C_-$ и $C_{j,t}>C_+$, соответственно. Должно соблюдаться условие $N_- \approx N_+$ (допускается расхождение на 1-3 нейрона). Каждый нейрон должен обрабатывать уникальную комбинацию мета-признаков и генерировать на выходе 2 бита. Нужное количество нейронов определяется, исходя из требуемой длины ключа L. Для практических целей вполне достаточно длины L=1024 бит, в этом случае $N_-=N_+=L/2/2=256$. Тогда если η=4, то для синтеза НПБК потребуется 2048 пар признаков (по 1024=256·4 пар для каждого уровня коррелированности). При использовании автокодировщиков количество признаков можно сделать произвольным. К примеру, 256 признаков дает 32640 потенциальных пар, среди которых нужно выбрать 2048 (≈6,27%).

Предложенный алгоритм обучения НПБК в общем виде можно изложить как последовательность шагов:

1. Расчет корреляционной матрицы признаков.

2. Подсчет пар отрицательно коррелированных признаков ($C_{j,t}<C_-$). Если количество пар менее $\eta \cdot N_-$, то C_- увеличивается на 0,05 и данный шаг повторяется.
3. Синтез и обучение N_- нейронов для анализа отрицательно коррелированных данных в соответствии с алгоритмом на рисунке 2.16. Если количество нейронов, удовлетворяющих условиям алгоритма на рисунке 2.16, оказалось менее N_-, то C_- увеличивается на 0,05 и шаги 2-3 повторяются.
4. Подсчет пар положительно коррелированных признаков ($C_{j,t}>C_+$). Если количество пар менее $\eta \cdot N_+$, то C_+ уменьшается на 0,05 и данный шаг повторяется.
5. Синтез и обучение N_+ нейронов в соответствии с алгоритмом на рисунке 2.16 для анализа положительно коррелированных данных. Если количество нейронов, удовлетворяющих условиям алгоритма на рисунке 2.16, оказалось менее N_+, то C_+ уменьшается на 0,05 и шаги 4-5 повторяются.

По мере сужения интервала (C_-; C_+) уже созданные нейроны допустимо не удалять (новые нейроны могут быть добавлены к существующим). Алгоритм исполняется, пока не выполнится условие $N_-=N_+=L/4$ либо пока не будет нарушено условие $|C_{-+}| \geq 0{,}3$. Последнее означает, что невозможно связать биометрический образ пользователя с ключом длины L.

Таблицы весовых коэффициентов w_i и номера хеш-таблиц обученного НПБК представляют собой защищенный эталон пользователя. Отметим следующие преимущества НПБК:

– корреляционные нейроны не подвержены проблеме несбалансированности обучения (объем выборки «Чужие», как правило, много больше объема выборки «Свой»);
– процесс настройки корреляционной сети является робастным и переобучения не возникает;
– длина ключа, связанного с НПБК, потенциально гораздо выше, чем для fuzzy extractors и neuro-extractors;

– предложенная модель НПБК должна иметь гораздо более высокий уровень устойчивости к состязательным атакам, чем классическая глубокая сеть с функцией активации Softmax на выходе [236, 328], по крайней мере, в плане влияния на показатель FAR. Добавление шумов и других модификаций вряд ли повлияет на близость корреляционных связей образа «Чужой» к образу «Свой». Для синтеза состязательных примеров следует не только подобрать верные распределения значений признаков, но и правильно задать корреляционную матрицу между признаками

В настоящей работе рассмотрено два интервала коррелированности признаков ($1>C_{j,t}>C_{+}$ и $-1<C_{j,t}<C_{-}$) и 4 интервала квантования для функции активации (2.13). Увеличение количества интервалов должно усиливать хеширующие свойства НПБК.

Проведенные исследования показали, что если объединить классические нейроны с другими типами нейронов в гибридный слой нейронов, синтезировав таким образом гибридуную нейронную сеть, можно снизить показатели FRR и FAR и повысить энтропию ответов НПБК. Еще более интересным является то, что создание многослойных гибридных нейронных сетей, где в каждом слое будут использоваться различные типы нейронов, также позволяет снизить вероятность ошибочных решений. В Приложении А приведены результаты экспериментов в краткой форме по распознаванию динамических биометрических образов с помощью различного рода нейронов, их сетей, а также гибридного НПБК на основе различных типов нейронов. Также в Приложении А указаны параметры модели гибридного НПБК, описана ее архитектура и схема построения. Недостатком приведенной в Приложении А модели гибридного НПБК является незащищенность знаний ИИ от компрометации. Тем не менее, полученные результаты говорят о высокой эффективности комплексирования различных типов нейронов и перспективности этого направления совершенствования НПБК.

2.9 Применение нейросетевых преобразователей биометрия-код на безе корреляционных нейронов для идентификации образов

Для реализации процедуры идентификации (сравнение один ко многим) в защищенном режиме можно использовать несколько НПБК, каждый из которых обучается генерировать криптографический ключ (например, закрытый ключ электронной подписи) в ответ на образ определенного класса и случайный шум в ответ на образы других классов (в соответствии с изложенным выше методом). На этапе идентификации каждый НПБК генерирует последовательность бит. Верной может оказаться бинарный код только одного НПБК. Таким образом, каждая сессия по идентификации образа порождает несколько потенциальных ключей. Сверка ключей может выполняться по следующему принципу: с использованием открытого ключа следует выполнить шифрование определенной информации, далее с помощью закрытого ключа выполнить ее дешифрование. Представленный ключ можно считать аутентичным, если исходные и расшифрованные данные равны. Так можно проверить все полученные ключи. Поступающие от НПБК данные следует игнорировать в том случае, если ни один из представленных ключей не является верным либо если верны более одного ключа.

2.10 Анализ результатов. Выводы.

Классическая теория математической статистики утверждает: если признаки коррелированы, то они дублируют определенную информацию. Однако полученные результаты говорят об обратном: сильно коррелированные пары признаков содержат дополнительную информацию. Конечно, проверенная десятилетиями теория не может быть неверной. Полученные данные лишь уточняют ее применительно к задачам классификации образов и дополняют новыми фундаментальными положениями, касающимися влияния

корреляционной зависимости между признаками на процесс распознавания образов. Корреляционные связи между признаками искривляют пространство признаков. Характер искривления в относительно каждого класса образов в общем случае различен, так как для разных классов образов корреляционные матрицы признаков могут существенно отличаться. Такое искривление затрудняет построения разделяющих гиперплоскостей в пространстве признаков в процессе машинного обучения. Из коррелированных пар признаков можно извлечь независимые (слабо коррелированные) информативные мета-признаки, причем один мета-признак может содержать в 2-3 раза больше информации, чем содержится в паре исходных признаков, от которых он порожден ($I'_{j,t}>I_j+I_t$), что потенциально может значительно повысить надежность распознавания образов. Пространство мета-признаков Байеса-Минковского не искривлено, в отличие от исходного пространства коррелированных признаков.

Предложена модель разностных корреляционных нейронов, которые позволяют использовать информацию о корреляционных связях между признаками для классификации образов. Одна сеть корреляционных нейронов может использоваться для верификации образов, но совокупность (комитет сетей) может решать задачи идентификации. Сеть корреляционных нейронов обучается автоматически на выборке небольшого объема и может быть объединена с предварительно обученной глубокой нейронной сетью. Это позволяет получить преимущества как в плане упрощения процедуры обучения, так и в плане информационной безопасности. Сети корреляционных нейронов потенциально являются более устойчивыми к таким деструктивным воздействием, как состязательные атаки (по сравнению, например, с глубокой нейронной сетью с функцией активации SoftMax в последнем слое). При идентификации образа, не относящегося ни к одному из известных классов, сети корреляционных нейронов должны генерировать почти случайный бинарный код, который следует игнорировать при принятии решений. В данной работе не утверждается превосходства сетей корреляционных нейронов над многослойными сетями, а лишь указывается на перспективность корреляционных нейронов в некоторых

отношениях (автоматическое обучение/дообучение, обеспечение безопасности процесса принятия решений).

Предложена модель НПБК на базе корреляционных нейронов для задач высоконадежной биометрической аутентификации и других приложений классификации образов в защищенном режиме. НПБК позволяет связать криптографический ключ или пароль с биометрическим образом пользователя и хранить обе эти составляющие безопасно без компрометации. Предложенная модель потенциально превосходит известные ранее модели (fuzzy extractors, neuro-extractors, ГОСТ Р 52633.5-2011, гибридный НПБК) по длине ключа.

Многие свойства сетей корреляционных нейронов еще предстоит исследовать. Существует много потенциальных конструкций нейронов, позволяющих анализировать корреляционные связи, не компрометируя биометрические эталоны. Перспективным направлением для будущих исследований видится синтез ансамблей классификаторов (нейронов), каждый из которых обрабатывает отдельную совокупность признаков в зависимости от уровня их коррелированности. Можно сказать, что нейроны Байеса-Минковского являются «антагонистами» по отношению к нейронам Минковского, так как обладают противоположными свойствами (совершает меньше ошибок, если признаки коррелированы, и больше ошибок, если признаки независимы). Поэтому их можно использовать совместно, анализируя сильно коррелированные признаки нейронами Байеса-Минковского, а слабо коррелированные – нейронами Минковского. Также планируется усовершенствовать алгоритм обучения корреляционных сетей.

Помимо задач биометрической аутентификации сети корреляционных нейронов могут применяться в других задачах классификации образов (в том числе, в сочетании с классическими глубокими сетями), особенно в тех случаях, когда объем обучающей выборки ограничен. Перспективным видится применение сетей корреляционных нейронов для синтеза искусственного интеллекта, устойчивого к состязательным атакам, а также атакам, целью которых является извлечение знаний ИИ и различные манипуляции с моделями ИИ.

3 Адаптивные нейро-иммунные модели искусственного интеллекта, устойчивые к дрейфу биометрических данных

Одной из ключевых проблем машинного обучения является проблема концептуального дрейфа. Если дрейф данных (сбои датчиков, изменение единиц измерения, появление данных, неучтенных при обучении и т.д.) часто устраняется относительно легко (следует продумать все возможные изменения, которые могут быть спрогнозированы при обучении модели), то концептуальных дрейф устранить затруднительно, так как нельзя заранее знать, как в будущем изменится прогнозируемое явление.

В биометрии дрейф модели можно условно разделить на:

- кратковременный (изменения в зависимости от эмоционального или психофизиологического состояния) [23, 144, 145];
- долговременный (медленные и, как-правило, необратимые изменения биометрического образа пользователя со временем, например, изменение почерка и т.д.) [279].

Не смотря на различные причины дрейфа, оба типа изменений крайне сложно спрогнозировать, что будет показано в следующем параграфе. Если для статических биометрических образов (отпечаток пальца, радужка, сетчатка, лицо) дрейф появляется только при физических нарушениях, таких как травма, порезы и т.д., то для динамических биометрических образов этих изменений почти невозможно избежать (например, клавиатурный почерк меняется в течения дня).

Можно сформулировать несколько общих приемов и подходов, предназначенных для снижения негативного влияния дрейфа. Например, периодическое обновление модели дает положительный эффект, если есть данные для переобучения. При этом может применяться взвешивание данных – присвоение большего веса наиболее актуальным обучающим примерам и меньшего веса данным, полученным давно. Для своевременного обнаружения дрейфа используются метрики, вычисляющие статистические характеристики

данных с учетом ретроспективы [88, 170, 226, 322], а также ансамблевые методы классификации [26, 166]. Однако наиболее эффективным подходом является онлайн обучение модели. Только этот подход при правильной реализации позволяет устранить или в значительной степени снизить влияние концептуального дрейфа. Для некоторых приложений методы онлайн обучения исследованы достаточно глубоко [48], тем не менее, применительно к биометрическими системам этот вопрос остается фактически нерешенным, так как существующие способы контроля изменений биометрических образов сильно зависимы от типа биометрических параметров.

В настоящей главе представлены:

– результаты экспериментальных исследований, демонстрирующие не только степень изменчивости динамических биометрических признаков, но и иллюстрирующие негативное влияние этих изменений на результаты биометрической аутентификации;
– разработанная адаптивная нейро-иммунная модель ИИ на основе корреляционных нейронов и биоинспирированного иммунного подхода к машинному обучению. Модели, в которых совместно применяются элементы аппарата искусственных нейронных сетей и искусственных иммунных систем или сетей (ИИС), принято называть нейро-иммунными (neural immune networks, neural immune systems) [286, 352, 354]. Синтез и обучение нейро-иммунных моделей выполняется с использованием принципов ИИС, однако процесс классификации образов при помощи нейро-иммунной модели после ее обучения схож работе ИНС. Адаптивная модель построена с учетом существующих представлений о принципах работы иммунной системы, которая является частным случаем естественной интеллектуальной системы;
– предложенные алгоритмы пакетного обучения адаптивной модели ИИ на малых выборках биометрических данных и онлайн-обучения модели ИИ в процессе функционирования;

– оценка эффективности предложенных адаптивной модели ИИ и алгоритмов ее обучения в задаче распознавания пользователей по клавиатурному почерку.

3.1 Изменчивость биометрических образов со временем и в зависимости от психофизиологического состояния

В настоящем параграфе приведем примеры изменений следующих типов динамических биометрических образов в зависимости от психофизиологического состояния:

- клавиатурный почерк;
- голос;
- рукописный почерк при воспроизведении подписи.

Прежде всего, следует определиться с понятием психофизиологическое состояние (ПФС) и сформировать наборы биометрических данных субъектов, находящихся в различных ПФС (существующие открытые наборы данных не содержат образцов, полученных в различных ПФС).

Имеется несколько трактовок этого понятия «состояние» человека [49], в данной работе интерес представляют следующие:

1. Функциональное состояние (ФС), которое характеризует эффективность деятельности или поведения человека и возможность выполнить конкретную работу [49]. Диагностика ФС выполняется на основании результатов измерения психофизиологической информации, а также информации о качестве деятельности субъекта. Данный термин часто употребляется в контексте рассмотрения эрганических систем и когда речь идет о производительности труда. Известны методы определения ФС человека на основе анализа по зрачково-двигательной реакции глаза [4, 5], клавиатурному почерку [145], вариабельности сердечного ритма [77], тепловому изображению лица [122].

2. Психофизиологическое состояние (ПФС) – совокупность свойств человека, отражающих биологические аспекты проявления адаптации к изменяющимся условиям окружающей среды и оцениваемых на основании измерения психофизиологической информации [49].

При определении сущности понятия «состояние» человека разные авторы опираются на разные уровни функционирования человека (физиологический, психологический, психофизиологический). В работе [79] приводится несколько точек зрения относительно смыслового различия понятий ФС и ПФС. Автор работы [79] не считает данные понятия тождественными, однако подчеркивает, что они взаимосвязаны, и придерживается термина ПФС. По мнению автора [79] «когда речь идет о функциональных состояниях, имеют в виду уровень функционирования человека в целом или его отдельных функциональных систем (сенсорной, интеллектуальной, моторной), а когда говорят о психических состояниях, то речь идет о качественной специфике реагирования человека на ту или иную ситуацию (без учета уровня функционирования)». Автор настоящей работы разделяет данную точку зрения и считает, что последнее определение (ПФС) является предпочтительным с точки зрения решаемых в рамках работы задач и в большей степени подчеркивает психофизиологическую природу распознаваемых состояний. Стоит отметить, что в отличии от психического состояния, которое является более долговременным и также отражается на биометрических данных и поведении пользователя [15, 16], ПФС отражает изменения не только психических, но и физиологических параметров субъекта.

Выбор критериев, определяющих состояние человека, представляет собой нетривиальную задачу. В настоящем исследовании решено адаптировать (упростить) модель 8 состояний, предложенную в работе [99]. Для изменения ПФС испытуемых можно использовать различные лекарственные и иные препараты, воздействие которых на нервную систему хорошо изучено. Для легкого возбуждения нервной системы могут быть использованы адаптогены растительного происхождения: лимонник, элеутерококк, женьшень, золотой корень, родиола. Состояние возбуждения и легкой эйфории может быть вызвано

также употреблением кофеина. Чтобы ввести испытуемого в расслабленное или сонное состояние, можно воспользоваться естественными растительными средствами (веществами) седативного действия, к которым относится пустырник, мята. Также успокаивающее действие на нервную систему оказывают валериана (в гранулах). Указанные вещества (средства) не вызывают привыкания и существенных побочных эффектов и могут быть безопасно использованы в рамках эксперимента. Важным является то, повлияло ли некоторое воздействие на оператора в целом, о чем можно судить по изменению частоты сердечных сокращений (ЧСС). В серии экспериментов использовались следующие ПФС:

1. Адекватное (или нормальное) состояние, при котором субъект не подвергался каким-либо воздействиям. Эксперимент проводился в начале рабочего дня. Обязательным условием было выполнение требования полноценного отдыха в предшествующие сутки.
2. Возбуждение – характерно для человека, сконцентрированного на решении ответственной задачи. Для легкого возбуждения нервной системы участники принимали кофеин либо адаптогены растительного происхождения: лимонник, элеутерококк, женьшень, золотой корень, радиола. ЧСС в среднем возрастала на 10%. Для сильно возбужденных людей также характерно учащенное дыхание и обильное потоотделение [165].
3. Усталость (после физической нагрузки) – возникает в период после выполнения ответственного задания, характеризуется угнетенностью, утомлением, учащением ЧСС на 10-30%. Для получения нужного эффекта испытуемые подвергались интенсивной физической нагрузке (упражнения, бег), минимальный объем которой определялся методом Мартине (20 приседаний за 30 секунд) и далее варьировался в зависимости от пола и возраста. В ряде работ под усталостью понимается состояние, близкое к сонному, при котором наблюдаются такие признаки, как зевание и закрытие глаз [83]. В настоящей работе усталостью названо состояние, возникающее непосредственно после нагрузки, предшествующее сонному состоянию, в то

время как сонное состояние можно наблюдать в конце рабочего дня, или если человек не выспался.

4. Глубокое расслабление (сонное) состояние, характеризующееся легкой сонливостью, низкой продуктивностью. Для имитации данного состояния испытуемые принимали естественные растительные средства седативного действия (пустырник, мяту, валериану). ЧСС снижалась на 3-5% по сравнению с нормальным состоянием.
5. Опьянение. Испытуемый принимал алкоголь, дозировка рассчитывалась по формуле Видмарка:

$$c=A/m\cdot V,$$

где c – концентрация алкоголя в крови в ‰, A – масса выпитого напитка в граммах, m – масса тела в килограммах, V – коэффициент Видмарка (0,7 – для мужчин, 0,6 – для женщин). Опираясь на Федеральные правила полетов США (91.17: Алкоголь и пилотирование) и методические указания Минздрава (от 03.07.1974 «О судебно-медицинской диагностике смертельных отравлений этиловым алкоголем и допускаемых при этом ошибках»), было выделено 3 условных стадии, каждая из которых определяет конкретное ПФС субъекта:

   5.1. Первая стадия опьянения, при котором содержание алкоголя в крови составляет от 0,2 до 0,3‰. Согласно методическим указаниям Минздрава в этом состоянии отсутствует существенное влияние алкоголя на организм. Однако могут наблюдаться скрытые проявления нарушений, которые могут быть обнаружены специальными тестами.

   5.2. Вторая стадия опьянения, при котором содержание алкоголя в крови составляет от 0,3 до 0,5‰. Согласно методическим указаниям Минздрава данная концентрация оказывает незначительное влияние на организм. Алкоголь в соответствующем количестве может влиять на мыслительный процесс, решения, координацию и концентрацию, а также может увеличивать время реакции. В данном состоянии наблюдается легкая эйфория, расслабление, ощущение радости, снижение предупреждений, понижение сдержанности.

5.3. Третья стадия опьянения (также называемая «легкое опьянение»), при котором содержание алкоголя в крови составляет от 0,5 до 1 ‰. Согласно методическим указаниям Минздрава концентрация алкоголя в пределах 0,5-1,5 ‰ превышает допустимый уровень концентрации алкоголя в крови для водителей транспортных средств в России и соответствует легкому опьянению. Количество алкоголя от 0,5 ‰ приводит к статистически значимым изменениям вариабельности сердечного ритма [213, 288]. Данный уровень опьянения выбран, исходя из критериев [119]. Согласно принятой схеме, при меньшей концентрации отсутствует влияние алкоголя на организм. На данной стадии опьянения наблюдаются притупление ощущений, экстравертность, нарушаются способность к рассуждениям, глубина восприятия, периферическое зрение, реакция зрачка на свет [4, 5]. Сбор биометрических образцов в этом состоянии производился в отдельный день.

Назовем любое состояние, отличное от нормального измененным.

Перед началом эксперимента проводилась оценка неврологического статуса испытуемых: состояния высших мозговых функций (настроение, уровень внимания, быстрота и адекватность ответов на вопросы), двигательных функций и координации (точность произвольных движений, способность поддерживать равновесие), чувствительность (наличие болевого синдрома). Неврологический статус всех испытуемых до начала эксперимента оценивался как нормальный. В процессе ввода биометрических данных за испытуемыми осуществлялся холтер-мониторинг ЧСС мышцы (использовался регистратор «Кардиотехника-04»). При изменении ПФС частота сердечных сокращений менялась минимум на 10%.

Для каждого вида образов осуществлялся отдельный эксперимент по сбору данных. Полученные файлы в дальнейшем использовались для вычисления биометрических признаков, характеризующих субъектов и/или их ПФС. Для разных экспериментов могло привлекаться различное количество испытуемых, а также могли использоваться отличающиеся наборы состояний. Тем не менее, условия, которые должен был выполнить субъект для погружения в определенное

ПФС, были идентичны для всех экспериментов. Контроль испытуемых и их состояний осуществлялся одинаково, независимо от типа биометрических данных.

Проведен эксперимент по вводу образцов клавиатурного почерка с привлечением 80 испытуемых. Каждый испытуемый при помощи стандартной клавиатуры вводил 3 парольные фразы («система защиты должна постоянно совершенствоваться», «прошу разрешить доступ к информации», «авторизация пользователя компьютерной системы») не менее чем по 50 раз каждую в следующих ПФС: нормальное, возбуждение, усталость, глубокое расслабление, легкое опьянение. В качестве признаков использовались временные промежутки между нажатием клавиш и времена удержания клавиш.

Исследования показали, что скорость ввода парольных фраз в измененных состояниях оператора в среднем снижается от 3% до 16% в зависимости от состояния (кроме, состояния усталости, рисунок 3.1). Также во всех измененных состояниях снижается стабильность времени набора парольных фраз на клавиатуре (от 37% до 203%, рисунок 3.1). Особенно это заметно для состояний глубокого расслабления (171%) и возбуждения (203%).

В измененных состояниях субъекта существенно снижается стабильность его клавиатурного почерка в целом при наборе парольных фраз. Об этом можно судить из рисунка 3.2. Площади пересечения эмпирических плотностей вероятностей коэффициентов корреляции между реализациями парольных фраз (векторами значений признаков), полученных от испытуемого в нормальном состоянии, с аналогичными плотностями, построенными по данным измененных состояний, характеризуют увеличение вероятности возникновения сбоев – некорректного воспроизведения динамики ввода парольной фразы на клавиатуре вследствие изменения функционального состояния оператора [50]. Наивысшее число сбоев наблюдается в состоянии легкого опьянения и глубокого расслабления (рис 3.2). Сбои (нехарактерные для субъекта значения признаков) наблюдаются, когда корреляция между реализациями парольной фразы одного и того же субъекта становится очень слабой (близкой к нулю) или отрицательной.

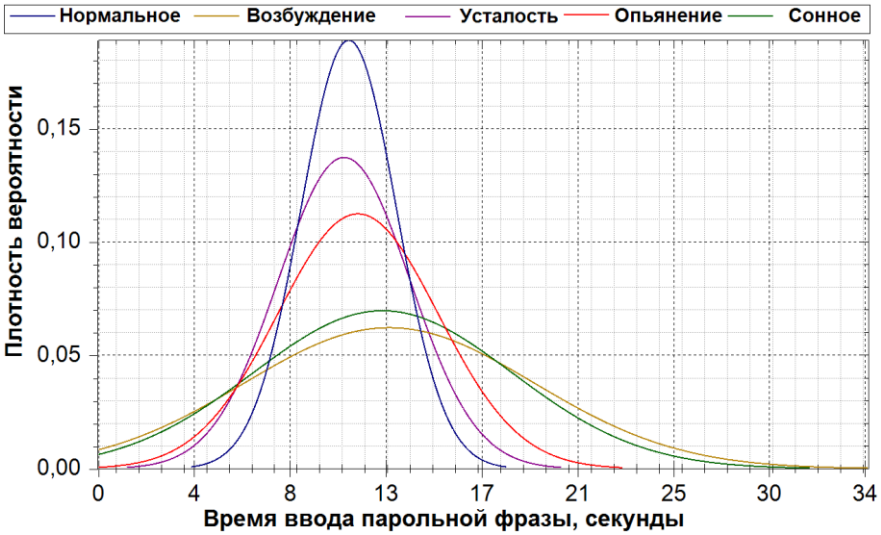


Рисунок 3.1 – Распределение времени ввода парольной фразы «система защиты должна постоянно совершенствоваться» 80 испытуемыми

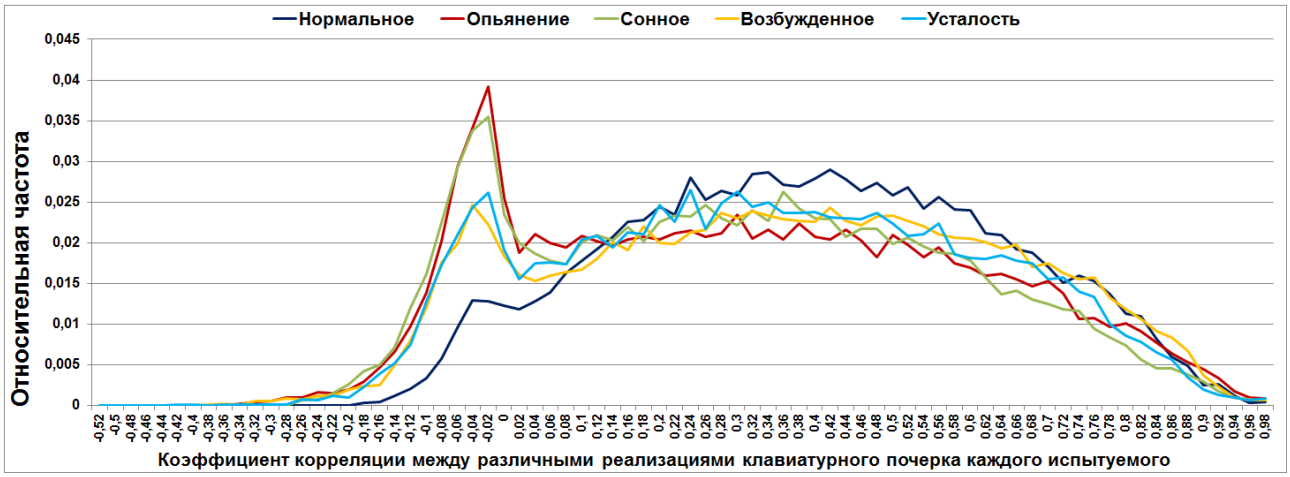


Рисунок 3.2 – Распределение парных коэффициентов корреляции между различными реализациями парольной фразы «система защиты должна постоянно совершенствоваться», введенных одним и тем же испытуемым (по данным 80 испытуемых)

Из рисунка 3.3 видно, что времена удержания клавиш при смене ФС оператора изменяются незначительно, что говорит о более высокой стабильности данной группы признаков, проявляющейся на подсознательном уровне субъекта. Паузы между нажатием клавиш гораздо менее стабильны. Сильней всего на признаки клавиатурного почерка влияет принятие алкоголя и успокоительного.

Характер изменения значений отдельных признаков в зависимости от состояния оператора можно видеть на рисунке 3.4. Времена удержания и паузы между нажатием определенных клавиш заметно изменились только в некоторых

ФС испытуемых. Определенной зависимости от удаленности или расположения клавиш выявлено не было. Прослеживается высокая корреляция между динамикой изменения средних значений признаков (Мизм/Мнорм) в зависимости от состояния оператора и динамикой изменения стандартных отклонений значений признаков (Sизм/Sнорм) в зависимости от ФС. Можно сделать вывод: при смене состояния оператора величина изменения математического ожидания значений признака пропорциональна величине изменения среднеквадратичного отклонения значений этого же признака. Этот вывод справедлив для признаков, изменение значений которых превышает 6% при смене ФС оператора.

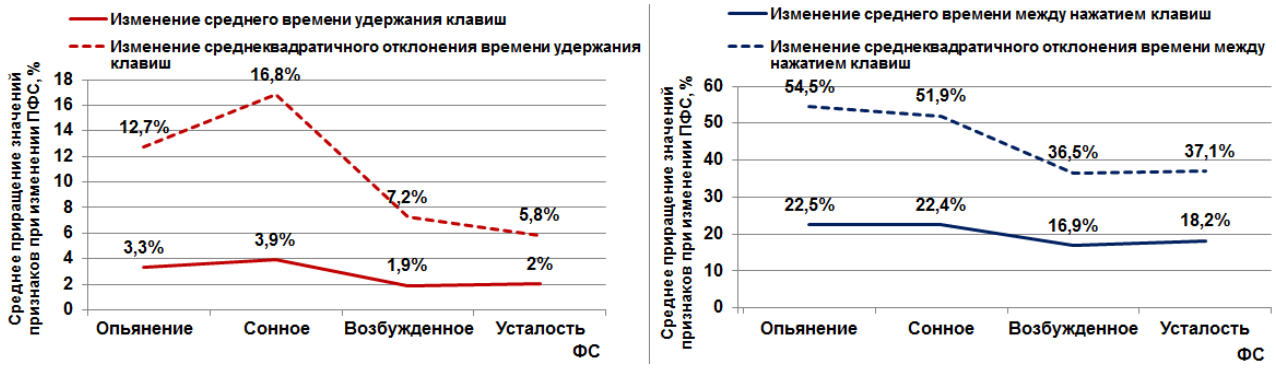


Рисунок 3.3 – Обобщенные показатели изменения значений признаков в зависимости от ФС оператора (по данным 80 испытуемых)

Были вычислены коэффициенты корреляции между векторами отношений Мизм/Мнорм и Sизм/Sнорми, в результате были выявлены наиболее близкие ФС испытуемых (таблица 3.1). Установлено, что очень схожие по характеру изменения признаков происходят при приеме успокоительного и кофеина. Также схожими являются состояния опьянения и усталости после физической нагрузки.

Таблица 3.1. Близость ФС с точки зрения влияния на клавиатурный почерк.

Средний коэффициент корреляции между обобщенными данными Мизм/Мнорм (внизу)				
ФС	Сонное	Усталость	Возбуждение	Опьянение
Сонное	1	0,061812007	0,995015638	0,013507268
Усталость	0,03997103	1	0,014045612	0,779658881
Возбуждение	0,996068037	0,004203967	1	-0,037508547
Опьянение	-0,003695652	0,814448644	-0,042318582	1
Средний коэффициент корреляции между обобщенными данными Sизм/Sнорм (вверху)				

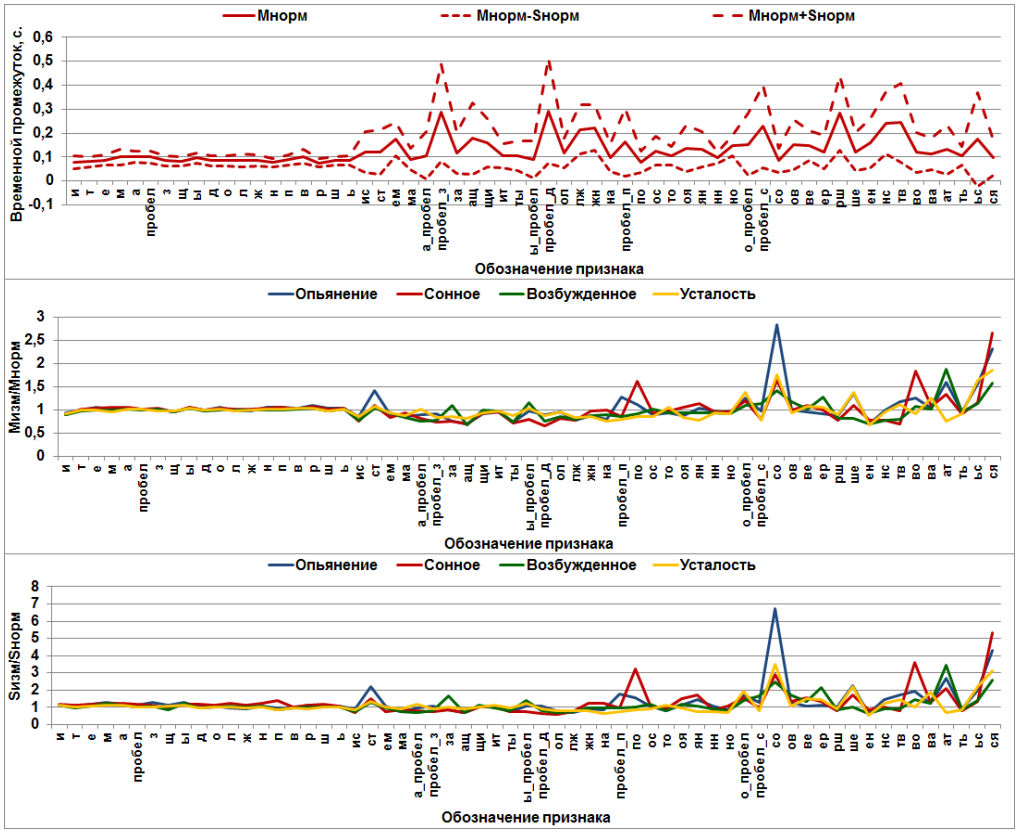


Рисунок 3.4 – Изменение значений признаков в парольной фразе «система защиты должна постоянно совершенствоваться» при смене состояния оператора (по данным 80 испытуемых)

Данные рисунка 3.4 носят обобщенный характер (вычислены по образцам всех испытуемых), у различных субъектов изменения признаков в зависимости от ФС могут быть индивидуальны либо отдаленно повторять динамику рисунка 3.4. Корреляция между функциями коэффициентов изменения математических ожиданий признаков (Мизм/Мнорм), относящихся к разным испытуемым в основном слабая и редко достигает 0,6-0,7, аналогичным образом коррелируют функции коэффициентов изменения стандартных отклонений признаков (Sизм/Snорм). Вывод: по выборке образцов клавиатурного почерка, введенных оператором в нормальном ФС, вычислить параметры распределения признаков, соответствующих измененному ФС оператора, затруднительно, по крайней мере, для произвольной фразы-пароля.

Из таблицы 3.1 можно видеть, что принятие алкоголя и интенсивная физическая нагрузка приводят к аналогичным изменениям клавиатурного почерка операторов. Эти изменения отличаются от тех, что наблюдаются после приема

успокоительного и кофеина. Успокоительное и кофеин также оказывают схожее действие на параметры клавиатурного почерка.

Более подробный анализ клавиатурного почерка дан в работах [3, 13, 54, 72, 103, 120, 143, 145, 149, 2910, 291, 335] (в том числе, с учетом дополнительных признаков, регистрируемых специальными датчиками, монтированными в клавиатуру, – силы нажатия на клавиши, вибрации корпуса клавиатуры и динамики движения рук над клавиатурой). Исследования показывают [145], что указанные изменения вызывают повышение вероятности ошибок идентификации и аутентификации пользователей по клавиатурному почерку, следовательно, происходит дрейф концепций. Этот дрейф невозможно заранее скорректировать, так как изменения признаков клавиатурного почерка не поддаются достаточно точному прогнозированию, что было показано в настоящем параграфе.

Для участия в натурном эксперименте по сбору данных голоса было привлечено 86 испытуемых в возрасте от 18 до 35 лет, мужчин и женщин (в равном соотношении), без выраженных заболеваний или неврологических нарушений. Эксперимент проводился в начале рабочего дня после полноценного отдыха. Испытуемые последовательно «вводились» в разные ПФС. В каждом ПФС испытуемые не менее 60 раз произносили 8 голосовых паролей, состоящих из 2-х слов («идентифицируйте меня», «разрешите доступ» и другие). Запись выполнялась с помощью микрофона Pioneer V-237. Аудиозаписи голосов дискретизированы с размером аудиообразца – 16 бит и частотой дискретизации 8000 Гц. Второй параметр задавался исходя из диапазона частот занимаемых речевым сигналом (до 4000 Гц), в соответствии с теоремой Котельникова кодирование без потерь для непрерывного сигнала из диапазона до определенной частоты возможно при его дискретизации с удвоенной частотой. Использовались следующие ПФС: нормальное, сонное, опьянение (3 стадии).

Метод извлечения признаков строился на интегрировании амплитудного спектра речевого сигнала в окрестности экстремумов. Предварительно спектр разбивается на некоторое число отрезков, равное количеству признаков. Данный подход взят из работ [85, 148]. В первой из них [85] вычислялось по 60 признаков

из парольной фразы целиком, во второй [148] – из каждой гласной фонемы. Однако в работе [148] не учитывались случаи ошибочного сегментирования речевого сигнала (ложное выделение фонемы, ложной пропуск фонемы). Количество ошибок при использовании методов разбиения, основанных на корреляционном анализе речевых сигналов из [148], в большинстве случаев оказалось значительным. Поэтому решено не делить парольную фразу на фрагменты, а обрабатывать сигнал целиком, получая из него вектор $\bar{a}$ значений некоторого количества признаков.

Установлено, что информативность голосовых признаков меняется в зависимости от ПФС (рисунок 3.5). Изменения информативности голосовых признаков, вызванные сменой ПФС диктора, носят волнообразный характер.

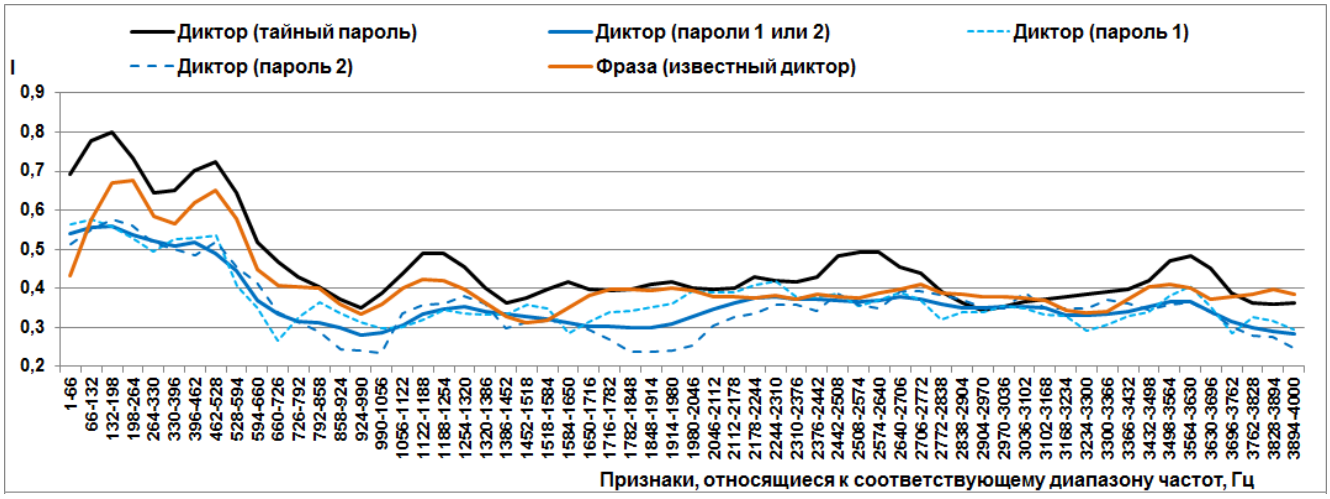


Рисунок 3.5 – Информативность голосовых признаков

По результатам исследований амплитудно-частотные характеристики голосовых паролей оказались чувствительны к изменению ПФС диктора, по крайней мере, в классе таких воздействий, как прием алкоголя и естественных растительных средств седативного действия (пустырник, мята, валериана). Характер изменений отличается для различных испытуемых (рисунок 3.6), что невозможно предсказать заранее. В работе [23, 36, 37, 85, 147, 148] дается более подробный анализ голосовых образов, а также показано, что вероятность ошибок идентификации и аутентификации диктора по голосу возрастает, если ПФС диктора изменяется.

Для проведения эксперимента по оценке влияния факторов утомления и возбуждения субъектов на результаты их идентификации по подписи сформирован набор данных рукописных образов 100 пользователей. В качестве контрольных слов использовались: «Безопасность», «Авторизация», «Идентификация», «Экранирование». При формировании базы паролей были привлечены 10 человек, каждый из которых по 50 раз написал каждое из указанных контрольных слов на графическом планшете Wacom в трех ПФС: нормальное, усталость, возбуждение. В процессе ввода регистрировались следующие функции, зависящие от времени:

– функция изменения координаты *x* при письме, *x_coord*(*t*);
– функция изменения координаты *y* при письме, *y_coord* (*t*);
– функция давления кончика пера на поверхность планшета при письме, *pressure*(*t*).

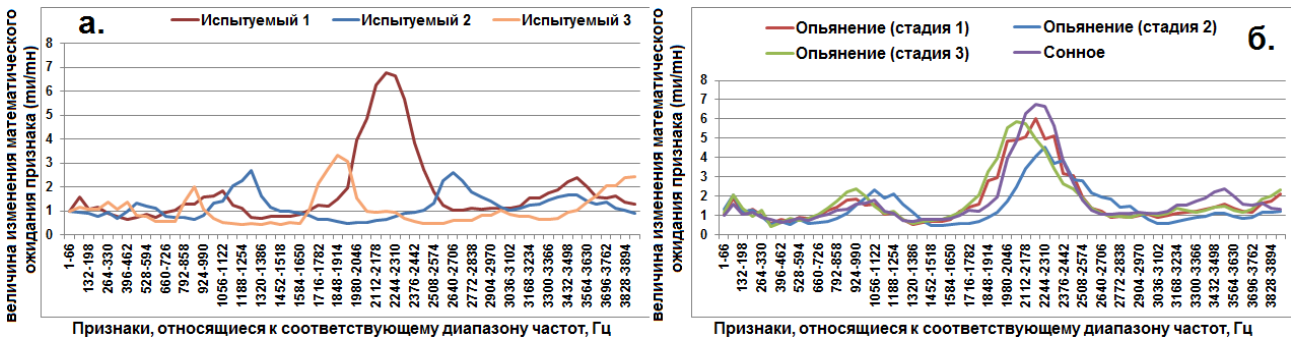


Рисунок 3.6 – Пример изменения средних значений признаков:
а. для 3-х дикторов в сонном состоянии,
б. для одного диктора в зависимости от состояния

Далее из этих функций извлекались признаки, получение которых из рукописного образа подробно описано в работах [84, 171, 291]:

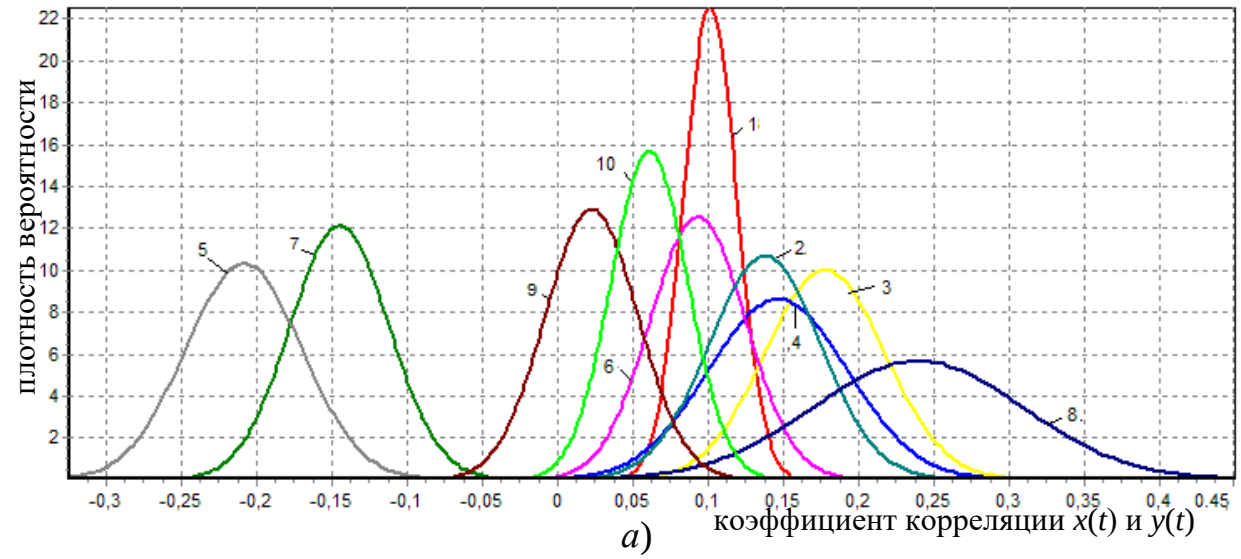
– нормированные по энергии амплитуды 16 самых низкочастотных гармоник функции давления *pressure*(*t*).
– нормированные по энергии амплитуды 16 самых низкочастотных гармоник функции скорости пера $v_{xy}(t)$, являющейся производной от функций координат.
– коэффициенты корреляции между функциями *x_coord* (*t*), *y_coord* (*t*), *pressure*(*t*) и их производными.

– расстояния между точками подписи в трехмерном пространстве (точки выбираются равномерно с определенным шагом, далее находятся расстояния между всеми парами этих точек, третье измерение – давление пара на планшет).
– характеристики изображения подписи: отношение длины подписи к ее ширине, центр подписи, углы наклона подписи и между центрами половин подписи.
– коэффициенты вейвлет-преобразований Добеши D6 функций $v_{xy}(t)$ и $pressure(t)$.

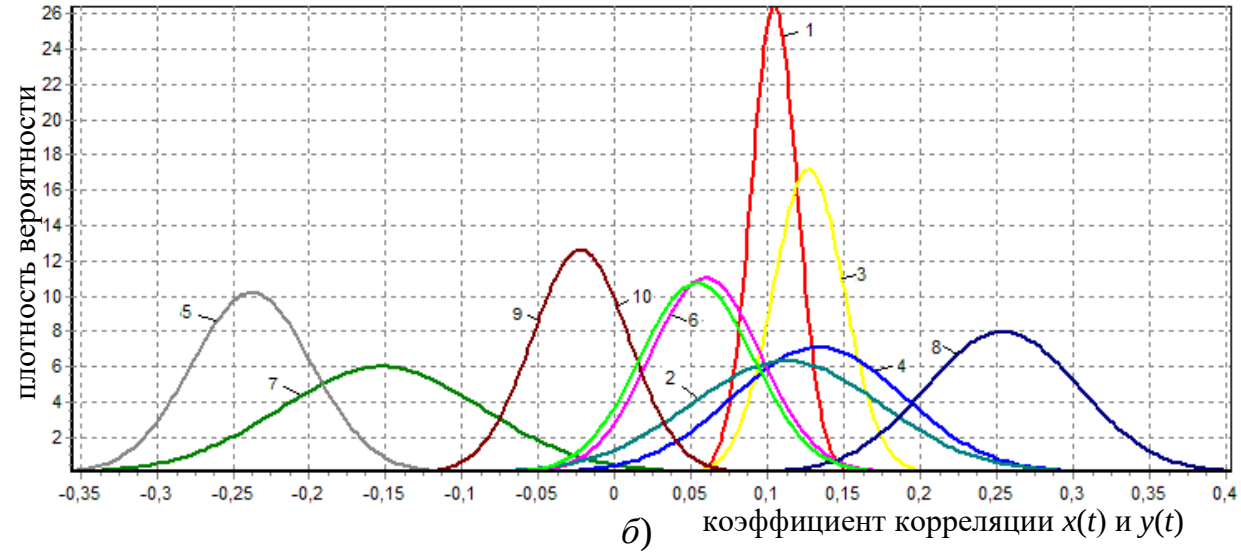
Анализ введенных образцов показал, что в измененном состоянии среднеквадратичное отклонение значений многих признаков возрастает, т.е. стабильность воспроизведения некоторых особенностей подписи снижается, однако это не является общим правилом для всех признаков. Математическое ожидание низкочастотных амплитуд функций $pressure(t)$ и $v_{xy}(t)$ в измененном состоянии чаще всего возрастает, т.е. доля низкочастотных колебаний руки при вводе автографа у большинства испытуемых увеличивается. Примеры графиков функций плотностей вероятности можно видеть на рисунке 3.7.

В работе [24] представлен анализ изменения вероятности ошибок идентификации подписей как искусственным интеллектом (на основе «наивного» классификатора Байеса), так и естественным интеллектом (при распознавании подписей человеком). Установлено, что на идентификационные решения, сделанные ИИ, любые изменения в психофизиологическом состоянии субъекта влияют отрицательно (вероятность ошибок возрастает в 3,3-3,7 раз). Искусственный интеллект существенно превосходит естественный при идентификации рукописных образов.

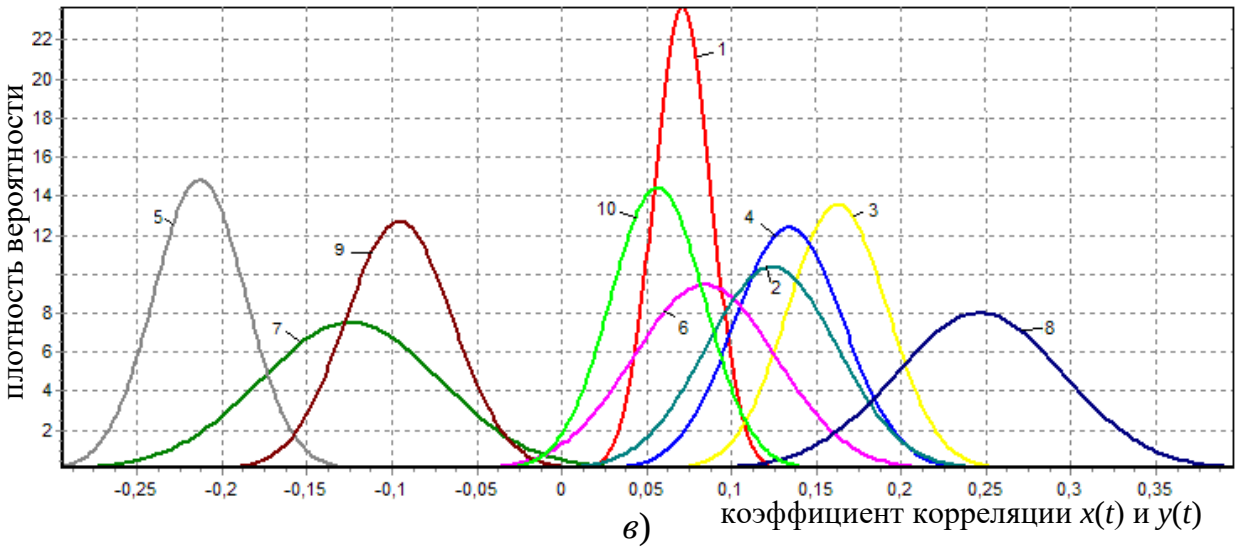
Из представленных данных можно видеть, что признаки подписи изменяются спонтанно в зависимости от ПФС. Более подробно рукописные образы и их зависимость от ПФС исследованы в работах [24, 74, 84, 86, 93, 100, 107, 113, 144, 146, 171, 248, 280, 308]. В частности показано, что вероятности ошибок верификации личности субъектов по подписи значительно возрастают, если на этапах обучения и тестирования их ПФС не были идентичными [144]. Эти исследования говорят о высоком уровне изменчивости рукописных образов в зависимости от ПФС, и как следствии появлении дрейфа моделей.



а)



б)



в)

Рисунок 3.7 – Графики функций плотностей вероятности коэффициентов корреляции функций $x(t)$ и $y(t)$ для 10 испытуемых (цифры на графике от 1 до 10) в различных психофизиологических состояниях:
а) – покоя, *б*) - возбуждения и *в*) расслабления

В работе [246] предложен метод идентификации ПФС субъекта на основе анализа таких характеристик, как клавиатурный почерк, голос, особенности работы с манипулятором «мышь». В работах [86, 302] для аналогичной цели использовались признаки рукописного почерка и голоса, а в [76] предложен метод идентификации ПФС по рукописной подписи, в [77] – по вариабельности сердечного ритма, в [122, 327] – по изображениям и термограммам лица. ПФС также влияет на другие типы биомедицинских данных [125], например, эти изменения отражаются на электроэнцефалограмме (ЭЭГ) [2, 78]. В работе [110, 123, 104] оценивалось влияние ПФС на вероятность верной идентификации личности по ЭЭГ, а также точность идентификации ПФС по ЭЭГ.

Таким образом, изменение ПФС оказывает почти непрогнозируемое изменение динамических биометрических образов. Решить эту проблему возможно с помощью алгоритмов онлайн-обучения.

3.2 Краткий обзор подходов к построению адаптивных моделей искусственного интеллекта

Под адаптивными (развивающимися, гибкими) моделями искусственного интеллекта подразумеваются модели, способные к онлайн-обучению.

В общем случае для нейросетевых архитектур классификаторов можно выделить следующие основные направления, которые позволяют создавать адаптивные модели: эволюционный подход (эволюционные нейронные сети), использование методов глубокого обучения с подкреплением и иммунный подход (искусственный иммунные системы и сети).

В основе эволюции нейронных сетей лежит процесс размножения и улучшения желаемых результатов путем выбора определенных параметров удовлетворяющих цели эволюционного процесса. Наиболее успешные топологии связей и весов нейронов порождают еще более эффективные модели [205]. Суть обучения эволюционных сетей состоит в следующем: на первом шаге, исходя из

данных обучающей выборки, создаются несколько сетей с определенной топологией и определенными весами. Данные сети тестируются, а их производительность сравнивается. После из этих сетей выбираются наиболее эффективные и скрещиваются между собой. Полученная сеть является своего рода порождающим элементом для нового поколения сетей [300]. Далее процесс тестирования, скрещивания и порождения повторяется для всех последующих поколений, пока не будет достигнута необходимая эффективность.

Эволюционный подход позволяет реализовать алгоритм обучения в процессе функционирования за счет использования методов обучения с подкреплением. Для поиска наиболее производительной архитектуры ИНС, адаптации топологии нейросетевой модели ИИ и ее весовых коэффициентов используются генетические алгоритмы. Однако, как уже было обозначено в главе 1, накладные расходы на производительность и объем обучающей выборки для данного подхода очень высоки. Кроме того, существующие методы направлены на классические модели нейронов (полносвязные, сверточные), и их крайне сложно адаптировать к использованию применительно к корреляционным нейронам.

Методы глубокого обучения с подкреплением перспективны и достаточно активно развивается. Однако, по мнению ведущих ученых, высокие результаты достижимы только при пакетном обучении на больших объемах данных [127].

Так как в настоящем исследовании обучение подразумевает использование относительно малых выборок, от эволюционного подхода и методов глубокого обучения с подкреплением решено отказаться.

В отличие от эволюционного подхода, где активно используются генетические алгоритмы, направленные на «усиление сильных сторон» сети, иммунные алгоритмы также направлены на поддержание разнообразия вычислительных элементов (которые в зависимости от конкретного иммунологического подхода принято называть антителами или детекторами), что приводит к более качественным решениям всей системы. Под разнообразием понимается стремление к снижению корреляции между ошибочными решениями

вычислительных элементов (детекторов или нейронов), а также «закладка» в сетевой классификатор некоторой избыточности, которая позволяет принимать решения в тех случаях, которые не учитывались при обучении.

Преимущество иммунных моделей заключается в том, что они обладают свойством двойной пластичности [47], позволяющего относительно легко изменять в процессе функционирования не только собственные параметры, но и структуру. Многослойные нейронные сети гораздо сложнее масштабировать в процессе функционирования и дообучать в условиях неопределенности, т.е. уже после пакетного обучения с учителем. По этим причинам в настоящей работе выбран иммунный подход к построению и обучению адаптивной модели ИИ.

3.3 Иммунные модели машинного обучения и их применение в биометрических системах

Прежде всего, отметим, что ИИС (как и ИНС) – крайне упрощенная конструкция, которая не подразумевает строгого соответствия своему биологическому прототипу. Исследователи определяют ИИС как «адаптивные системы, основанные на теоретической иммунологии и имеющихся иммунных функциях, принципах и моделях, которые применяются для решения проблем» [342]. Данная область развивается с 1985 года [188]. Изначально ИИС рассматривалась в качестве классификатора. Позже ИИС стали применяться к более широкому кругу задач, включая регрессию и оптимизацию [276]. За последние три десятилетия научное сообщество разработало разнообразный набор алгоритмов, основанных на следующих базовых теориях функционирования естественной иммунной системы:

1. Алгоритмы на основе теории опасности и дендритных клеток (DCA), разработанной Polly Matzinger в 2002 году [235].

2. Алгоритмы на основе теории негативного (отрицательного) отбора. Впервые алгоритм отрицательного отбора предложен Форестом и др. в 1994 году

по отношению к проблеме обнаружения компьютерных вирусов. С тех пор было разработано несколько вариантов алгоритма отрицательного отбора, хотя основные характеристики исходного алгоритма отрицательного выбора все еще остаются неизменными [227].

3. Алгоритмы на основе теории клональной селекции (положительного отбора), выдвинутой в 1959 году Бурнетом [186]. Наиболее известным базовым алгоритмом ИИС является CLONALG, который в 2000 году был предложен de Castro и von Zuben для решения задач распознавания образов [212]. После он был адаптирован к задачам оптимизации [217]. Вариация клеток-кандидатов в CLONALG в некоторой степени похожа на генетический алгоритм. CLONALG обладает способностью к обучению и развивает высококачественную память.

4. Алгоритмы на основе теории иммунной сети, разработанной Нильсом К. Жерне в 1974 году. Первый алгоритм иммунной сети был предложен Ishida в 1990 году, позже Timmis и др. переопределили и повторно внедрили модель искусственной иммунной сети и представили модель под названием AINE (Artificial Immune NEtwork) [264]. Стюарт и Карнейро в 1999 году предложили расширенную модель иммунной сети, которая стала известна как модель третьего поколения иммунных сетей. Оно включает концепцию центральной иммунной системы (ЦИС) и периферической иммунной системы (ПИС) [168].

Известны также алгоритмы на основе теории гуморального иммунного ответа [210] и модели рецептора распознавания образов [357].

Среди российских ученых большой вклад в развитие искусственных иммунных систем и сетей внес Ю.А. Брюхомицкий [17-22]. В его работах и работах его научной группы были обозначены перспективы использования ИИС в динамической биометрии, исследованы и предложены различные архитектуры ИИС и рассматривались, в том числе, следующие вопросы:

– генерация детекторов в процессе обучения и функционирования (самообучения) ИИС, принципы двойной пластичности (структурной и параметрической) [19], выделение и обоснование гиперпараметров, влияющих на работу ИИС;

- удаление вторичных детекторов в процессе функционирования ИИС, «старение» ИИС [20], баланс реагировавших ранее и новых детекторов, позволяющий провести черту между способностью ИИС к восприятию новой информации (самообучению) и формированием памяти (этот вопрос тесно связан с проблемами переобучения);
- меры близости для сопоставления детекторов (антител) и антигенов, распознавание субъектов по клавиатурному почерку на основе ИИС [18];
- распознавание рукописных образов на основе ИИС [21];
- алгоритмы клональной селекции, распознавание субъектов по голосу на основе ИИС [22].

Эти и другие работы заложили базис для дальнейших исследований и разработки гибридных нейро-иммунных ПБК. Кроме того, Ю.А. Брюхомицкий применял и другие подходы, в том числе нейросетевой [17], для построения биометрических систем, а также сравнивал нейросетевой и иммунологический подходы в задачах распознавания рукописных текстов [21].

Подробнее с этими направлениями исследований, а также с основными алгоритмами ИИС и теориями, лежащими в их основе, можно ознакомиться в работах [90, 173, 202, 204, 233, 234, 271, 293, 294, 353].

Результаты [293] исследования показывают, что использование алгоритмов ИИС позволяет добиться надежности распознавания человека по параметрам радужки в 90% и по параметрам подписи в 83,88%. Также алгоритмы использовались в системах распознавания по голосу [227], проверки грамматики английского языка [271], аудио поиска по контенту [294] и распознавания цифр [233].

В работе [283] предлагается объединить иммунную сеть с генетическим алгоритмом для распознавания человека по лицу. Надежность предложенного метода составляет 99,7%. Все эксперименты выполнялись с одним случайно выбранным учебным изображением (девять тестовых изображений на человека по 40 тренировок, то есть 360 тестовых изображений). Каждый случай повторялся 30

раз, при этом выбирались различные комплекты обучения и тестирования. В работах [57, 101, 211, 219] также говорится о повышении надежности распознавания образов при использовании иммунных алгоритмов.

В таблице 3.2 представлен перечень биометрических систем, построенных на основе ИИС.

В работе 2005 года [345] описаны проблемы аппарата ИИС, к которым относится его слабая теоретизация (недостаток строгих доказательств работоспособности, сходимости алгоритмов обучения). Эта проблема остается актуальной в 2020 году [80, 292], хотя работы по расширению доказательной базы ведутся [207]. Поэтому решено взять учитывать общие принципы существующих подходов к построению и обучению ИИС, но не брать за основу ни один из базовых алгоритмов и моделей.

В обзоре [80] также представлена сопостовительная информация о достигнутых результатах в области распознавания биометрических данных на основе иммунных подходов к классификации образов.

Таблица 3.2. Достигнутые результаты по применению иммунных алгоритмов в задачах идентификации и верификации биометрических образов.

Авторы	Тип биометрического образа	Алгоритм ИИС	Параметры эксперимента	Процент верных решений
K.M. Faraoun and A Boukelif [227]	Голос (текстозависимая система)	Алгоритм отрицательного отбора	База данных из записей фраз 10 дикторов	64%
A.K. Muda, S.M. Shamsuddin [186]	Рукописный	Алгоритм отрицательного отбора	Условия эксперимента не описаны	90%
Guan-Chun Luh [283]	Лицо	Алгоритм клональной селекции	40 человек (4 женщины и 36 мужчин), по 10 изображений лица на каждого, обучающая выборка – 1 изображение, тестовая – 9	99,7%
Р. М. Михерский, [101]	Лицо	Алгоритм отрицательного отбора	7 человек	85,7%

Продолжение таблицы 3.2

Еременко Ю.И., Мельникова И.В., Шаталов [57]	Рукописный	Алгоритм клональной селекции, алгоритм дендритных клеток, AINET (сетевой алгоритм)	База данных из 200 рукописных классов образов	70%
C. Djeddi and L. Souici-Meslati [219]	Рукописный	AIRS1, AIRS2, Parallel AIRS2.	База данных IFN/ENIT (130 человек, по 5 образцов на каждого), обучающая выборка – 390 образцов текста (по 3 образца от каждого субъекта), тестовая – 260	87,77%
U.Garain, Mangal P. Chakraborty, D. Dasgupta [292]	Рукописный	Алгоритм клональной селекции	Два набора данных: - 122556 рукописных образцов цифр, написанных 1049 людьми, - 212938 рукописных образцов цифр, написанных 556 людьми.	96%

Настоящее исследование апеллирует к хорошо зарекомендовавшим себя методам построения и обучения ансамблей моделей [360] для усиления формально-теоретической основы предлагаемых решений.

3.4 Модель искусственной иммунокомпетентной клетки на базе корреляционного нейрона

Как уже было указано, иммунная система содержит множество клеток (макрофаги, дендритные клетки, лимфоциты), которые обладают способностью обнаруживать и удалять чужеродные организмы (антигены). Назовем все такие клетки *детекторами* [80] – вычислительными элементами, способными анализировать распознаваемый образ либо его отдельные фрагменты и реагировать на него пропорционально тому, насколько этот образ соответствует антигену. Шкала реакций (аффинности) задана на интервале действительных чисел [0;1], где 0 – полная уверенность в том, что клетка принадлежит организму

(гипотеза «Свой»), а 1 – полная уверенность в обратном (гипотеза «Чужой»). Каждый детектор следует рассматривать как бинарный классификатор, состоящий из нескольких функций, последовательно применяющихся к биометрическому образу. Образ представляет собой вектор признаков фиксированной длины $\bar{a} = \{a_1, a_2, \ldots, a_n\}$, где n – количество признаков, которое должно присутствовать в образе. В общем виде получение реакции i-го детектора на входной образ $\bar{a}$ можно описать формулой (3.1):

$$u_i = \phi_\chi\left(y' = \varphi\left(y = f_r(\bar{\alpha} = R(\bar{a}, \Psi_i), \breve{g}, \Theta_i), T_i\right)\right) \quad (3.1)$$

Опишем функции детектора и их параметры:

1. $\bar{\alpha} = R(\bar{a}, \Psi_i)$ – функция-рецептор, предоставляющая интерфейс взаимодействия для детектора и антигена. Данная функция извлекает η из n признаков, содержащихся в $\bar{a}$, Ψ_i – множество номеров признаков из $\bar{a}$, которые должен анализировать i-й детектор. Вектор $\bar{\alpha}=\{a_1,a_2,\ldots,a_\eta\}$ является подмножеством $\bar{a}$ с собственной сквозной нумерацией элементов;

2. $y = f_r(\bar{\alpha}, \breve{g}, \Theta_i)$ – функция-ядро детектора, параметрический функционал, который вычисляет близость вектора $\bar{\alpha}$ к эталону класса образов «Свой»; r – тип функционала (ниже даны формулы (3.2)-(3.11)); $\breve{g}$ – вектор параметров, которые влияют на характер вычислений; $\Theta_i=\{\mu_1,\mu_2,\ldots,\mu_\eta,\sigma_1,\sigma_2,\ldots,\sigma_\eta\}$, μ_j и σ_j – параметры распределения j-го признака из вектора $\bar{\alpha}$ (для класса «Свой»). Данные из множества Θ_i рассчитываются на основании нескольких случайных примеров из обучающей выборки (далее *фолд*). Выбор мер близости, на которых базируются детекторы, обусловлен результатами исследований, представленных в [52, 118, 282, 310, 329] и в Приложении А. В настоящей работе для построения адаптивной модели ИИ, работающей в незащищенном режиме, для построения ядер детекторов применялись следующие функционалы:

2.1 Чувствительные к корреляции метрики (корреляционные метрики) – мера Минковского (3.2), разные вариации меры Байеса-Минковского (3.3)-(3.5).

$$f_1(\bar{\alpha}, \breve{g} = \{g\}, \Theta_i) = \sqrt[g]{\sum_{j=1}^{\eta}\left|\frac{(m_j - a_j)}{\sigma_j}\right|^g}, \qquad f_1(\bar{\alpha}, \breve{g} = \{g\}, \Theta_i) = \sqrt[g]{\sum_{j=1}^{\eta}\left|\frac{(\mu_j - a_j)}{\delta_j}\right|^g}, \quad (3.2)$$

$$f_2(\overline{\alpha},\breve{g}=\{g\},\Theta_i)=\sum_{j=1}^{\eta}\left|\left|\frac{(m_t-a_t)}{\sigma_t}\right|^g-\left|\frac{(m_j-a_j)}{\sigma_j}\right|^g\right|,\qquad f_2(\overline{\alpha},\breve{g}=\{g\},\Theta_i)=\sum_{j=1}^{\eta}\left|\left|\frac{(\mu_t-a_t)}{\delta_t}\right|^g-\left|\frac{(\mu_j-a_j)}{\delta_j}\right|^g\right|, \quad (3.3)$$

$$f_3(\overline{\alpha},\breve{g}=\{g\},\Theta_i)=\sqrt[g]{\sum_{j=1}^{\eta}\left|\left|\frac{(m_t-a_t)}{\sigma_t}\right|^g-\left|\frac{(m_j-a_j)}{\sigma_j}\right|^g\right|},\qquad f_3(\overline{\alpha},\breve{g}=\{g\},\Theta_i)=\sqrt[g]{\sum_{j=1}^{\eta}\left|\left|\frac{(\mu_t-a_t)}{\delta_t}\right|^g-\left|\frac{(\mu_j-a_j)}{\delta_j}\right|^g\right|}, \quad (3.4)$$

$$f_4(\overline{\alpha},\breve{g}=\{g\},\Theta_i)=\sqrt[g]{\sum_{j=1}^{\eta}\left|\frac{a_t}{\sigma_t}-\frac{a_j}{\sigma_j}\right|^g},\qquad f_4(\overline{\alpha},\breve{g}=\{g\},\Theta_i)=\sqrt[g]{\sum_{j=1}^{\eta}\left|\frac{a_t}{\delta_t}-\frac{a_j}{\delta_j}\right|^g}, \quad (3.5)$$

Изменение степенного коэффициента g є [0,01;100] позволяет давать более точную оценку близости при разных уровнях взаимной коррелированности признаков. Желательно, подбирать признаки Ψ_i таким образом, чтобы коэффициенты парной корреляции $C_{j,t}$ были близки по значению [66], причем для меры Минковского $C_{j,t}$<0,5, а для меры Байеса-Минковского $C_{j,t}$>0,5.

2.2 Нечувствительные к корреляции метрики – «наивный Байес» в дифференциальной (3.6) и интегральной форме (3.7), параметрические критерии (3.8)-(3.11). Разные функционалы образуют различные виды детекторов, которые дают слабо коррелированные решения относительно друг друга. Из любого функционала можно получить разные меры близости за счет изменения параметров $\breve{g}$. Часть представленных мер близости применялась в работе [147] при построении «гибких» нейронных сетей.

$$f_5(\overline{\alpha},\breve{g}=\{g_1,g_2,...,g_\eta\},\Theta_i)=\prod_{j=1}^{\eta}p_{g_j}(a_j,m_j,\sigma_j), \quad (3.6)$$

$$f_6(\overline{\alpha},\breve{g}=\{g_1,g_2,...,g_\eta\},\Theta_i)=\prod_{j=1}^{\eta}P_{g_j}(a_j,m_j,\sigma_j), \quad (3.7)$$

где $P_g(a,m,\sigma)$ и $p_g(a,m,\sigma)$ – значение функции распределения и плотности вероятности, соответственно, с учетом признака a_j и его параметров распределения (m_j и σ_j) для класса «Свой». Реализация этих функций зависит от закона распределения, который определяется параметром g_j. В настоящей работе использовалось три вида закона распределения: нормальный, логнормальный, Лапласса. Большинство биометрических признаков имеют законы распределения, близкие к указанным выше [329].

При решении иных задач распознавания образов перечень законов распределения может быть расширен.

- *нормальный* (g_j = 1):

$$p_1(a_j, m_j, \sigma_j) = \frac{1}{\sigma_j\sqrt{2\pi}} e^{-\frac{(a_j - m_j)^2}{2\sigma_j^2}},$$

$$P_1(a_j, m_j, \sigma_j) = \frac{1}{\sigma_j\sqrt{2\pi}} \int_{m_j - 5\sigma_j}^{a_j} e^{-\frac{(\vartheta - m_j)^2}{2\sigma_j^2}} d\vartheta,$$

где m_j и σ_j – математическое ожидание и среднеквадратичное отклонение j-го признака;

- *логнормальный* (g_j = 2):

$$p_2(a_j, m_j, \sigma_j) = \frac{1}{a_j\sigma_j\sqrt{2\pi}} e^{-\frac{(Ln(\vartheta) - m_j)^2}{2\sigma_j^2}},,$$

$$P_2(a_j, m_j, \sigma_j) = \frac{1}{a_j\sigma_j\sqrt{2\pi}} \int_{m_j - 5\sigma_j}^{a_j} e^{-\frac{(Ln(\vartheta) - m_j)^2}{2\sigma_j^2}} d\vartheta,,$$

где m_j и σ_j – параметр масштаба и формы;

- закон распределения *Лапласа* (g_j = 3):

$$p_3(a_j, m_j, \sigma_j) = \frac{\sigma_j}{2} e^{-\sigma_j|a_j - m_j|},$$

$$P_3(a_j, m_j, \sigma_j) = \begin{cases} 0{,}5e^{\sigma_j(a_j - m_j)}, a_j \le m_j \\ 1 - 0{,}5e^{-\sigma_j(a_j - m_j)}, a_j > m_j \end{cases},$$

где m_j и σ_j – коэффициенты сдвига и масштаба;

$$f_7(\overline{\alpha}, \breve{g} = \{g\}, \Theta_i) = \int_{-5}^{5} \sqrt[g]{|P_1(\vartheta, m_{\grave{a}}, \sigma_{\grave{a}}) - P_1(v, 0, 1)|^g} \cdot d\vartheta \quad (3.8)$$

$$f_8(\overline{\alpha}, \breve{g} = \{g\}, \Theta_i) = \int_{-5}^{5} \sqrt[g]{|p_1(\vartheta, m_{\grave{a}}, \sigma_{\grave{a}}) - p_1(\vartheta, 0, 1)|^g} \cdot d\vartheta \quad (3.9)$$

$$f_9(\overline{\alpha}, \breve{g} = \{g\}, \Theta_i) = \int_{-5}^{5} \sqrt[g]{|P_1(\vartheta, m_{\grave{a}}, \sigma_{\grave{a}}) \cdot (1 - P_1(\vartheta, 0, 1))|^g} \cdot d\vartheta \quad (3.10)$$

$$f_{10}(\overline{\alpha}, \breve{g} = \{g\}, \Theta_i) = \int_{-5}^{5} \sqrt[g]{|p_1(\vartheta, m_{\grave{a}}, \sigma_{\grave{a}}) \cdot p_1(\vartheta, 0, 1)|^g} \cdot d\vartheta \quad (3.11)$$

Группа функционалов (3.8)-(3.11) представляет собой модификации критериев согласия, используемых для сравнения эмпирически

наблюдаемой функции распределения (плотности вероятности) для величины $\mathring{a}$ с теоритической (эталонной), при этом предполагается, что закон распределения близок к нормальному, $\dot{a}$ – нормированное значение любого признака, $\mu_{\mathring{a}}$ и $\sigma_{\mathring{a}}$ – математическое ожидание и среднеквадратичное отклонение. Нормирование (приведение всех a_j к единой случайной величине $\dot{a}$) производится по данным из Θ_i в соответствии с формулой:

$$\dot{a} = \frac{a_j - m_j}{\sigma_j},$$

Параметры распределения $\mathring{a}$ для класса «Свой» должны принимать стандартные значения ($\mu_{\mathring{a}} \approx 0$ и $\sigma_{\mathring{a}} \approx 1$), для образов «Чужих» будут наблюдаться значительные отклонения ($\mu_{\mathring{a}} \neq 0$ и/или $\sigma_{\mathring{a}} > 1$). Предложенные функционалы (3.8)-(3.11) обобщают критерии Джини, среднего геометрического вероятности и плотности вероятности, рассмотренные в работах [136, 316].

2.3 В защищенном режиме исполнения возможно использовать только функционал (3.12) в основе детектора:

$$f_{11}(\bar{\alpha}, \breve{g} = \{g\}, \Theta_i) = \sqrt[g_1]{\frac{1}{\eta}\sum_{j^*=1}^{n'} w_{j^*}\left(a'_{j^*} - m'\right)^{g2}} = \sqrt[g_1]{\frac{1}{\eta}\sum_{t=1}^{\eta} w_t\left(a'_t - m'\right)^{g2}}, \qquad (3.12)$$

$$m' = \frac{1}{\eta}\sum_{t=1}^{\eta} a'_t,\ a'_t = a'_{t,j} = f(a_t, a_j) = \left|\left|\frac{a_t}{\delta_t}\right|^{g_3} - \left|\frac{a_j}{\delta_j}\right|^{g_3}\right|,$$

где веса w_t вычисляются в соответствии с (2.12).

3. $y' = \varphi(y, T_i)$ – функция нормирования откликов y относительно порога T_i, который вычисляется в процессе настройки i-го детектора. Функция нормирования может иметь 2 реализации:

$$\varphi(y, T_i) = y / T_i,$$

где y – это расстояние от $\bar{\alpha}$ до эталона класса «Свой» (чем меньше, тем ближе), тогда T_i – это максимальное значение функции-ядра i-го детектора, при поступлении на его вход обучающих образов «Свой», либо:

$$\varphi(y, T_i) = T_i / y,$$

где y – это вероятность того, что $\bar{\alpha}$ принадлежит классу «Свой», тогда T_i – это минимальное значение функции-ядра i-го детектора, при поступлении на его вход обучающих образов «Свой». Физический смысл y (расстояние или вероятность) зависит от функции-ядра детектора (например, для меры Евклида требуется использовать первый вариант, а для «наивного Байеса» – второй);

4. $u_i = \phi_\chi(y'_i)$ – функция активации, дополнительный нелинейный элемент детектора, который определяет особенности реагирования на антиген. Функция активации также необходима, чтобы привести отклик детектора к области значений [0;1]. В настоящей работе применялись сигмоиды (арктангенс, гиперболический тангенс и др.). В качестве функций активации имеет смысл использовать либо наиболее быструю из сигмоидальных, либо применять функции, которые дают как можно более отличающиеся результаты, чтобы создавать детекторы с низкой коррелированностью решений на базе однотипных мер близости (в целом r в большей степени влияет на характер преобразований (1) детектора, чем χ).

Одной из теоритических проблем аппарата ИИС является слабая обоснованность используемых мер близости [345] (чаще всего, применяется мера Евклида). Согласно теореме «об отсутствии бесплатных завтраков» (No Free Lunch) ни одна мера близости не может быть оптимальной для всего множества задач распознавания образов. В настоящей работе каждый детектор определяет аффинность уникальным способом, а состав детекторов «подстраивается» под задачу и определяется в процессе обучения адаптивной модели ИИ.

3.5 Адаптивная нейро-иммунная модель искусственного интеллекта на основе иммунного подхода

Предлагается разделить детекторы на две группы: врожденный и приобретенный иммунитет, и рассматривать их как *два комитета (ансамбля)* слабых классификаторов, обучаемых при помощи разных алгоритмов.

Коллективное решение комитета из N детекторов может быть вычислено как среднее частных решений:

$$\ddot{u} = \Phi(\overline{D}^* = \{D_1^*, \ldots D_N^*\}, \bar{a}) = \frac{1}{N} \cdot \sum_{i=1}^{N} \phi(D_i^*, \bar{a}) = \frac{1}{N} \cdot \sum_{i=1}^{N} u_i \,.$$

Врожденный иммунитет (ВИ) передается посредством генов, органы иммунной системы формируются еще при эмбриональном развитии. У эмбрионов В-лимфоциты образуются в печени и костном мозге. В предлагаемой модели костный мозг является местом пребывания иммунокомпетентных детекторов, параметры и состав которых определяются в ходе эмбрионального (и постэмбрионального) развития. Так ВИ формируется в процессе итерационного обучения с использованием *тренировочной* и *валидационной* выборок (рисунок 3.9). Последняя используется для промежуточной оценки надежности решений модели при смене поколения детекторов. Обе выборки являются *непересекающимися* подмножествами *обучающей* выборки.

Приобретённый иммунитет (ПИ) развивается с течением жизни и определяет способность организма обезвреживать специфические антигены, которые попадали в организм ранее. В предложенной модели тимус осуществляет настройку и отбор иммунокомпетентных детекторов, используя валидационную выборку. Адаптивный иммунный ответ приводит к появлению иммунологических клеток памяти (представленных детекторами), которые долгое время пребывают в «спящем состоянии» до повторной встречи с тем же антигеном. Приобретенный иммунитет формируется в процессе функционирования модели. Если решение об отнесении образа к категории «Свой» или «Чужой» является неоднозначным, могут генерироваться новые иммунокомпетентные детекторы.

Идея объединения классификаторов в комитет основана на теореме Кондорсе, которая утверждает: если мнения экспертов независимы, и вероятность правильного решения каждого из них больше 0,5, то с увеличением количества экспертов вероятность правильного решения комитета экспертов возрастает и стремится к единице. Причем, чем выше вероятность верного решения для каждого эксперта в отдельности, тем выше вероятность верного решения

комитета. Отметим, что решение любого детектора можно инвертировать, чтобы преодолеть барьер Кондорсе в 0,5 (известны также доказательства других теорем [136], позволяющих обойти барьер Кондорсе).

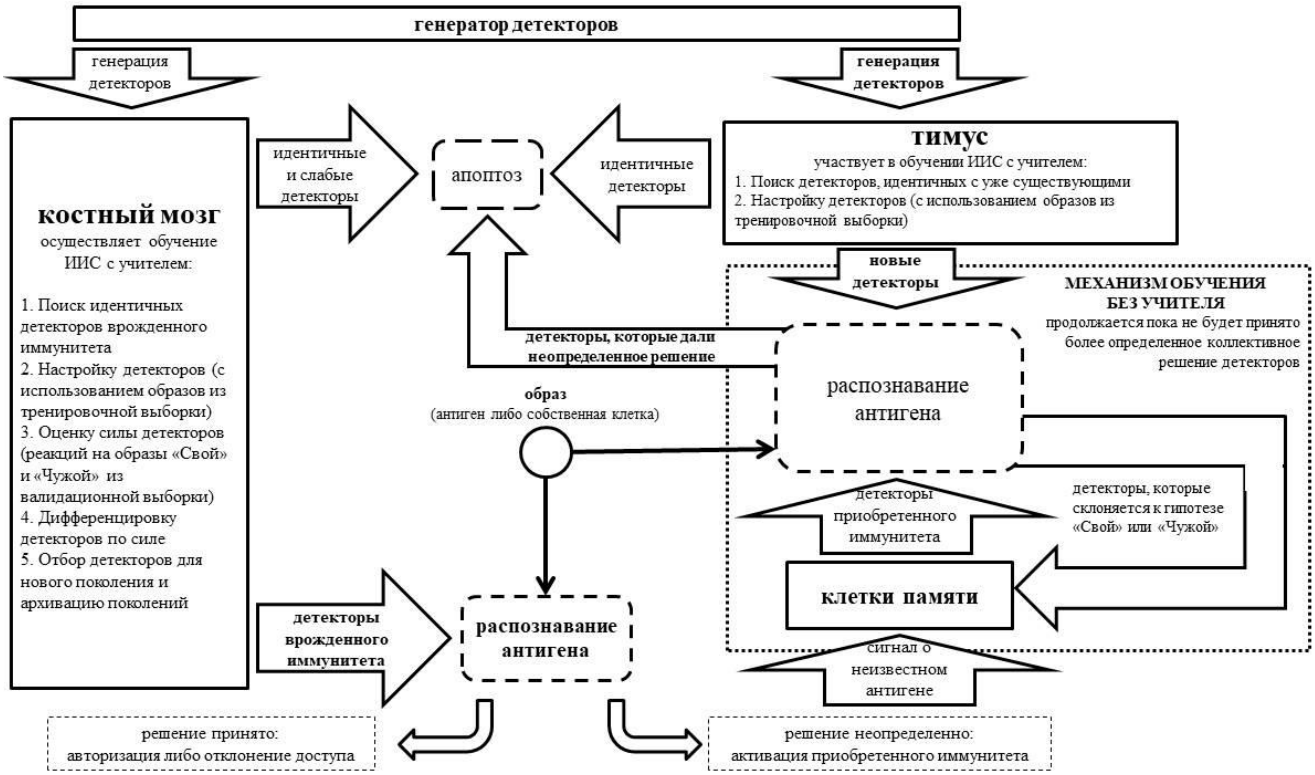


Рисунок 3.9 – Функциональная схема адаптивной нейро-иммунной модели ИИ, основанной на иммунном подходе

Однако на практике решения классификаторов, играющих роль экспертов, в той или иной мере коррелированы, чем ниже коррелированность решающих правил, тем более ощутим положительный эффект при их комплексировании. Таким образом, имеются следующие основные характеристики (в том числе *гиперпараметры*), которые влияют на эффективность комитета детекторов:

- N – количество детекторов (гиперпараметр);

- RD – матрица коэффициентов корреляции Пирсона $C(\bar{u}_i, \bar{u}_l)$ между решениями всех возможных пар детекторов, где $\bar{u}_i$ – вектор реакций i-го детектора на примеры образов «Чужих» из тренировочной или валидационной выборки;

- Δu – сила детекторов, их способность давать как можно более высокие показатели разницы средних уровней реакции на образы «Свой» и «Чужой» (3.13):

$$\Delta u=\mu_u^{(Ч)}-\mu_u^{(С)}, \quad \mu_u^{(Ч)}>\mu_u^{(С)} \qquad (3.13)$$

В рамках предыдущих исследований [143] была апробирована стратегия снижения уровня коррелированности решений детекторов, которая оказалась недостаточно продуктивной. Процедура оценки показателей $C(\bar{u}_i, \bar{u}_l)$ и дифференциации по ним детекторов оказалась ресурсоемкой. Кроме того, не наблюдалось сходимости алгоритма настройки модели. Процесс обучения был слишком длительным и не всегда приводил к ожидаемому результату (не всегда удавалось найти N детекторов с заданным минимальным уровнем взаимной коррелированности решений). По этой причине в настоящей работе выбрана стратегия повышения силы детекторов при условии, что они не должны быть идентичными. При появлении идентичных или слабых детекторов происходит *апоптоз* – процесс программируемой клеточной гибели для уничтожения дефектных клеток (рисунок 3.9).

Необходимо, чтобы решения всех детекторов врожденного и приобретенного иммунитета не являлись полностью коррелированными. Поэтому после генерации детектора должна осуществляться проверка идентичности параметров нового детектора и уже существующих. При обнаружении «двойника» его следует удалить и сгенерировать детектор снова. При этом значения параметров $\breve{g}$ можно считать равными, когда они отличаются менее чем на 10^{-1}. Конечно, решения детекторов будут в той или иной степени коррелированы (но не на 100%). Чем сильней различаются параметры D_i и D_l, тем менее коррелированы решения i-го и l-го детекторов.

Детектор можно описать множеством параметров $D_i = \{\Psi_i, \breve{g}, r, \chi\}$. *Сгенерировать детектор* означает сгенерировать данные параметры. Приведем псевдокод функции генерации детектора:

method GenerateDetector(RF)

// RF – матрица парных коэффициентов корреляции

// между признаками

// random(*min*; *max*) – генерация случайного числа

η = random(2;n/2)

C_{max} = random(0,1; 1)

if C_{max} > 0,5 **then** C_{min} = random(0; C_{max})

else C_{min} = random(-0,5; C_{max})

// Выбрать признаки с уровнем взаимной

// корреляции более C_{min}, но менее C_{max}

Ψ_i = GetFeatures(C_{min}; C_{max}; RF)

// Если признаков с заданным уровнем корреляции

// нет, сгенерировать номера признаков снова

if Length(Ψ_i)< η **go to begin**

// Выбор меры близости (Мера Минковского не

// подходит для обработки сильно коррелированных

// признаков, а Байеса-Минковского – для слабо

// коррелированных)

if C_{max} > 0,5 **then** r = random(2; 11)

else r = random(5; 11)

if r = 11 then

if C_{max} < 0,5 then

do

C_{max} = - C_{max}

C_{min} = C_{min} - 0,5

end

if $5 \leq r \leq 6$ **then**

for j **from** 1 **to** η **do** g_j = random(1; 3) **end**

else g = random(0,01; 100)

// Выбрать случайную функцию активации

χ = GetRandomActivationFunction()

$D_i = \{\Psi_i, \breve{g}, r, \chi\}$

// Если новый детектор идентичен одному из

// детекторов ВИ или ПИ, то сгенерировать заново

if IsIdenticalToExisting(D_i) = true **then**

D_i = GenerateDetector(RF)

return D_i

В разработанной модели реализуется идея *случайных подпространств признаков*, но в отличие от алгоритма «случайный лес» Ψ_i задается с учетом корреляции между признаками. Этот прием называется *симметризацией корреляционных связей* [254].

Другая идея, которая реализована при генерации детекторов, заключается в объединении разнородных *случайных классификаторов* (например, описывая признак разными законами распределения можно получить несколько «наивных» классификаторов Байеса, решения которых не полностью коррелированы). Примером подобной техники является нейросетевое обобщения множества различных критериев [136].

Настройка детектора связана с вычислением порога T_i и эталонных описаний признаков Θ_i (μ_j и σ_j). Настроенный детектор можно обозначить как $D^*_i = \{\Psi_i, \breve{g}, r, \chi, \Theta_i, T_i\}$, а функцию (3.1), как $\phi(D^*_i)$.

3.6 Алгоритм пакетного обучения адаптивной нейро-иммунной модели искусственного интеллекта с учителем

Известны следующие базовые методы и подходы для обучения ансамблей моделей:

1. Бэггинг (bootstrap aggregating) – мета-алгоритм композиционного обучения машин, основная идея которого заключается в обучении базовых (слабых) классификаторов на разных подмножествах обучающей выборки. Базовые классификаторы могут быть идентичными или иметь разные архитектуры. Бэггинг уменьшает дисперсию голосов базовых классификаторов и помогает избежать переобучения. Принцип работы бэггинга схож с принципами работы метода случайных классификаторов, а также методов

накопления сигналов при их обнаружении и заключается в повышении отношения сигнал/шум.

2. Бустинг (boosting) – семейство алгоритмов машинного обучения, преобразующих слабые обучающие алгоритмы к сильным. Бустинг строит ансамбль путём тренировки каждого нового классификатора, уделяя больше внимания обучению на тех тренировочных примерах, которые предыдущие модели классифицировали ошибочно (например, путем присвоения весов обучающим примерам), и имеет тенденцию к переобучению. Эффективность этого подхода доказана экспериментально и теоритически, что впервые подтверждено для алгоритма AdaBoost [198].
3. Стекинг (stacked generalization) предполагает построение многослойных структур из ансамблей классификаторов, когда выходные данные ансамбля первого слоя воспринимаются ансамблем второго слоя, как входные данные (мета-признаки). При использовании стекинга увеличивается необходимый для обучения объем выборки, так как для корректной настройки мета-модели каждый слой нужно обучать на разных тренировочных примерах.

При разработке итерационного алгоритма обучения модели были учтены первые два подхода (бэггинг позволяет компенсировать склонность к переобучению бустинга), но от стекинга решено отказаться, учитывая малый объем обучающей выборки и что ИИС не образует конструкций в виде слоев.

В разработанном алгоритме на каждой итерации происходит генерация новой популяции детекторов, которые настраиваются с учетом нескольких случайных тренировочных примеров (бэггинг) и выполняется промежуточная оценка их эффективности, как на тренировочной, так и на валидационной выборке (рисунок 3.10), слабые детекторы уничтожаются (апоптоз), в результате появляется новое поколение иммунокомпетентных (более эффективных) детекторов. Мерой эффективности (обученности) детекторов можно считать Δu (3.13). По результатам последней валидации вычисляются оценки $\mu_u^{(С)}$ и $\mu_u^{(Ч)}$ для коллективного решения детекторов ВИ. Эти параметры используются для построения *интервала неопределенности решения (ИНР)* $[\mu_u^{(С)}; \mu_u^{(Ч)}]$. ИНР

является частью механизма подкрепления при онлайн-обучении модели. Этот механизм активируется при формировании ПИ, о чем будет изложено в следующем параграфе.

На каждой итерации обучения синтезируются новые образы «Чужих» (рисунок 3.10) путем скрещивания тренировочных примеров, которые хуже всего классифицируются детекторами ВИ (далее сильные «Чужие»). Сильные «Чужие» дают наименьшую среднюю совокупную реакцию детекторов *ïi*. Скрещивание образов $\bar{a}_k$ и $\bar{a}_z$ происходит с помощью линейной интерполяции значений признаков в соответствии с ГОСТ Р 52633.2-2010.

$$a_{к,j} = \frac{K_{syn} + 1 - k}{K_{syn} + 1} \cdot a_{k,j} + \frac{k}{K_{syn} + 1} \cdot a_{z,j}, \quad (3.14)$$

где K_{syn} – количество синтетических примеров, порождаемых парой «сильных Чужих» предыдущего поколения (в настоящей работе K_{syn} = 1), k – номер синтетического примера, j – номер признака. Этот способ синтеза «Чужих» эффективен, если признаки имеют законы распределения близкие к нормальному.

Синтетические образы, получаемые путем скрещивания сильных «Чужих», добавляются в тренировочную выборку. Детекторы нового поколения настраиваются с учетом синтетических примеров, что позволяет им лучше классифицировать образы «Чужих», наиболее близких к образам «Свой». Таким образом, *модель одновременно «учится» создавать образы более сильных «Чужих» и распознавать их. Предложенный механизм размножения сильных «Чужих» при обучении является еще одной вариацией бустинга.*

На скорость и эффективность алгоритма обучения, представленного на рисунке 3.10, более всего влияют следующие основные параметры:

- $I_{ВИ}$ – количество итераций обучения (валидаций);
- $N_{ВИ}$ – количество детекторов ВИ, которые остаются после валидации;
- N_{gen} – количество генерируемых детекторов на каждой итерации;
- N_{valid} – количество детекторов, которые отсеиваются на этапе валидации;
- Q – количество сильных «Чужих» (на каждой итерации генерируется $K_{syn} \cdot Q \cdot (Q - 1)/2$ примеров).

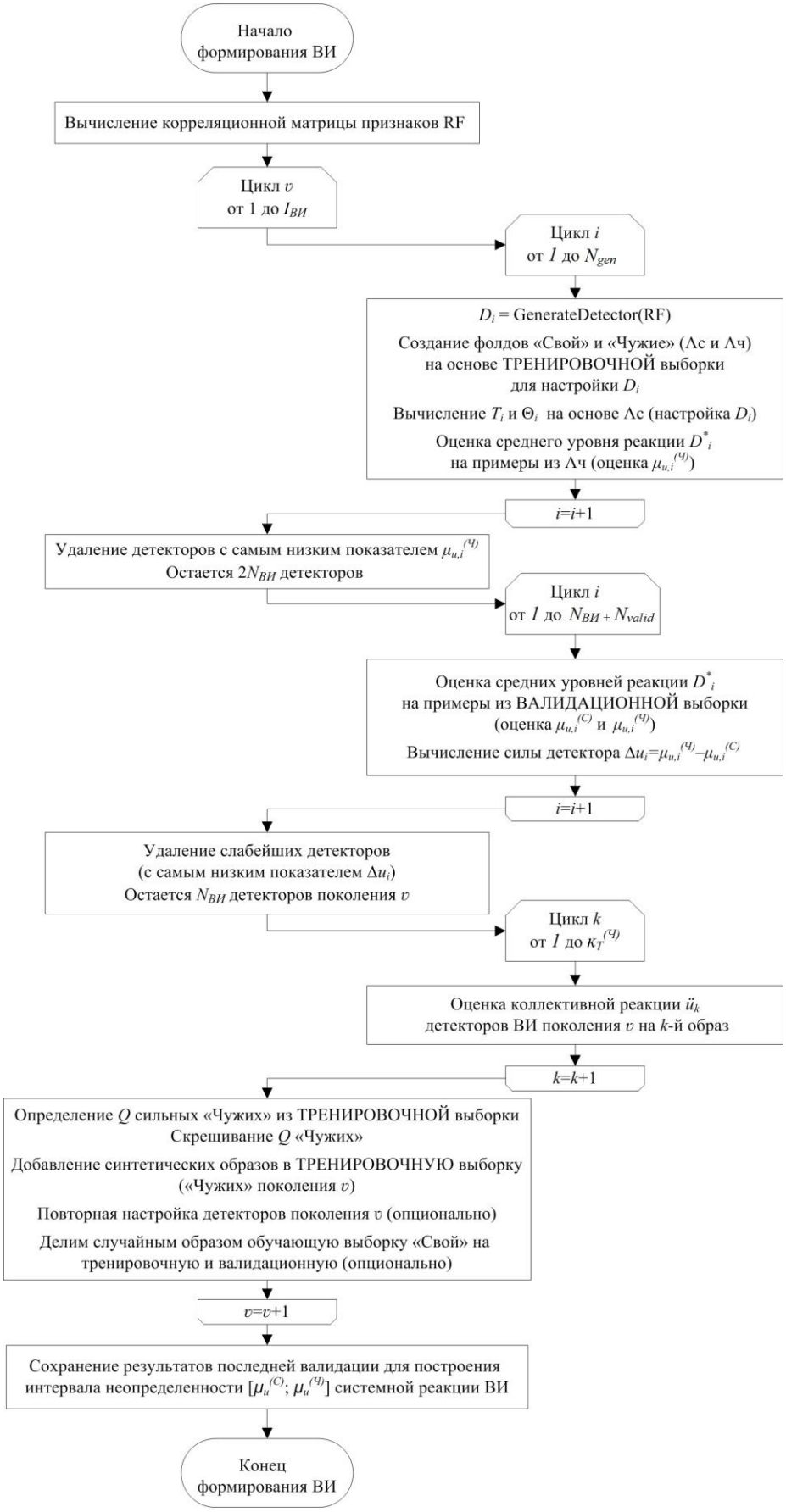


Рисунок 3.10 – Алгоритм формирования ВИ

Также параметрами алгоритма являются объемы тренировочной ($K_G^{(T)}$ и $K_I^{(T)}$) и валидационной выборок ($K_G^{(V)}$ и $K_I^{(V)}$), размеры фолдов «Свой» и «Чужой» ($K_G^{(F)}$ и $K_I^{(F)}$). В настоящем исследовании размер фолдов определялся из соотношения $K_I^{(F)}=K_I^{(T)}/3$, $K_G^{(F)}=2\cdot K_G^{(T)}/3$. Объем тренировочной выборки «Чужие» и размер фолда «Чужие» не являются фиксированными, а увеличиваются с каждой итерацией обучения при добавлении в тренировочную выборку *синтетических примеров*. Валидационная выборка всегда остается неизменной.

3.7 Алгоритм онлайн-обучения адаптивной нейро-иммунной модели искусственного интеллекта с подкреплением

Если обучающая выборка нерепрезентативна, эффективность детекторов ВИ может не соответствовать оценке Δu (3.13), при этом нет гарантий, что плохо настроенные детекторы в действительности преодалевают барьер Кондорсе (для таких детекторов оценки на тестовой выборке должны принимать вид $\mu_{u,i}^{(Ч)}<\mu_{u,i}^{(С)}$). Обойти барьер Кондорсе можно, если дать возможность детекторам ПИ голосовать за коллективное решение детекторов ВИ.

Введем следующее правило, основанное на ИНР: при $u_i>\mu_u^{(Ч)}$ или $u_i<\mu_{u,i}^{(С)}$ решение D^*_i считается определенным, а при $\mu_u^{(С)}<u_i<\mu_u^{(Ч)}$ решение D^*_i не определено. Если при распознавании образа решение детекторов ВИ считается неопределенным, то активируется механизм ПИ (рисунок 3.11).

Генерируются новые детекторы, которые настраиваются на других данных – примерах из валидационной выборки. Для новых детекторов вычисляются реакции u_i, но при формировании коллективного решения учитываются голоса только тех детекторов, которые дают определенный ответ (эти детекторы становятся клетками памяти), детекторы ПИ с неопределенным ответом уничтожаются.

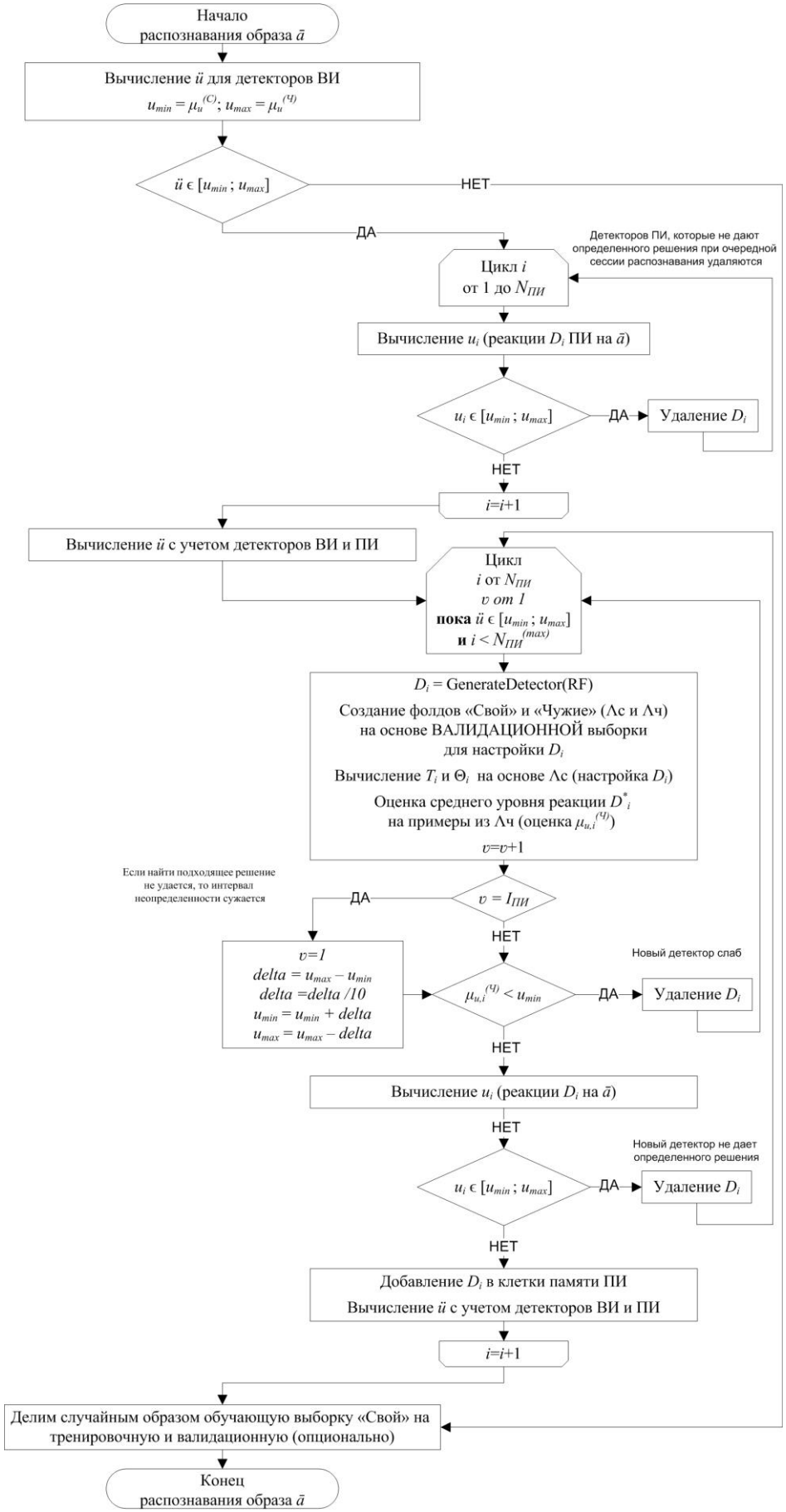


Рисунок 3.11 – Алгоритм работы ПИ

На скорость и эффективность алгоритма дообучения, представленного на рисунке 3.11, более всего влияют следующие основные параметры:

– $I_{ПИ}$ – количество итераций обучения;
– $N_{ПИ}^{(max)}$ – максимальное количество детекторов ПИ (тогда $N_{ПИ}$ – их фактическое количество).

Введение $I_{ПИ}$ позволяет избежать бесконечного цикла дообучения. Детекторы приобретенного иммунитета могут компенсировать недостаток априорных знаний о классах образов «Свой» и «Чужой». Описанный алгоритм реализует механизм обучения с подкреплением. Попадание u_i в интервал [$\mu_u^{(С)}$; $\mu_u^{(Ч)}$] является сигналом подкрепления – откликом среды на принятые решения.

3.8 Экспериментальная оценка надежности адаптивной нейро-иммунной модели искусственного интеллекта на примере задачи верификации образов клавиатурного почерка

Экспериментальная проверка адекватности предложенной модели и алгоритмов ее обучения выполнялась на примере задачи классификации образов клавиатурного почерка. Данная задача актуальна в контексте проблем информационной безопасности и относится к трудноразрешимым.

Вероятность «ложного допуска Чужого» (FAR) должна быть ничтожно мала, а вероятность «ложного отказа Своему» (FRR) может быть выше (приемлемый уровень – 10-25% ошибок). Показатели FRR и FAR взаимозависимы, их соотношение определяет пороговый коэффициент, с помощью которого их можно балансировать. Сравнение методов распознавания иногда осуществляется по средней точности (Mean Accuracy, далее MAC=1–(FRR+FAR)/2) – соотношению верных решений и количества опытов. Однако чаще используется коэффициент равной вероятности ошибок EER, когда выполняется условие: EER≈FAR≈FRR≈1-MAC. Для этого при тестировании строятся характеристические кривые (ROC-кривые), отображающие взаимную

зависимость FRR, FAR и порогового коэффициента. Если MAC принимает низкие значения, например, менее 0,9 (или EER>0,1), то настроив систему на FAR=0,0001, мы можем получить 1≥FRR>0,5 (более 50% «ложных отказов»). Все зависит от характера ROC-кривых.

Достигнутые ранее результаты (таблица 3.3) говорят о том, что найти решение этой задачи затруднительно, в том числе в нейросетевом логическом базисе. Для использования потенциала методов «глубокого» обучения требуется большой объем обучающей выборки, нереализуемый на практике (320 примеров на человека). Рассматриваемая научная задача хорошо иллюстрирует ограниченность применимости классических архитектур многослойных нейронных сетей.

Для проверки эффективности предложенных модели и алгоритмов использовалось 3 базы клавиатурного почерка: из работ [262] (Б1), [295] (Б2) (таблица 3.3) и собственная база (Б3). Последняя включает 32 человека, каждый из которых 50 раз ввел на клавиатуре фразу «система защиты должна постоянно совершенствоваться» (n=63, учтены только попытки безошибочного ввода).

Опыты проводились при различном объеме обучающей выборки примеров «Свой»: от K_G=20 до K_G=40. Тренировочная и валидационная выборки примеров «Свой» перед обучением делились в соотношении: $K_G^{(T)}=2\cdot K_G^{(V)}$. Тренировочная и валидационная выборки примеров «Чужих» включали по одному примеру от каждого испытуемого из набора данных, кроме примеров «Свой». Остальные примеры использовались в качестве тестовой выборки. Тестирование проводилось методом перекрёстного сравнения (при расчете FAR образ каждого субъекта сравнивался с эталонами всех остальных субъектов). Таким образом, для каждого испытуемого объем тестовой выборки «Чужих» варьировался в зависимости от набора данных так: Б1 – 19900, Б2 – 1968, Б3 – 1488. Показатели FRR и FAR вычислялись как отношение числа ошибок «ложного отказа» или «ложного доступа» к числу соответствующих опытов. Результаты представлены на рисунке 3.12 и в таблице 3.4.

Таблица 3.3. Достигнутые показатели надежности аутентификации по клавиатурному почерку

Сведения о методе классификации	K_G	Описание базы данных образов	EER	MAC
			в долях / вероятностях	
Рекуррентные ИНС (2 LSTM блока, оптимизатор Adam) [265]		**Б1 (n=31)**: 51 испытуемый (по 400 примеров ввода пароля ".tie5Roanl" на клавиатуре, всего 20400 образов), данные получены за 6 месяцев (испытуемые вводили по 50 примеров через определенный период времени) [262]	0,227	0,773
Рекуррентные ИНС (3 GRU блока, оптимизатор Adam) [265]			0,15	0,85
Рекуррентные ИНС (3 LSTM блока, оптимизатор Adam) [265]			0,219	0,781
Малые и «глубокие» сверточные ИНС с предобучением и без [239]	200-300 примеров			**≤ 0,925**
«Манхэттенская» масштабируемая метрика [262]	200 примеров		**0,096**	**0,904**
Меры Махаланобиса и ближайшего соседа [262]	200 примеров		0,10	0,9
Статистическая техника (на базе z-оценки) [262]	200 примеров		0,102	0,898
Машина опорных векторов (SVM) [262]	200 примеров		0,102	0,898
Автоассоциативный многослойный персептрон [262]	200 примеров		0,161	0,839
Мера Евклида [262]	200 примеров		0,215	0,785
Нечеткая логика [262]	200 примеров		0,221	0,779
Метод k-ближайших соседей (k-NN) [262]	200 примеров		0,372	0,628
«Глубокая» ИНС, оптимизатор Nadam [295]	320 примеров			**0,92**
SVM [295]	320 примеров			0,7115
Наивный Байес [183]	34 примера	**Б2 (n=71)**: 42 испытуемых (по 51 примеру ввода ".tie5Roanl" на планшете или смартфоне), данные временных задержек, силы нажатия, площади касания [183]		0,7893
Сети Байеса [183]	34 примера			0,9194
C4.5 (J48) [183]	34 примера			0,6902
k-NN [183]	34 примера			0,7298
SVM [183]	34 примера			0,8833
Случайный лес [183]	34 примера			**0,9304**
Многослойный персептрон [183]	34 примера			0,8626
Мера Евклида [183]	34 примера		0,157	0,843
«Манхэттенская» метрика [183]	34 примера		**0,129**	**0,871**
Мера Махаланобиса [183]	34 примера		0,166	0,834

Рисунок 3.13 иллюстрирует, как определить EER, а также что FRR и FAR можно балансировать, например FRR=0,193 при FAR=0,0001 (рисунок 3.13б).

Таблица 3.4. Сводная таблица основных показателей надежности в зависимости от объема обучающей выборки K_G и от использования ПИ, при $I_{ПИ}$=5, Q=4

База	K_G	$I_{ВИ}$	$N_{ВИ}$	$N_{ПИ}^{(max)}$	EER
Б1	25	1500	50	0	0,1167
Б1	25	1500	50	50	0,1028
Б1	40	1050	50	0	0,0821
Б1	40	1050	50	25	0,0798
Б2	27	1500	50	0	0,058
Б2	27	1500	50	25	0,0612
Б3	27	1500	50	0	0,0284
Б3	27	1500	50	25	0,0265

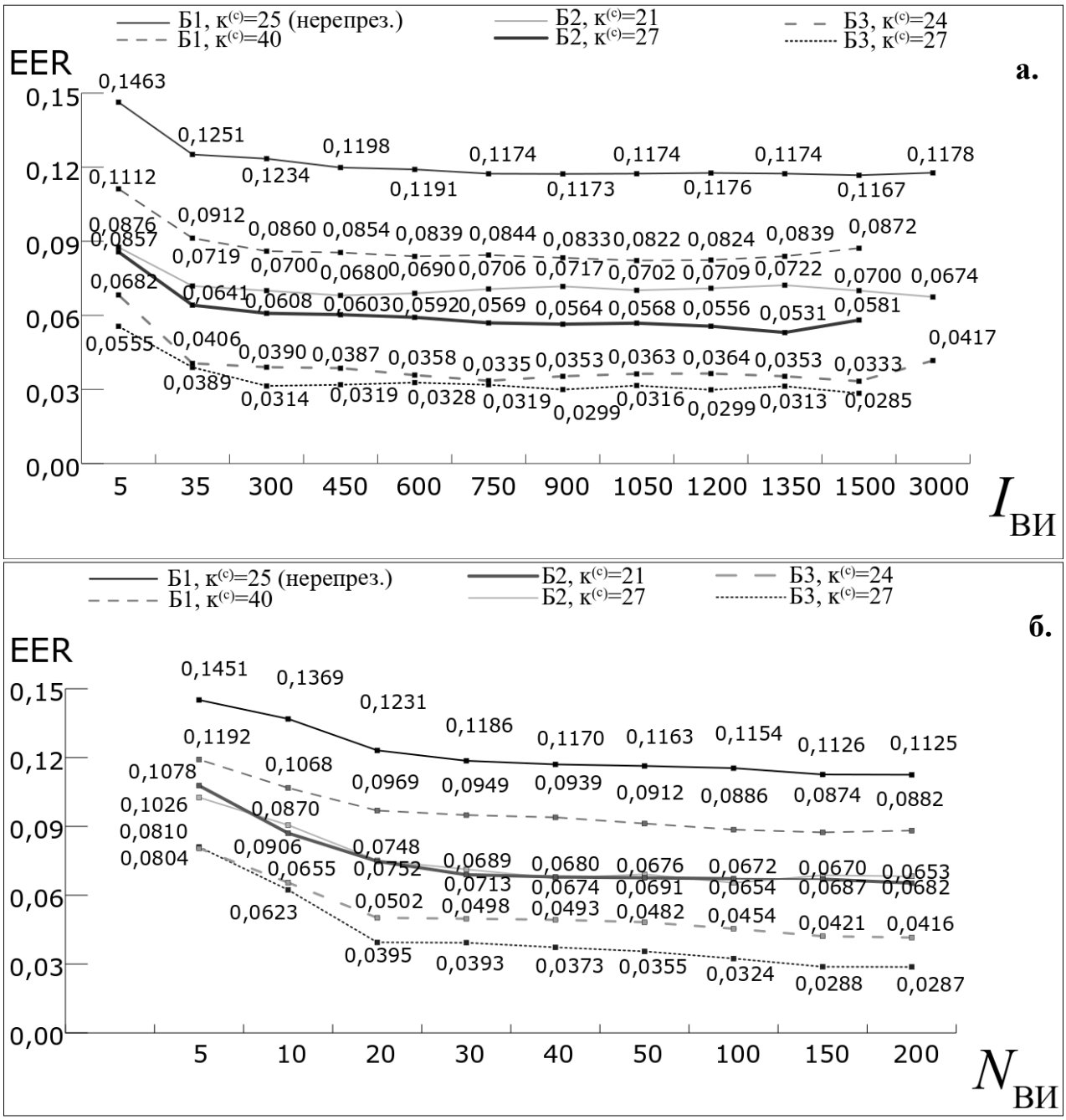


Рисунок 3.12 – ROC-кривые, демонстрирующие результаты эксперимента при использовании только врожденного иммунитета ($N_{ПИ}^{(max)}$=0) при Q=4: а. $N_{ВИ}$=50, б. $I_{ВИ}$=50

Надежность решений зависит от того, как была составлена обучающая выборка. Это наглядно видно при тестировании на базе Б1 [265], которая отсортирована по временным меткам. Если выбирать обучающие примеры равномерно (как в [295], где MAC=92,6), то обучающая выборка будет репрезентативной, но если обучать и тестировать систему на образах испытуемого, которые были введены в разные дни (с большим перерывом), то

репрезентативность выборки окажется низкой. В этом случае наблюдается более существенная разница в надежности решений для режима ВИ (когда используются детекторы только врожденного иммунитета, т.е. $N_{ПИ}^{(max)}=0$) и ВИ+ПИ (когда используются детекторы врожденного и приобретенного иммунитета, т.е. $N_{ПИ}^{(max)}>0$).

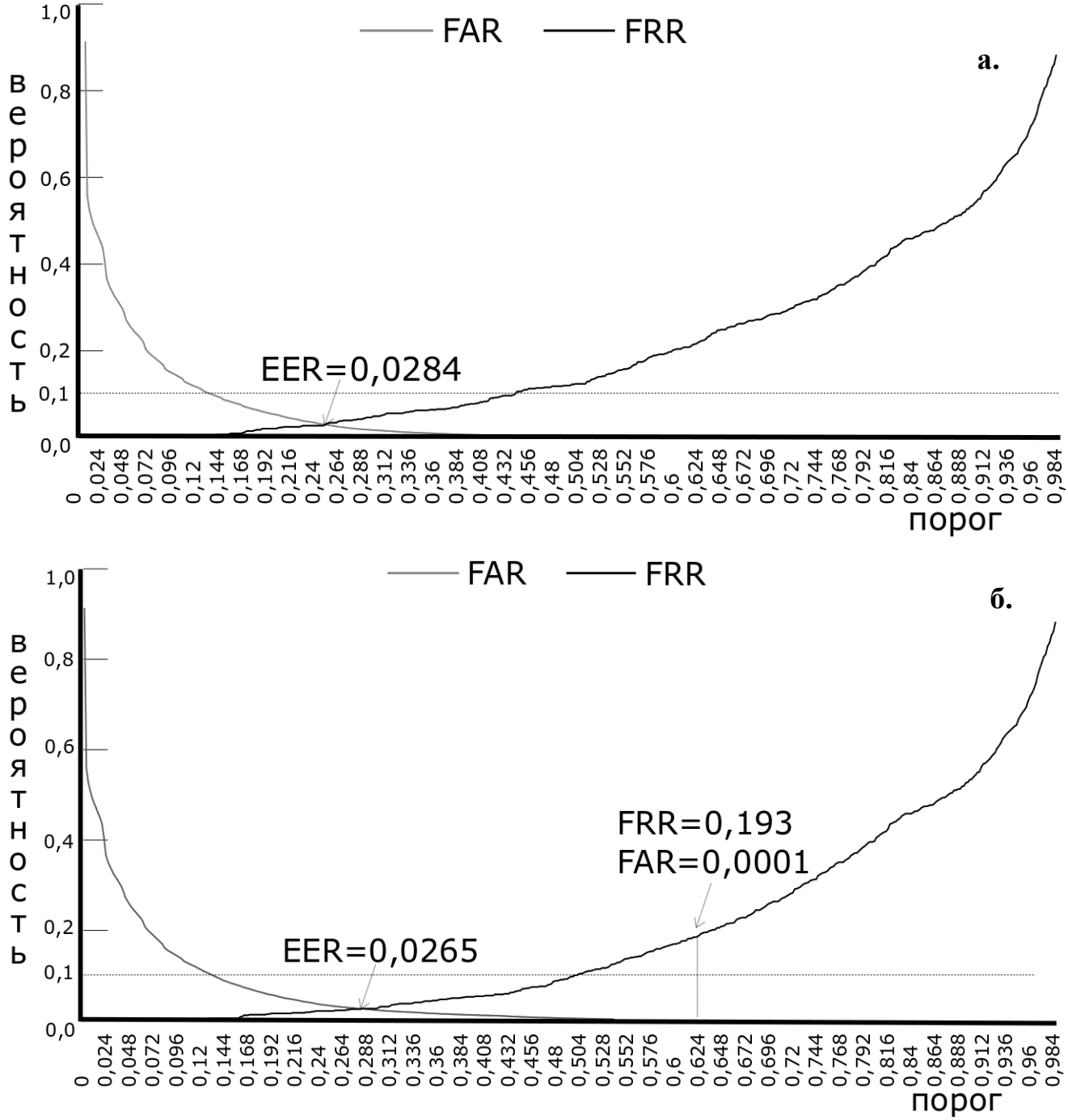


Рисунок 3.13 – ROC-кривые, демонстрирующие результаты эксперимента для Б3, при Q=4, $I_{ВИ}$=1500, $к^{(C)}$=27: а. ВИ ($N_{ВИ}$=50, $N_{ПИ}^{(max)}$=0), б. ВИ + ПИ ($N_{ВИ}$=50, $N_{ПИ}^{(max)}$=25)

Чем дольше обучается модель, тем надежнее ее решения, но чем лучше обучена нейро-иммунная модель, тем больше времени требуется, чтобы поднять надежность еще выше (рисунок 3.12а). Процесс обучения является достаточно устойчивым, но при высоких значениях $I_{ВИ}$ все-таки наблюдается незначительная склонность к переобучению. При увеличении $N_{ВИ}$ (рисунок 3.12б) показатель EER монотонно снижается, темп снижения постепенно ослабляется и почти полностью останавливается, когда решения детекторов ВИ становятся сильно зависимыми (что неизбежно при росте их количества).

На данном этапе можно утверждать, что разработанная модель и алгоритмы ее обучения удовлетворяют основным принципам построения ИИС [192], которые применительно к настоящей работе можно переформулировать так:

1. Распределенный характер вычислений и проявление эмерджентности (нейро-иммунная модель может повышать качество решений в процессе функционирования, в отличие от ее базовых узлов – детекторов, точность распознавания для нейро-иммунной модели выше, чем для каждого детектора в отдельности).

2. Достаточно устойчивый процесс обучения (склонность к переобучению незначительна при формировании ВИ).

3. Способность нейро-иммунной модели к адаптации, обусловленная двойной пластичностью: структурной и параметрической (меняются параметры и состав детекторов).

4. Взаимодействие – врожденный иммунитет формирует параметры, которые влияют на механизм подкрепления детекторов приобретенного иммунитета.

5. Надежность решений нейро-иммунной модели зависит от объема и чувствительности популяции детекторов.

6. Формирование памяти при помощи механизма приобретенного иммунитета.

3.9 Защищенные нейро-иммунные контейнеры

С помощью спецификации [162] могут быть созданы защищенные нейросетевые биометрические контейнеры – нейросетевые биометрические контейнеры, в которых некоторые части скрыты от непосредственного изучения путем использования обратимого или необратимого преобразования (в соответствии с ГОСТ Р 52633.5-2011). Обычный нейросетевой биометрический контейнер – это структурированный блок данных, содержащий параметры обученного НПБК.

Принцип защиты из [162] состоит в следующем. Нейроны выстраиваются в цепочке путём создания перекрёстных связей (рисунок 3.14). После обучения таблицы каждого нейрона шифруются наложением гаммы, представляющей собой контрольную сумму выходов всех предыдущих нейронов в цепочке:

$$tables_l' = XOR(tables_l, hash(pass, b_1, ..., b_{l-1}))$$

где l – номер нейрона в цепочке, $hash()$ – криптографическая хеш-функция, $pass$ – пароль, который является опциональным и служит как дополнительный фактор защиты. Первый нейрон остаётся незащищённым, однако все классические нейроны имеют «встроенную» защиту в виде весов, из которых нельзя прямым численным методом восстановить данные обучающей выборки. Поэтому параметры b_l также можно считать неизвестными для злоумышленника, как и pass. Такой метод применяется для повышения энтропии откликов НПБК от образы «Чужих» для защиты от атак «извлечения знаний».

Нейро-иммунный контейнер может компрометировать знания модели ИИ так как большинство детекторов не скрывают параметров распределения признаков, в отличие от корреляционных нейронов. Для применения аналогичного принципа защиты по отношению к параметрам обученной адаптивной нейро-иммунной модели следует комплексировать ее с классическим НПБК, обученным по ГОСТ Р 52633.-2011, или с НПБК на базе корреляционных нейронов. Далее следует выстроить нейроны и детекторы в цепочке.

Принципиальным является то, что классические или корреляционные нейроны следует размещать в начале цепочки, а детекторы – в конце, так как они компрометируют эталон и ключ (пароль) пользователя (рисунок 3.14). Этот вопрос прорабатывался в [81, 146, 147, 154].

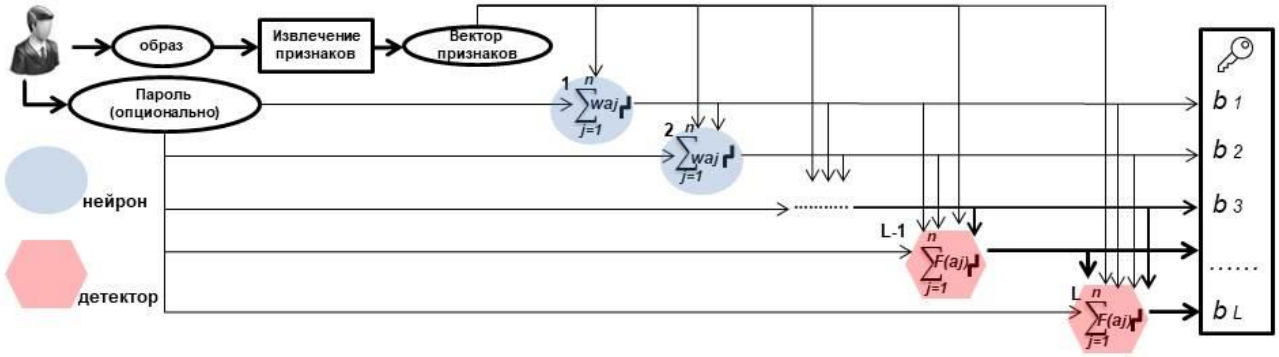


Рисунок 3.14 – Механизм защиты нейро-иммунных контейнеров

Следует отметить, что после применения данного принципа защиты нейро-иммунная модель ИИ потеряет способность к онлайн-обучению. Поэтому такой принцип защиты нужно применять, если риск дрейфа данных или концепций минимален, а риск извлечения знаний высок. Чтобы сохранить возможность дообучения следует разработать иной метод комплексирования НПБК и адаптивной нейро-иммунной модели ИИ (что будет предложено в следующей главе).

3.10 Анализ результатов. Выводы

Предложены адаптивная нейро-иммунная модель ИИ и алгоритмы ее обучения с учителем и с подкреплением. Предложенные решения основаны на применении методов математической статистики, ансамблей классификаторов, концепции искривления пространства признаков, корреляционных метрик и нейронов Байеса-Минковского, а также аналогий с теориями о естественной иммунной системе. В настоящей работе даны достоверные свидетельства, доводы и результаты эмпирических исследований, которые говорят об их высокой

эффективности. В задаче аутентификации по клавиатурному почерку разработанная модель превосходит многослойные ИНС (и другие рассмотренные в работах [183, 239, 262, 265, 295] методы) либо дает сопоставимые с ними показатели ошибок распознавания при гораздо меньшем объеме обучающей выборки (в разы). При этом предложенную модель легко обучить, достаточно лишь указать параметры $N_{ВИ}$ и $N_{ПИ}$, напрямую влияющие на объем сети и надежность ее решений, а также $I_{ВИ}$ и $I_{ПИ}$, напрямую влияющие на длительность обучения, время принятия решений и их надежность (обучение является устойчивым независимо от данных параметров).

Хочется подчеркнуть, что в настоящей работе не утверждается превосходство предложенного аппарата над нейронными сетями в общем случае. Это разные инструменты для решения разных задач. Их также можно использовать совместно. Например, извлечение признаков может выполняться при помощи глубокой сверточной нейронной сети (предварительно обученной «сжимать» образ до определенного количества признаков), а распознавание образов – с помощью предложенной адаптивной нейро-иммунной модели ИИ.

Мощность предложенной модели ИИС связана с количеством вариаций мер близости, используемых в основе детекторов. Детектор – это аналог нейрона (либо искусственной иммунной клетки в терминах ИИС), с несколько модифицированной структурой. В разработанной модели имеется незначительная склонность к переобучению при формировании ВИ (исправить этот недостаток видится возможным, если контролировать коррелированность решений детекторов непосредственно либо изменять гиперпараметры ИИС в зависимости от ИНР).

Вопросы синтеза, обучения и защиты нейро-иммунных моделей, а также выбора функционалов для построения детекторов также прорабатвались в [52, 75, 81, 118, 139, 142, 143, 154, 282, 310, 329, 336].

4 Высоконадежная многофакторная биометрическая аутентификация на основе тайных биометрических образов

Данная глава посвящена разработке методов аутентификации на основе тайных биометрических образов. Прежде всего, следует отметить, что открытые биометрические образы (отпечаток пальца, радужка, лицо) хотя и являются достаточно уникальными для высоконадежной биометрической аутентификации, уязвимы перед атаками представления (спуфингом). Злоумышленник может снять эти характеристики бесконтактно или скрыто от владельца (например, с ручки двери, фотографии). Процесс ввода биометрического образа можно непосредственно контролировать только по месту прохождения аутентификации (эти вопросы рассматриваются в серии из 4-х стандартов ISO/IEC 30107). Для удаленной аутентификации аналогичные методы контроля не действуют – биометрические данные могут быть синтезированы искусственно. Таким образом, при удаленной аутентификации открытую биометрическую характеристику стоит рассматривать не в качестве подтверждающей информации, а только как идентификатор. Например, лицо субъекта можно использовать для скрытой идентификации [28] (даже в условиях масочного режима [9]) или при расследовании инцидентов (когда нужно опознать личность на видео).

К открытым образам также относится рукописная подпись (автограф). Вероятность ложного принятия подделки подписи в десятки раз выше, чем случайного совпадения [237, 324]. Текстонезависимое распознавание диктора тоже строится на открытых образах. Последние достижения в области синтеза речи показали, что можно сгенерировать речевой сигнал конкретного человека произвольного содержания, почти идентичный натуральному голосу, предварительно обучив нейросетевой синтезатор на нескольких часах голосовой информации (со стенограммой) за ночь на обычном компьютере [339]. Созданы архитектуры синтезаторов речи (на сверточных или рекуррентных сетях) – WaveNet, Char2Wav, DeepVoice, Tacotron.

Поэтому для аутентификации нужно использовать тайный образ, отражающий особенности воспроизведения пароля его владельцем. У злоумышленника не должно быть информации о том, какой биометрический пароль синтезировать. В настоятоящем исследовании использовались следующие типы образов (модальности):

- рукописный пароль, который в отличие от подписи (автографа) можно держать в секрете и изменить в любой момент, однако недостатком рукописного пароля является более низкая стабильность его воспроизведения по сравнению с подписью;
- голосовой пароль, который может быть также одноразовым (система может случайным образом выбирать парольные фразы из словаря, который был заранее заготовлен при ее обучении);
- данные о внутреннем строении наружного уха, получаемые при помощи эхографии. Индивидуальные особенности ушного канала субъектов скрыты от непосредственного наблюдения и не могут быть скопированы путем фотографирования. «Плоское» изображение уха недостаточно информативно для изготовления «муляжа».

Выбор данных типов биометрических образов также обусловлен большим количеством проведенных экспериментов по анализу биометрических данных различной природы и распознаванию личности субъектов (рукописные образы [35, 53, 74, 76, 86, 92-94, 100, 107, 113, 126, 144, 146, 147, 150, 153, 171, 248, 280, 281, 291, 308], клавиатурный почерк и/или особенности работы субъектов с манипулятором «мышь» [3, 13, 25, 54, 72, 102, 103, 111, 112, 124, 139, 142, 143, 145, 149, 156, 157, 160, 290, 335], голос [23, 36, 37, 147, 148], лицо [3, 35, 75, 92], термограммы лица [122, 337], ЭЭГ [2, 78, 104, 110, 123, 149, 193], геометрия ушной раковины [34, 152, 167, 333]). Существенная часть результатов этих экспериментов также представлена в краткой форме в Приложении А.

Помимо аспектов, связанных с регистрацией и анализом биометрических образов в настоящей главе рассматриваются вопросы комплексирования этих образов, а также совместного использования разработанных и описанных в

предыдущих главах моделей ИИ и алгоритмов их обучения. В настоящей главе предложено и описано:

- метод биометрической аутентификации по особенностям внутреннего строения наружного уха (по акустическим свойствам ушного канала) с защитой биометрических данных от компрометации;
- метод биометрической двухфакторной аутентификации по рукописным и голосовым паролям, устойчивый к концептуальному дрейфу биометрических данных;
- комплексный алгоритм высоконадежной многофакторной аутентификации по особенностям внутреннего строения наружного уха, голосовому и рукописному паролям, основанному на использовании разработанных моделей ИИ и алгоритмов их обучения.

Начнем описание представленных результатов с вопросов комплексирования биометрических образов и предложенного алгоритма. Далее перейдем к описанию методов аутентификации, которые лежат в основе алгоритма и результатам проведенных экспериментов.

4.1 Комплексирование независимых биометрических образов и моделей искусственного интеллекта

На данный момент не утверждено единого стандарта для безопасного объединения нескольких разнородных биометрических образов при защищенном исполнении процедур аутентификации (проект ГОСТ Р 52633.7 «Защита информации. Техника защиты информации. Высоконадежная мультибиометрическая аутентификация» находится на публичном обсуждении). Существующий стандарт ГОСТ Р 54411-2011 «Мультимодальные и другие мультибиометрические технологии» не рассматривает вопросы защищенного исполнения процедур биометрической аутентификации, а только варианты объединения классификационных решений и биометрических образов при их

последовательном или одновременном представлении. В настоящей работе предлагается иной вариант комплексирования биометрических образов.

Представлены комплексный метод и алгоритм трехфакторной высоконадежной аутентификации с последовательным предоставлением образов (рисунок 4.1). Первый фактор аутентификации – это образ уха. Для защиты данных уха от компрометации при хранении и передачи по каналам связи используется разработанная модель НПБК на базе корреляционных нейронов. Второй и третий факторы могут быть открытыми (рукописная подпись, фиксированная фраза) или тайными (рукописный и голосовой пароли), что рекомендуется. Для этих типов биометрических данных важно поддерживать актуальность, так как они изменчивы, поэтому для их анализа применяется адаптивная нейро-иммунная модель ИИ. Для защиты знаний адаптивной модели ИИ ее параметры после обучения шифруются на ключе, формируемом НПБК. Небольшое число ошибочных бит в ключе, генерируемом НПБК, может быть скорректировано за счет использования алгоритмов помехоустойчивого кодирования и кодов, исправляющих ошибки. Это позволяет балансировать показатели FRR и FAR на выходе НПБК. Если число ошибочных бит в ключе больше исправляющей способности кода (когда на вход НПБК поступает образ «Чужой»), то параметры адаптивной модели ИИ дешифруются неверно и доступ отклоняется, иначе исполняется алгоритм классификации и онлайн-обучения, который иллюстрируется на рисунке 3.11.

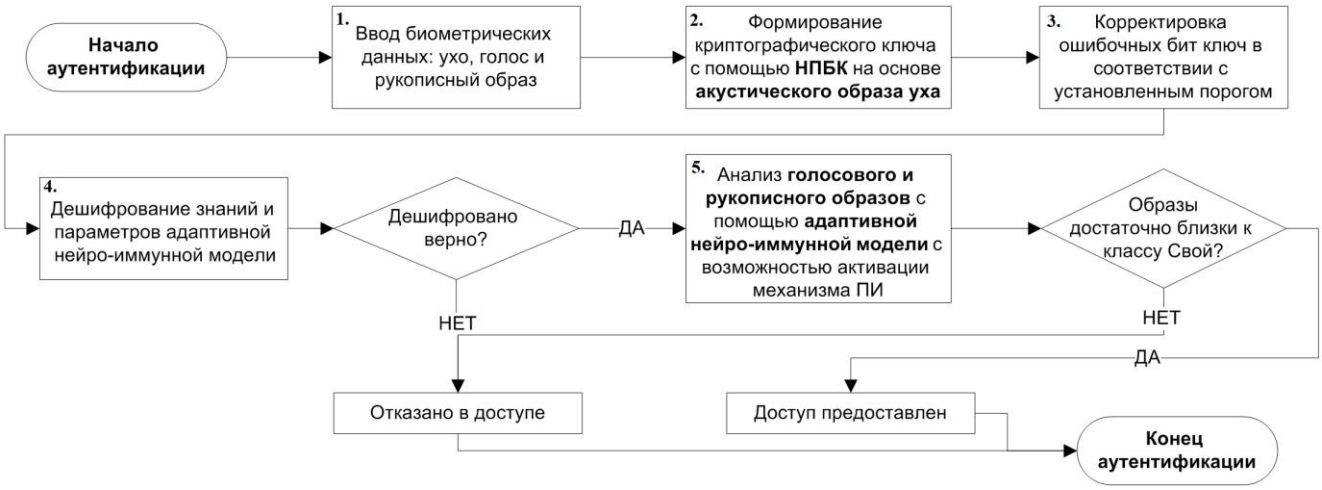


Рисунок 4.1 – Алгоритм трехфакторной аутентификации в защищенном режиме

Блок 3 может быть реализован на основе специальных помехоустойчивых кодов, предложенных для биометрии Безяевым А.В. Корректирующие коды из работы [11] позволяют безопасно хранить синдромы ошибок в виде усеченных хеш-функций и не накладывают избыточности на ключ, извлекаемый с помощью НПБК в отличие от нечеткого экстрактора. Классическая схема нечеткого экстрактора, опирающаяся на традиционные алгоритмы помехоустойчивого кодирования (Адарама, БЧХ, Рида-Соломона), подразумевает объединение биометрических данных, криптографического ключа (или пароля) пользователя, на который накладывается помехоустойчивый код, и синдромов ошибок в один контейнер – открытую строку (secure hash). Коды Безяева А.В. позволяют отделить синдромы ошибок и хранить их отдельно, не компрометируя ключ (пароль). Этим вопросам посвящены многочисленные исследования [97]. Таким образом, коды Безяева А.В. позволяют не создавать открытую строку, а просто корректировать ключ, генерируемый на выходе НПБК с помощью заранее сформированных на этапе обучения синдромов ошибок.

Дешифрование знаний и параметров адаптивной модели ИИ (блок 4) может выполняться любым современным алгоритмом. Требования к алгоритму шифрования должны определяться криптографами. Разработка стандартов по криптографической защите находится в компетенции ТК 26 «Криптографическая защита информации».

Сущностная часть блоков 1, 2, 5 определяется предложенными методами аутентификации, описанными далее, которые могут на практике применяться как отдельно, так и в составе алгоритма многофакторной аутентификации, представленного на рисунке 4.1. При аутентификации только по голосовому и рукописному паролю биометрические данные пользователей могут быть скомпрометированы при хранении в базе знаний, так как НПБК не будет использоваться для дешифрования параметров адаптивной модели. В случае если задачи защиты биометрических данных на практике не стоит (что может быть

обусловлено особенностями архитектуры или назначением системы аутентификации), тогда голосовой и рукописный образы могут быть открытыми.

Алгоритм, представленный на рисунке 4.1, можно считать алгоритмом высоконадежной многофакторной аутентификации, если вероятность ошибок второго рода будет не выше 10^{-12} (в соответствии с ГОСТ Р 52633.5-2011), что будет показано далее в настоящей главе.

Рассмотрим подробнее предложенные методы биометрической аутентификации, лежащие в основе представленного алгоритма.

Начнем описание представленных результатов с вопросов комплексирования биометрических образов и предложенного алгоритма. Далее перейдем к описанию методов аутентификации, которые лежат в основе алгоритма и результатам проведенных экспериментов.

4.2 Методы распознавания личности на основе анализа оптических образов наружного уха

Наружное ухо состоит из ушной раковины и наружного слухового прохода, называемого также слуховым (ушным) каналом, отделяемого от среднего уха барабанной перепонкой. Строение человеческого уха закладывается в детстве (до 8 лет), во взрослом возрасте значительных изменений в пропорциях ушных раковины и канала не происходит [314]. Исследования показывают [182], что структура ушной раковины, длина и форма ушного канала (рисунок 4.2) очень различаются у людей.

Оптические образы уха представляют собой данные, получаемые путем измерения параметров уха оптическими методами. Обычно считывающим устройством является фото/видео камера или 3D-сканер [355]. На этапе предварительной обработки 2D и 3D образа уха часто происходит обнаружение так называемой области интереса (Region-of-interest, ROI) – фрагмента, который непосредственно содержит основную информацию о субъекте. Эта процедура

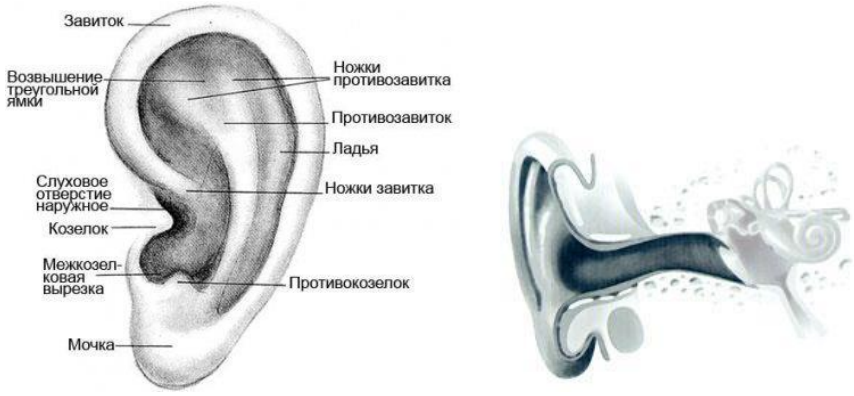
является одной из наиболее важных в общем конвейере обработки, и значительно влияет на производительность [206]. 2D изображение может содержать не только образы «чистого» уха, но и другие элементы (например, профиль лица, посторонних людей). Также возможна частичная окклюзия (сокрытие) ушей, например, посредством украшений, очков или волос. В связи с этим задача обнаружения уха на входном изображении становится затруднительна.

Можно выделить следующие подходы к распознаванию ушной раковины на 2D изображениях [225]:

1) *геометрический*. Методы из этой категории основаны на обнаружении границ внешнего уха. Как правило, информация о границах используется для описания геометрических свойств ушей или получения статистических данных, связанных с геометрией.
2) *целостный (структурный)*. Вычисляются некоторые характеристики изображения уха, такие как коэффициенты Фурье, цветовые градиенты и т.д. Поскольку внешний вид уха значительно меняется с изменением позы или освещения, необходимо применять методы нормализации для коррекции этих изменений при извлечении признаков;
3) *локальный* (чаще всего применяется при частичной окклюзии (сокрытии) ушей). Получение описания локальной окрестности вокруг уха на изображении. Точки, представляющие интерес, не обязательно должны соответствовать конструктивно значимым частям уха.

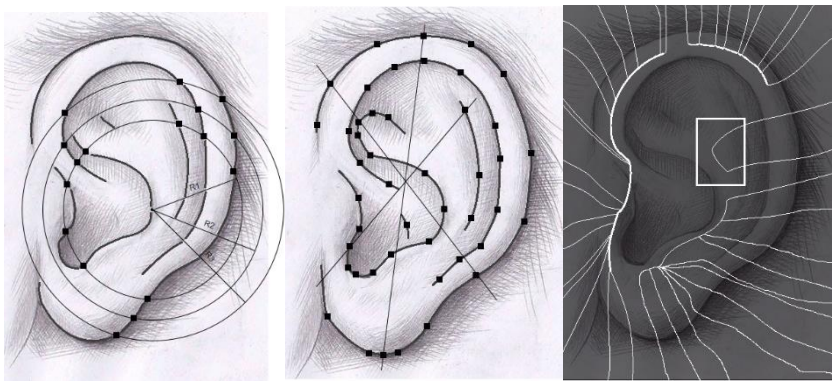
В качестве признаков (согласно системе Янарелли [245]) обычно выступают характерные точки структуры уха. Внешний вид наружного уха определяется формами завитка, противозавитка, козелка, противокозелка и других важных структурных частей (рисунок 4.2). Представление ушной раковины в зависимости от метода извлечения признаков может отличаться (рисунок 4.3). В научных работах находят применение следующие методы: концентрических окружностей (Concentric Circle Model, CCM) [203], активного контура (Active Contour Model, ACM) [257], силовых полей (Force Field Feature Extraction, FFFE) [244], главных компонент (Principal Component Analysis, PCA), масштабно-инвариантной

трансформации признаков (Scale-Invariant Feature Transform, SIFT) [187], итеративный алгоритм ближайших точек и другие. В работе [359] для извлечения пространственных признаков различных направлений и масштабов применяется фильтр Габора (*GaborFilter*). Признаки, извлечённые фильтром, имеют высокую размерность, для снижения размерности применялся линейный дискриминантный анализ Фишера (Fisher's Linear Discriminant Analysis, FLDA).



а) б)

Рисунок 4.2 – Наружное ухо: (а) структура ушной раковины, (б) ушной канал



а) б) в)

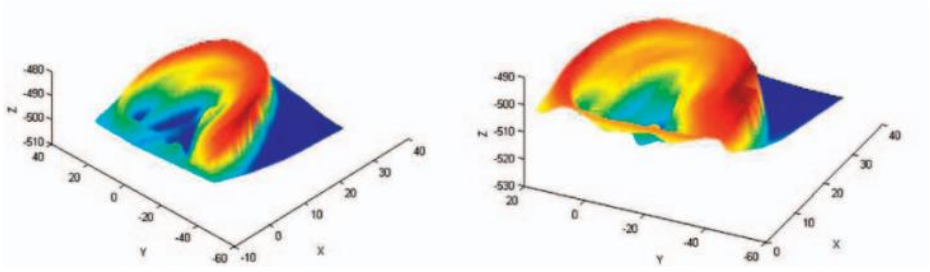
Рисунок 4.3 – Примеры извлечения признаков разными методами: а) метод концентрических окружностей; б) метод активного контура; в) метод силовых полей

В работе [251] предложили метод автоматического обнаружения уха на основе алгоритма AdaBoost. В работе «слабые» классификаторы построены на использовании примитивов Хаара, позволяющие обнаружить вертикальные и горизонтальные края уха. Алгоритм использует обучение с учителем с помощью метода обертывания. Он выбирает лучший слабый классификатор с учетом заданной взвешенной ошибки входных выборок на каждой итерации. По

результатам эксперимента по распознаванию 203 изображений ушей без окклюзий метод обеспечил правильное обнаружение во всех случаях. При тестировании метода в условиях окклюзии из 104 изображений, не участвовавших в обучающей выборке, удалось верно распознать только 54 примера. В работах [251, 358] исследователи также использовали *AdaBoost* для автоматического обнаружения уха на изображении.

Трёхмерные характеристики уха являются более информативными и могут предоставить более подробную информацию о глубине анатомической структуры уха [309]. Построение трехмерной модели уха может осуществляться двумя способами. Первый способ использует видеопоток как источник входных данных. Последовательная смена видеоизображений подразумевает изменение положения уха, а вместе с тем изменяется его ракурс, на основе которого получают трёхмерную модель. Второй способ предполагает использование 3D-сканеров. Вычисление объема и формы объекта базируется либо на определении расстояния до него по времени возврата отражённых от его поверхности лучей, либо на стереоскопии (такие камеры имеют несколько объективов).

При 3D распознавании анализируются аналогичные элементы уха [201], но при этом построение высокоточной 3D-модели ушной раковины дают гораздо больше информации. На рисунке 4.4 показан результат сканирования поверхности ушной раковины 3D-сканером. Преимущество такого подхода в том, что трёхмерная модель нечувствительна к освещению, а также к развороту и изменению ракурса (модель всегда можно повернуть в пространстве).



а) б)

Рисунок 4.4 – Пример 3D-изображения. На рисунках (а) и (б) показано два изображения одного уха, снятых с разных ракурсов (единицы измерения X, Y и Z – мм)

В исследовании [362] авторы предлагают гибридную систему распознавания ушной раковины на основе как локальных, так целостных характеристик. Для обнаружения уха использовался алгоритм быстрых свёрточных сетей Fast R-CNN (Faster Region-based Convolutional Neural Networks) [228]. В данном алгоритме детектирование ушей происходит в два этапа. Первый этап основан на так называемой сети предложения регионов (Region Proposal Network, RPN), которая при помощи выборочного поиска (selective search) выделяет регионы, в которых может находиться ухо. Второй этап предполагает применение свёрточной нейронной сети для поиска ушной раковины в предложенных регионах. Извлечение локальных признаков в этой работе происходит с помощью дескриптора 3D-формы (Surface Patch Histogram Of Indexed Shapes, SPHIS). Целостные признаки извлекаются методом, предложенном в [361].

4.3 Методы распознавания личности на основе анализа акустических образов наружного уха

В отличие от оптических методов (2D, 3D), акустический метод требует прямого контакта для извлечения характеристик уха.

Ушная раковина и слуховой канал являются резонансными системами. Чтобы получить информацию о внутреннем строении наружного уха, можно воздействовать на ушной канал акустическими волнами, которые будут искажаться, отражаясь от стенок канала. Отраженный сигнал будет иметь отличия от исходного, обусловленные индивидуальными особенностями ушных раковины и канала человека. Параметры эхо-сигнала или его передаточной функции могут содержать информацию о геометрии слухового канала и ушной раковины, поэтому их можно воспринимать как вектор биометрических параметров (*признаков*), характеризующих строения наружного уха индивидуума.

Резонанс ушной раковины в среднем составляет около 5 кГц, а ушного канала – 2,5 кГц [182] (в грубом приближении это одномерная система, резонирующая на четверти длины акустической волны [182]). Человек способен слышать звуки в диапазоне от 16 Гц до 20 кГц, однако считается, что человеческое ухо в среднем наиболее чувствительно к акустическим колебаниям с частотами от 1 кГц до 5 кГц (сюда относятся звуки речи [298]). Нижние частоты (до 1000 Гц) при воздействии звукового сигнала на стенки канала, как правило, не вызывают резонанса, поэтому должны быть малоинформативны с точки зрения выявления отличительных особенностей его строения. Верхняя граница частоты зондирующего звукового сигнала ограничена возможностями системы воспроизведения и записи звука (высокие частоты имеет смысл анализировать, так как полезная информация также может присутствовать в обертонах).

Помимо геометрии наружного слухового канала на регистрируемый эхо-сигнал влияет отоакустическая эмиссия (ОАЭ) – очень слабые звуки (менее 20 дБ), регистрируемые в наружном слуховом проходе, но происходящие из улитки внутреннего уха как побочный продукт работы наружных волосковых клеток по усилению колебаний базилярной мембраны улитки. Различают спонтанную (самопроизвольно появляющуюся) и вызванную ОАЭ (возникает в ответ на предъявление в слуховой проход звукового стимула). Зарегистрировать ОАЭ можно с помощью высокочувствительного микрофона, в клинической практике возможность регистрации вызванной ОАЭ ограничена частотным диапазоном 0,5-8 кГц. ОАЭ «сливается» с отраженным сигналом, таким образом, являясь компонентом регистрируемого эхо-сигнала. При использовании громкого звукового стимула влияние индивидуальных особенностей ОАЭ на эхо-сигнал незначительно.

Принцип работы устройства для измерения акустических характеристик уха описан в работе [222]. Съём характеристик производится следующим образом: источник рядом с ушным каналом генерирует сигнал возбуждения, а приёмник измеряет отраженные отклики. В общем случае возбуждение может представлять собой акустический сигнал, который имеет довольно плоский частотный спектр.

Примерами устройств, измеряющих акустические характеристики ушного прохода (раковины), могут служить внутриканальные и накладные наушники либо мобильный телефон (при контакте с ухом).

Рассмотрим гипотетические факторы, способные повлиять на параметры эхо-сигнала и соотношение сигнал/шум. Источники искажений можно разделить на следующие категории: отологические (обусловленные анатомией или патологиями уха), технические (обусловленные особенностью используемого оборудования) и связанные с условиями использования (в том числе, «человеческий фактор»).

К отологическим факторам относятся:

– кондуктивная тугоухость – это нарушение слуха, при котором затруднено проведение звуковых волн по пути от наружного уха к внутреннему. Поражения наружного уха (новообразования, абсцесс) могут вызвать наибольшие затруднения с точки зрения использования предлагаемого метода, так как сигнал будет искажен из-за искривленной геометрии слухового канала. При поражениях барабанной перепонки или слуховых косточек пользователь может испытывать неприятные ощущения и боль во время прослушивания звукового сигнала. Эти поражения редко затрагивают оба уха сразу, поэтому субъект может использовать здоровое ухо при прохождении процедуры идентификации (что как будет показано далее, повышает количество ошибок распознавания примерно в 4 раза).

– нейросенсорная тугоухость – это потеря слуха, вызванная поражением структур внутреннего уха, преддверно-улиткового нерва или центральных отделов слухового анализатора. При повреждениях внутреннего уха может отсутствовать отоакустическая эмиссия, что может повлиять на характеристики эхограммы лишь незначительно. Полная или частичная глухота – фактор, не влияющий на параметры эхо-сигнала, но способный приносить неудобства пользователю.

– серные пробки – патологическое состояние, характеризующееся перекрыванием просвета наружного слухового прохода серным веществом (серой) плотной консистенции. Со временем сера накапливается в слуховом

канале, однако воздействие этого фактора на параметры эхо-сигнала минимально (далее показано, что через полгода после обучения системы, точность идентификации пользователей почти не меняется). Если на этапе идентификации у пользователя образовалась серная пробка, которая отсутствовала на этапе обучения системы (или наоборот), то частота ошибок «ложного отказа» пользователю возрастает, но это не может препятствовать авторизации.

Если количество ошибок «ложного отказа» возросло до некомфортного уровня вследствие отологических факторов, повторное обучение биометрической системы исправит ситуацию.

Технические факторы определяются следующими параметрами оборудования:

- чувствительность микрофона (достаточно обеспечить покрытие частот в диапазоне от 1 кГц до 14 кГц);
- количество внутренних шумов;
- диапазон воспроизводимых частот динамика и звуковой платы (должен соответствовать чувствительности микрофона);
- звукоизолирующие свойства корпуса наушников.

Степень влияния последнего фактора пока не изучалась. Использование одной модели оборудования на этапе обучения и идентификации гарантирует неизменность сигнала.

Условия использования. Окружающая обстановка при хороших звукоизолирующих свойствах корпуса наушников не влияет на эхо-сигнал. Причиной искажения эхо-сигнала может стать неплотное прилегание наушников, выполнение активных движений головой, а также прием пищи (движения челюстью) во время идентификации. Однако эти факторы нивелируются выполнением простых требований.

Рассмотрим некоторые работы, в которых использовался анализ акустических характеристик уха.

Работа [307] является одной из первых, где впервые описана идея построения вектора биометрических параметров на основе акустических свойств ушной раковины. Однако в ней отсутствуют данные о надежности предложенного подхода (EER, FRR, FAR).

В работе [182] ушной канал зондировался звуковыми волнами в диапазоне частот 1,5Гц–22кГц. Вектор признаков включал по 256 значений на каждое ухо (всего 512 биометрических параметров). Признаки извлекались с использованием фильтра на основе метода перевалов (Method Of Steepest Descent). Для сравнения образов использовался средний уровень корреляций между векторами признаков (предъявленным и эталонным). Полученные показатели EER зависят от диапазона частот и используемого устройства и варьировались от 0,8% до 18%.

В работе [313] акустические характеристики уха используются для генерации криптографических ключей. Данные характеристики регистрируются путем измерения передаточных функций от наушника к уху (headphone-to-ear-canal Transfer Functions, HpTF). Квантование данных производится нечётким экстрактором (Fuzzy Commitment) с применением метода главных компонент и случайного ортогонального преобразования. Размер вектора признаков составляет 512 бит, по 256 бит для каждого уха. Сравнение образов производится методом ближайших соседей.

В [313] было проведено 2 эксперимента с привлечением 45 и 65 испытуемых (по 8 замеров на каждого). В первом случае наушники в течение всего эксперимента находились на испытуемых, во втором – наушники каждый раз снимались и одевались при каждом измерении. Монтаж динамика и микрофона существенно повлиял на вероятность ошибочных решений.

В сентябре 2019 года команда исследователей [222] разработала прототип системы непрерывной биометрической аутентификации пользователей. Прототип получил название EarEcho и служит для подтверждения личности пользователя при ношении наушников. Процесс аутентификации EarEcho состоит из нескольких этапов: акустическое зондирование, предварительная обработка сигнала, извлечение признаков и аутентификация пользователя. На первом этапе

динамиком излучается возрастающий ЛЧМ-сигнал (линейная частотная модуляция), охватывающий наиболее часто используемые звуковые полосы частот в диапазоне от 20 Гц до 6 кГц, который распространяется через ушной канал, отражается и фиксируется микрофоном. Конструкция наушников выполнена таким образом, чтобы снизить помехи при излучении сигнала. Силиконовый вкладыш служит для шумоизоляции от внешней среды. На этапе предварительной обработки используется модуль контроля адаптивного усиления (Adaptive Gain Control, AGC), который подавляет нежелательные шумовые сегменты. Для исключения влияния сегментов с низким отношением «сигнал-шум» применяются тест отношения правдоподобия (Likelihood Ratio Test, LRT) и скрытые марковские модели. При помощи избирательного фильтра нижних частот отфильтровываются сигналы выше 6 кГц.

В эксперименте [222] приняло участие 20 испытуемых (возрастной диапазон 24-30 лет, 6 женщин и 14 мужчин). Во время сбора данных каждому субъекту было предложено надеть прототип наушников и прослушать 5 аудиозаписей длительностью в 2 минуты. Участники снимали и надевали наушники между каждой аудиозаписью. Для имитации сценария ежедневного использования участники принимали разные позы (например, сидя и стоя). Данные собирались в различных условиях (при разном уровне шума, например, комната, торговый центр, кафе, улица). Было собрано 11 900 образцов из 5 записей, каждый из которых представляет собой отрезок продолжительностью в 1 секунду. Образцы разделены на две части: 80% образцов отнесено к обучающей выборке, 20% - к тестовой. Уровень звукового давления по умолчанию для обучения и тестирования составляет 55 дБ. При испытании использовались классификаторы ближайших соседей (K-Nearest Neighbor, k-NN), дерево принятия решений (Decision Tree, DT), наивный Байес (Naive Bayesian, NB), метод опорных векторов (Support Vector Machine, SVM), многослойный персептрон (Multi-layer Perceptron, MLP).

Преимущество методов распознавания личности человека по параметрам внешнего уха заключается в том, что этот тип образов скрыт от

непосредственного наблюдения. У злоумышленника нет возможности создать достаточно информативную копию трехмерной модели внешнего уха скрыто от владельца или дистанционно.

4.4 Формирование и анализ набора данных акустических образов ушного канала субъектов

Разработано устройство для регистрации биометрических характеристик уха (рисунок 4.5), которое состоит из двух электретных микрофонов (с шумом 36 дБА, чувствительностью 60 мВ/Па и диапазоном частот 20–20000 Гц), звукоизолирующего корпуса (в виде наушников), экранированного медного провода, двух динамиков (мощностью 0,5 Вт и диапазоном частот 850–20000 Гц), штекера (3,5 мм.) и звуковой карты фирмы CREATIVE (с частотой квантования 44000 Гц и разрядностью 24 бита).

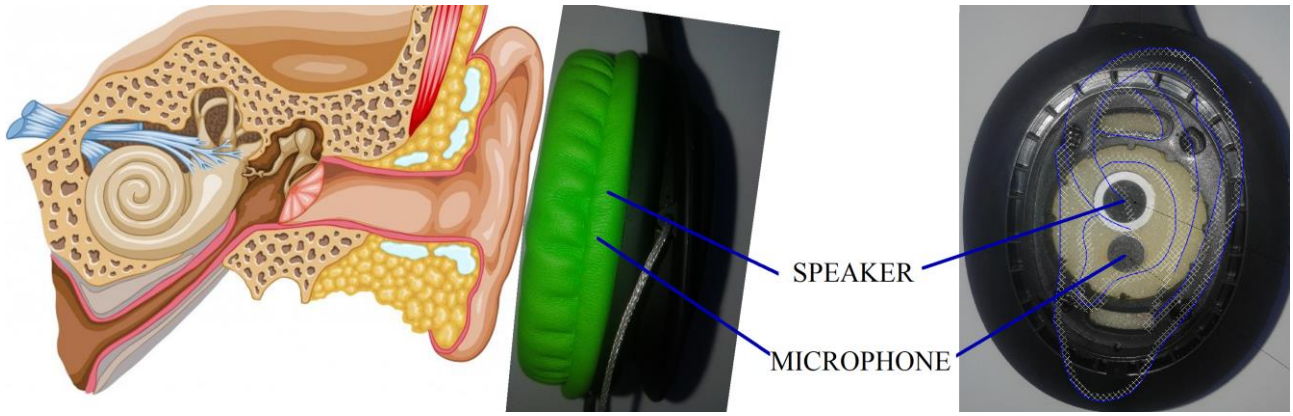


Рисунок 4.5 – Устройство для записи акустических образов наружного уха

Для сбора биометрических образов привлечено 75 человек (мужчин и женщин примерно в равном соотношении, в возрасте от 18 до 40 лет *без отологических патологий*). Каждому испытуемому было предложено прослушать звуковой моно-сигнал $\sigma(t)$ возрастающей и убывающей частоты (скользящий модулированный синус), получаемый путем линейной частотной модуляции (ЛЧМ), где t – это время в дискретной форме. Частота сигнала варьировалась в

диапазоне от 1 кГц до 14 кГц, длительность сигнала составляла 10 секунд, средняя громкость – 80 дБ. Сигнал воспроизводился через два динамика (для правого и левого уха), эхо-сигнал одновременно регистрировался монтированными в корпус наушников микрофонами. Частота дискретизации регистрируемого эхо-сигнала составляла 44 кГц (запись выполнялась в режиме моно). Все испытуемые прослушали сигнал по 15 раз, каждый раз снимая и надевая наушники заново (чтобы учесть зависимость эхо-сигнала от монтажа).

Регистрируемый звуковой эхо-сигнал $u_{i,k}(t)$ можно назвать эхограммой наружного уха или *акустическим образом уха* субъекта, где i – номер испытуемого, k – номер попытки ввода. Сформированный набор данных представлен в виде совокупности wav-файлов (моно, 44 кГц, 16 бит) и находится в открытом доступе [178] (AIC-ears-75).

Так как устройства ввода и сигнал $\upsilon(t)$ для всех испытуемых были идентичны, то для поиска индивидуальных отличий в строении ушей субъектов можно непосредственно анализировать параметры эхо-сигнала $u_{i,k}(t)$ без построения передаточных функций на основе $u_{i,k}(t)$ и $\upsilon(t)$. Для анализа сигналов в работе применялось быстрое оконное преобразование Фурье (STFT).

Сигналы $u_{i,k}(t)$, а также их спектрограммы $S_{i,k}$ малоинформативны – отличия образов разных испытуемых слабо различимы, при этом эхо-сигналы в целом изменчивы, так как зависят от монтажа наушников (рис 4.6). Временная шкала ЛЧМ-сигнала жестко связана с частотной шкалой (рис 4.6). Чтобы выделить из акустических образов полезную информацию и снизить дисперсию случайных выбросов при разложении сигнала в ряды Фурье, спектрограммы были преобразованы в усредненный по всем окнам (по всем временным промежуткам) амплитудный спектр $\bar{A}'$ (рисунок 4.7):

$$\overline{A}' = \{A'_{\nu_{1494}}, A'_{\nu_{1495}}, ..., A'_{\nu_{20893}}\},$$

$$A'_{\nu_\kappa} = A'(\nu_\kappa) = \frac{\sum_{\tau=1}^{Q_{interval}} A_{\nu_\kappa,\tau}}{Q_{interval}},$$

$$S(\tau) = \{A_{\nu_1,\tau}, ..., A_{\nu_{W_{size}/2},\tau}\}$$

$$\nu_\kappa = \nu_1 \cdot \kappa, \nu_1 = \frac{1}{T}, T = W_{size} / 44000$$

где $Q_{interval}$ – количество временных промежутков по T секунд, на которые делится эхо-сигнал $u_{i,k}(t)$ с учетом размера окна W_{size}=65536 и шага W_{step}=32768; ν_κ – частота κ-х гармоник (в дискретной форме); $A_{\nu,\tau}$ – амплитуда гармоники с частотой ν, соответствующая временному промежутку под номером τ; A'_ν – средняя амплитуда гармоник с частотой ν. Так как частота ЛЧМ-сигнала $\upsilon(t)$ менялась в диапазоне от 1 кГц до 14 кГц, анализ ограничивался данным частотным диапазоном. Так при размере окна W_{size}=65536 (T≈1,49, ν_1≈0,67) учитывались гармоники с частотой от 1000,98 Гц до 13998,31 Гц, поэтому спектр $\bar{A}'$ составлен из 19400 усредненных амплитуд, с учетом гармоник с номерами 1494≤κ≤20893. Размер окна выбран таковым, чтобы можно было проанализировать частоты сигнала $u_{i,k}(t)$ с точностью не менее 1 Гц. Результаты описанного преобразования, продемонстрированного на рисунке 4.7, достаточно робастные. Усредненный спектр $\bar{A}'_{i,k}$ гораздо информативнее, чем эхо-сигнал $u_{i,k}(t)$ и его спектрограмма $S_{i,k}(\tau)$: спектры $\bar{A}'_{i,k}$ имеют отличия у разных испытуемых, которые видны на графиках (рисунок 4.8).

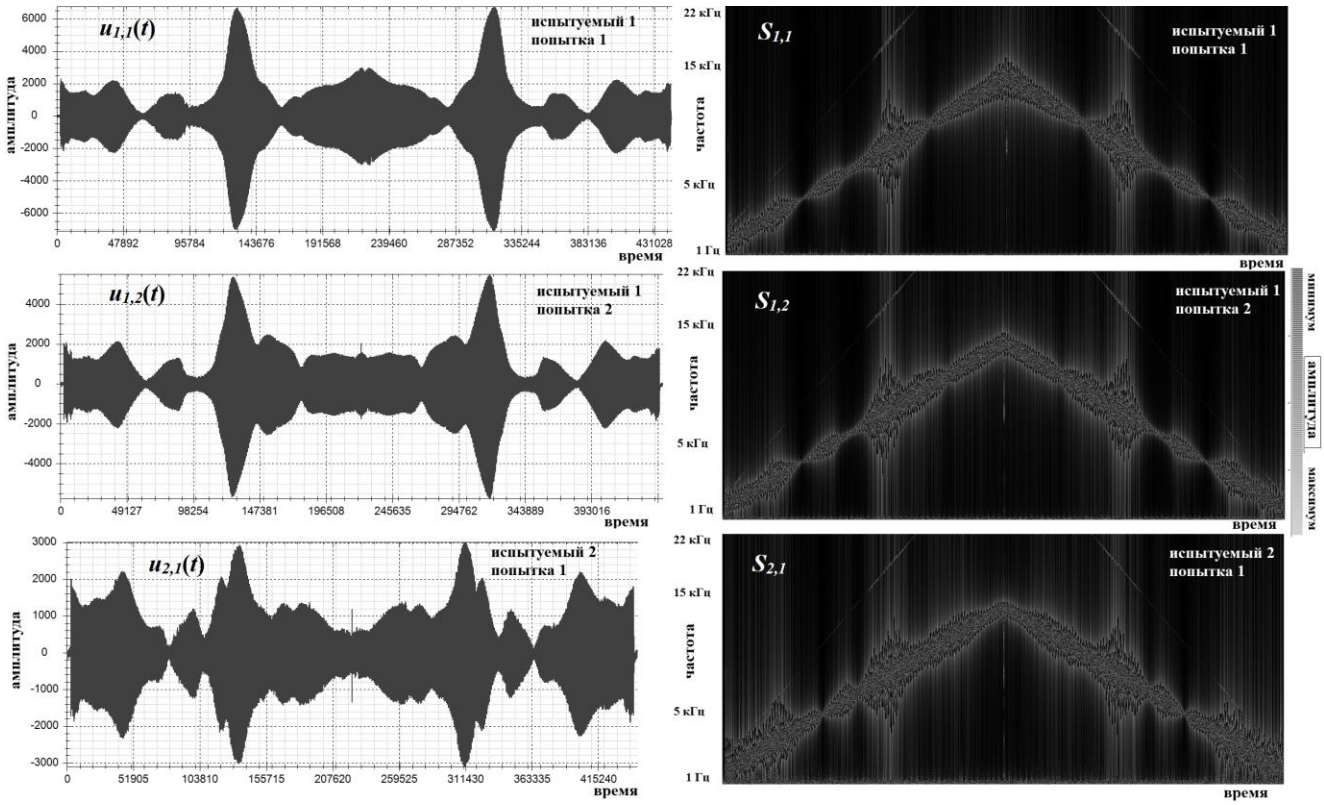


Рисунок 4.6 – Эхо-сигналы (слева) и их спектрограммы при параметрах STFT: прямоугольное окно, W_{size}=65536, W_{step}=32768 (справа)

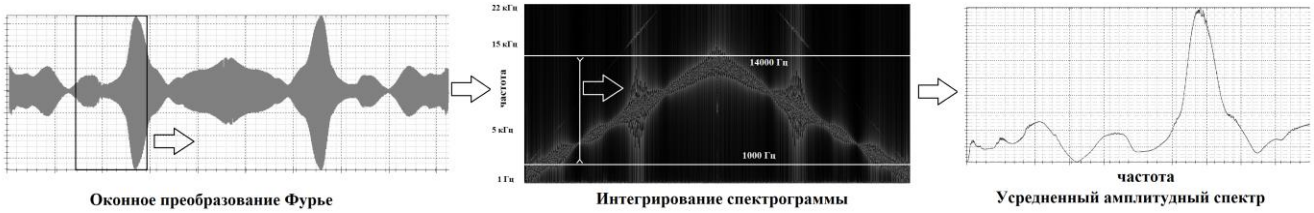


Рисунок 4.7 – Процесс получения усредненного амплитудного спектра эхо-сигнала $u_{i,k}(t)$.

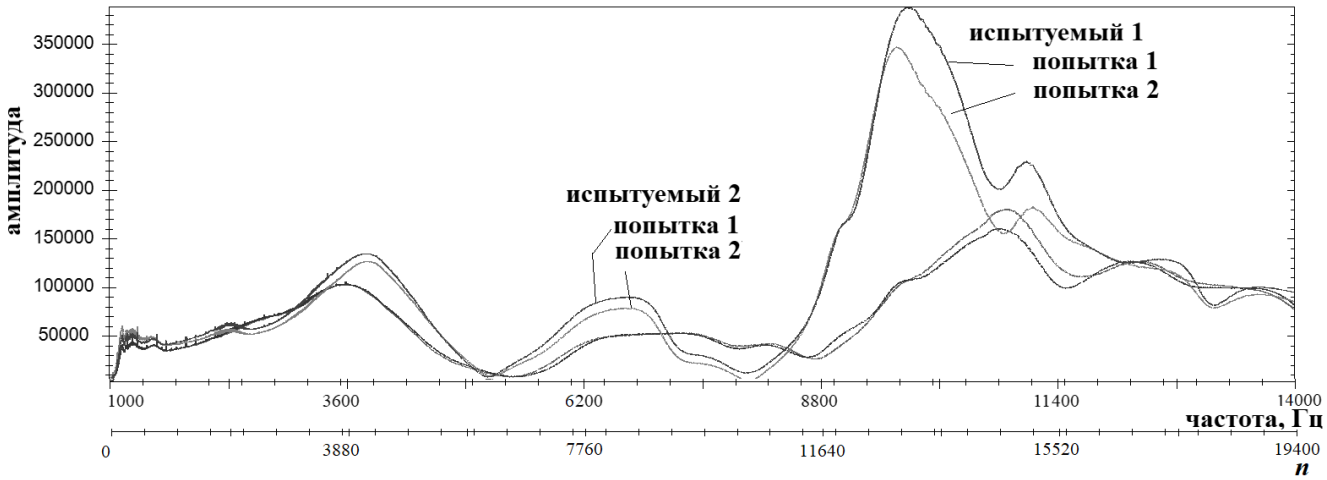


Рисунок 4.8 – Усредненные спектры эхо-сигнала для правого уха (окно Хэмминга, W_{size}=65536, W_{step}=32768)

Чтобы выявить локальные особенности усредненного спектра были построены кепстрограммы $K_{i,k}$ (рисунок 4.9). Обычно под кепстром понимается результат обратного преобразования Фурье от логарифма спектра мощности сигнала [258]. Однако в настоящей работе предложено получать кепстрограммы путем применения STFT по отношению к усредненному спектру $\bar{A}'_{i,k}$ без применения операции возведения в логарифм, по аналогии с тем, как это выполнялось по отношению к исходному сигналу $u_{i,k}(t)$ при построении спектрограммы. Так частотная шкала v спектральной функции $A'_{i,k}(v)$ принималась за временную шкалу, усредненный спектр делился на частотные интервалы $\Delta v = W^*_{size}/19400$ в соответствии с размером окна W^*_{size} и шагом W^*_{step}. Далее каждый интервал раскладывался в ряд Фурье, и для интервалов строились амплитудные спектры или *спектры кепстральных коэффициентов* $C_{\kappa,\iota}$, совокупность которых представляет собой кепстрограмму:

$$K_{i,k} = \begin{pmatrix} C_{1,1} & \dots & C_{1,\iota} & C_{1,\iota+1} & \dots \\ \dots & \dots & \dots & \dots & \dots \\ C_{\kappa,1} & \dots & C_{\kappa,\iota} & C_{\kappa,\iota+1} & \dots \\ C_{\kappa+1,1} & \dots & C_{\kappa+1,\iota} & C_{\kappa+1,\iota+1} & \dots \\ \dots & \dots & \dots & \dots & \dots \\ C_{W^*_{size}/2,1} & \dots & C_{W^*_{size}/2,\iota} & C_{W^*_{size}/2,\iota+1} & \dots \end{pmatrix}$$

где ι – номер частотного интервала, κ – номер кепстрального коэффициента, соответствующего частоте $\kappa/\Delta v$. В ходе эмпирических исследований установлены следующие оптимальные параметры для STFT: $W^*_{size}=16$ и $W^*_{step}=13$. При $W^*_{size}<16$ наблюдалось более высокое количество ошибок идентификации испытуемых, при $W^*_{size}>16$ – количество ошибок не снижалось. Шаг W^*_{step} устанавливался так, чтобы было незначительное перекрытие окон.

На кепстрограмме видны отличительные особенности для разных испытуемых, которые сложно заметить на усредненных спектрах (рисунок 4.9).

Отметим, что информативность кепстрограмм и усредненного спектра зависит от типа оконной функции. Комбинируя разные типы окон на этапе вычисления усредненного спектра (W_{type}) и кепстрограммы (W^*_{type}) можно получить больше информации об особенностях строения ушного канала

испытуемых (рисунок 4.9). Иначе выражаясь, при разложении функций $u_{i,k}(t)$ и $A'(v_к)$ в ряды Фурье оптимальными могут оказаться различные оконные функции. В исследовании использовались следующие типы окон и их комбинации: прямоугольное, Барлетта (треугольное), гауссиана классическая (с параметром формы p=1), Лапласа, гауссиана параметрическая (со значением параметра p=1,5), Блэкмана, Хэмминга.

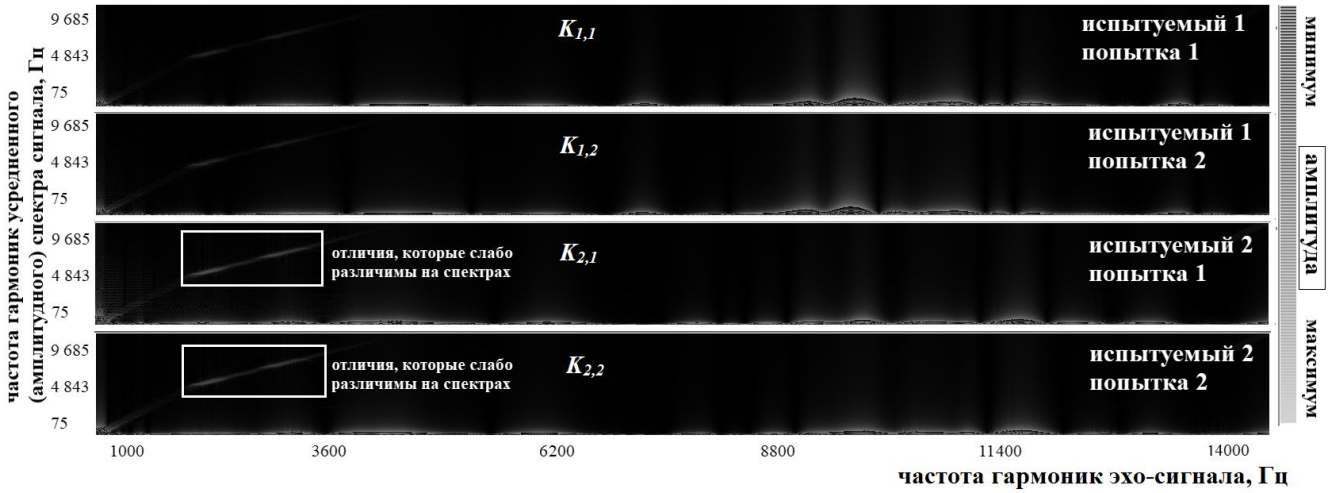


Рисунок 4.9 – Кепстрограммы эхо-сигналов (параметры STFT: W_{type} – окно Хэмминга, W_{size}=65536, W_{step}=32768; W^*_{type} – прямоугольное окно, W^*_{size}=256, W^*_{step}=13)

В качестве признаков могут быть использованы непосредственно показатели амплитуд спектров и кепстрограмм (далее спектральные и кепстральные признаки).

4.5 Эксперименты по распознаванию испытуемых с использованием классификатора Байеса, многослойных сверточных и полносвязных нейронных сетей

Без дополнительной обработки размерность пространства спектральных и кепстральных признаков слишком велика (n=19400 и n=11040, соответственно). Как показали эмпирические исследования количество спектральных признаков можно уменьшить без потерь в надежности идентификации до n=970, если в

качестве признаков a_j использовать показатели энергий (сумм из 20 амплитуд с близкими частотами):

$$a_j = E(\nu_{1494+20(j-1)}, \nu_{1494+20j}) = \sum_{\kappa=1494+20(j-1)}^{1494+20j} A'_{\nu_\kappa}$$

Число кепстральных признаков удалось уменьшить без снижения надежности распознавания испытуемых только в 2 раза (до n=5520), используя в качестве признаков энергии:

$$a_{j=(\iota-1)W^*_{size}/2+\kappa} = E_\iota(\nu_{2\kappa-1}, \nu_{2\kappa}) = C_{2\kappa-1,\iota} + C_{2\kappa,\iota}$$

Каждый признак можно рассматривать как случайную величину. Исследование показало, что законы распределения значений рассматриваемых признаков близки к нормальному для большинства испытуемых (что проверялось на основании критерия хи-квадрат Пирсона).

Схему «наивной» Байесовской классификации можно свести к следующему алгоритму [246, 280]. За n шагов вычисляются апостериорные вероятности гипотез, каждая из которых ассоциирована с определенным пользователем, зарегистрированным в системе (n – это количество признаков). На первом шаге алгоритма гипотезы обычно считаются равновероятными, если нет статистических данных относительно них [246, 280], в настоящей работе придерживались данного правила. На каждом шаге апостериорные вероятности пересчитываются по формуле (4.1), при этом за априорную вероятность принимается апостериорная вероятность, вычисленная на предыдущем шаге. Решение принимается в пользу гипотезы с наивысшей апостериорной вероятностью на последнем шаге.

$$P_h(a_j) = \frac{P_h(a_{j-1})\,p_h(a_j)}{\sum_{i=1}^{N} P_i(a_{j-1})\,p_i(a_j)}, \qquad (4.1)$$

где N – количество гипотез (идентифицируемых субъектов, N=75), $P_h(a_j)$ – апостериорная вероятность h-й гипотезы, зависящая от j-го признака ($P_h(a_0)$=1/N), $p_h(a_j)$ – условная плотность вероятности h-й гипотезы на j-м шаге классификации. Плотности вероятности $p_h(a_j)$ при Байесовской классификации допустимо

использовать вместо условных вероятностей (безразмерные значения в числителе и знаменателе при расчетах сокращаются и $P_h(a_j)$ принимает значения от 0 до 1). Так как в упрощенной форме признаки можно условно описать функцией плотности вероятности нормального закона распределения, то $p_h(a_j)$ вычислялись по формуле:

$$p_h(a_j) = \frac{1}{\sigma_{h,j}\sqrt{2\pi}} e^{-\frac{(a_j-\mu_{h,j})^2}{2\sigma_{h,j}^2}}$$

где $\mu_{h,j}$ – математическое ожидание значений j-го признака, характерное для испытуемого под номером h, $\sigma_{h,j}$ – среднеквадратичное отклонение значений j-го признака, характерное для испытуемого под номером h. Обучить «наивный» Байесовский классификатор означает вычислить параметры $\mu_{h,j}$ и $\sigma_{h,j}$ по данным обучающей выборки пользователей (испытуемых), ассоциированных с соответствующими гипотезами.

Проведен эксперимент по идентификации субъектов. Строился классификатор Байеса, который обучался на 8 примерах образа от каждого испытуемого. Остальные образы использовались для тестирования. Вероятность ошибок вычислялась как отношение числа неверных решений (когда была признака ошибочная гипотеза) к общему количеству опытов. Так как идентификация производилась на замкнутом множестве (при N=75), то неверное решение одновременно приводило к «ложному отказу» (FRR) и «ложному допуску» (FAR), соответственно вероятности FRR и FAR в данном случае были равны. Поэтому процент ошибок, вычисленный по критерию «максимальной апостериорной вероятности», будет численно равен коэффициенту равной вероятности ошибок (EER) при пороге $P_h(a_j)$ = 0,5 (только одна гипотеза может его преодолеть).

Эквивалентность критериев «наибольшей апостериорной вероятности» и EER можно проиллюстрировать, если вести подсчет ошибок следующим образом. Ошибку «ложного отказа» следует фиксировать, когда апостериорная вероятность гипотезы, ассоциированной с заявленным логином (номером) испытуемого, не

преодолевает порог, а ошибку «ложного допуска» – когда апостериорная вероятность любой другой гипотезы преодолевает порог. На рисунке 4.10 можно видеть, что при пороге $P_h(a_j)$=0,5 выполняется равенство EER=FRR=FAR (это справедливо, только если речь идет о «закрытом» множестве классов). Также из рисунка 4.10 можно заключить, что при использовании классификатора Байеса показатели ошибок очень сложно балансировать – соотношение FRR и FAR почти не меняется на интервале (0;1).

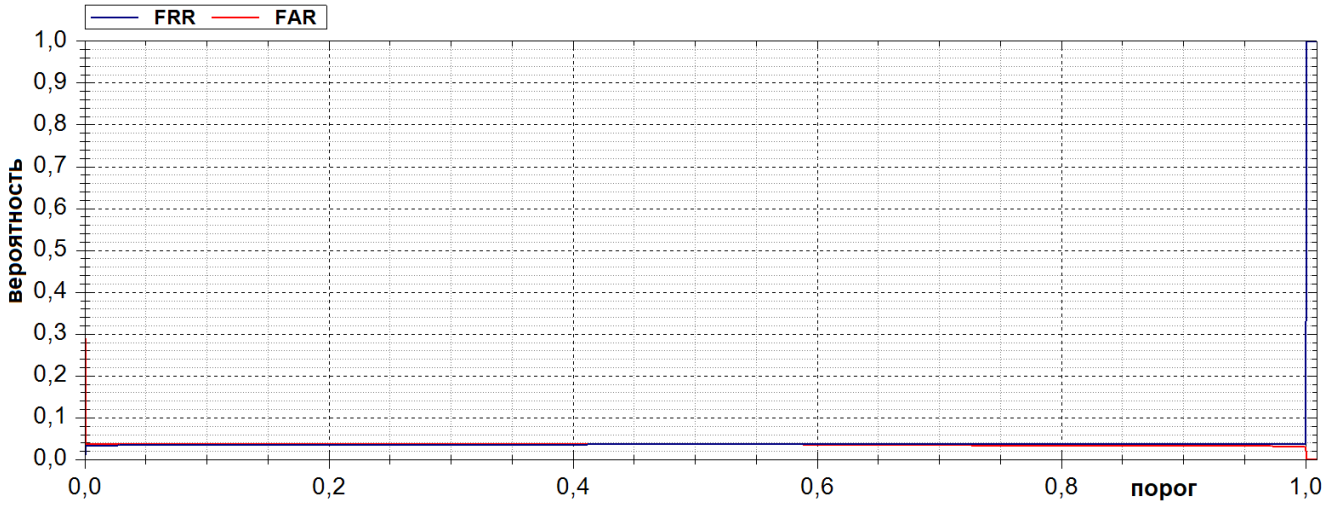


Рисунок 4.10 – ROC-кривые, построенные по результатам идентификации образов правого уха 75 испытуемых с учетом 2326 кепстральных признаков (Hamming + rectangular)

Результаты по идентификации испытуемых на основе параметров одного уха представлены в таблицах 4.1 и 4.2. Как можно видеть, усредненные спектры менее информативны, чем кепстрограммы (кепстральные признаки дают меньший процент ошибок). Лучшие результаты получаются при сочетании одного из окон Барлетта, Блэкмана или Хэмминга с прямоугольным окном (таблица 4.2). При сочетании окна Хэмминга (на этапе вычисления спектра) с прямоугольным окном (на этапе вычисления кепстрограмм) достигнут наименьший уровень ошибок EER=0,0239 при наименьшем количестве признаков n=2326. Таким образом, использование 2326 первых кепстральных признаков является оптимальным для рассматриваемого набора данных.

Таблица 4.1 – EER при идентификации 75 испытуемых по параметрам правого уха на основе спектральных признаков в зависимости от типа оконных функций (при n=970)

Прямоугольное	Барлетта	гауссиана (p=1)	Лапласса	гауссиана (p=1,5)	Блэкмана	Хэмминга
0,1117	0,1223	0,1223	0,1117	0,1196	0,125	0,125

Таблица 4.2 – EER при идентификации 75 испытуемых по параметрам правого уха на основе кепстральных признаков в зависимости от типа оконных функций (при n=5520)

Тип окна: спектров / кепстров	Прямоугольное	Барлетта	гауссиана (p=1)	Лапласса	гауссиана (p=1,5)	Блэкмана	Хэмминга
Прямоугольное	0,0877	**0,0398**	0,1196	0,101	0,1303	**0,0398**	**0,0398**
Барлетта	0,0984	0,1037	0,1037	0,1063	0,1063	0,1117	0,1037
гауссиана (p=1)	0,125	0,1303	0,125	0,1329	0,125	0,1329	0,1329
Лапласа	0,125	0,1303	0,1223	0,1329	0,1223	0,1303	0,1303
гауссиана (p=1,5)	0,125	0,1303	0,1223	0,1329	0,125	0,1329	0,1329
Блэкмана	0,0984	0,077	0,1117	0,101	0,1143	0,0904	0,0771
Хэмминга	0,101	0,0984	0,0984	0,0957	0,0957	0,0957	0,0984

На рисунке 4.11 представлены показатели ошибок при объединении по 2326 кепстральных признаков (окно Хэмминга + прямоугольное окно) для сигналов от правого и левого ушей, соответственно (n=4652). При идентификации в «двухканальном» режиме (когда зондируются сразу два уха субъекта) вероятность ошибок становится значительно ниже и составляет EER=0,0053.

Был проведен аналогичный эксперимент, но в режиме верификации по методу перекрестной проверки. В режиме верификации для каждого пользователя обучался отдельный Байесовский классификатор, и определялось две гипотезы (N=2): «Свой» (ассоциирована с тем испытуемым, логин/номер которого заявлен) и «Чужой» (ассоциирована с генеральной совокупностью всех возможных пользователей). Для гипотезы «Свой» условные плотности вероятности $p_h(a_j)$ вычислялись с учетом параметров $\mu_{0,j}$ и $\sigma_{0,j}$ конкретного испытуемого, для гипотезы «Чужой» – параметры $\mu_{1,j}$ и $\sigma_{1,j}$ рассчитывались на основании выборки данных других испытуемых (с учетом гипотезы нормального распределения

значений признаков). Обучение Байесовских классификаторов проводилось на данных 50 испытуемых («Своих») по 8 случайных образов от каждого. Для подсчета ошибок «ложного отказа» бралось 7 тестовых образов «Своих», не использовавшихся при обучении. Образы остальных 25 испытуемых («Чужих») применялись для тестирования на выявление ошибок «ложного допуска».

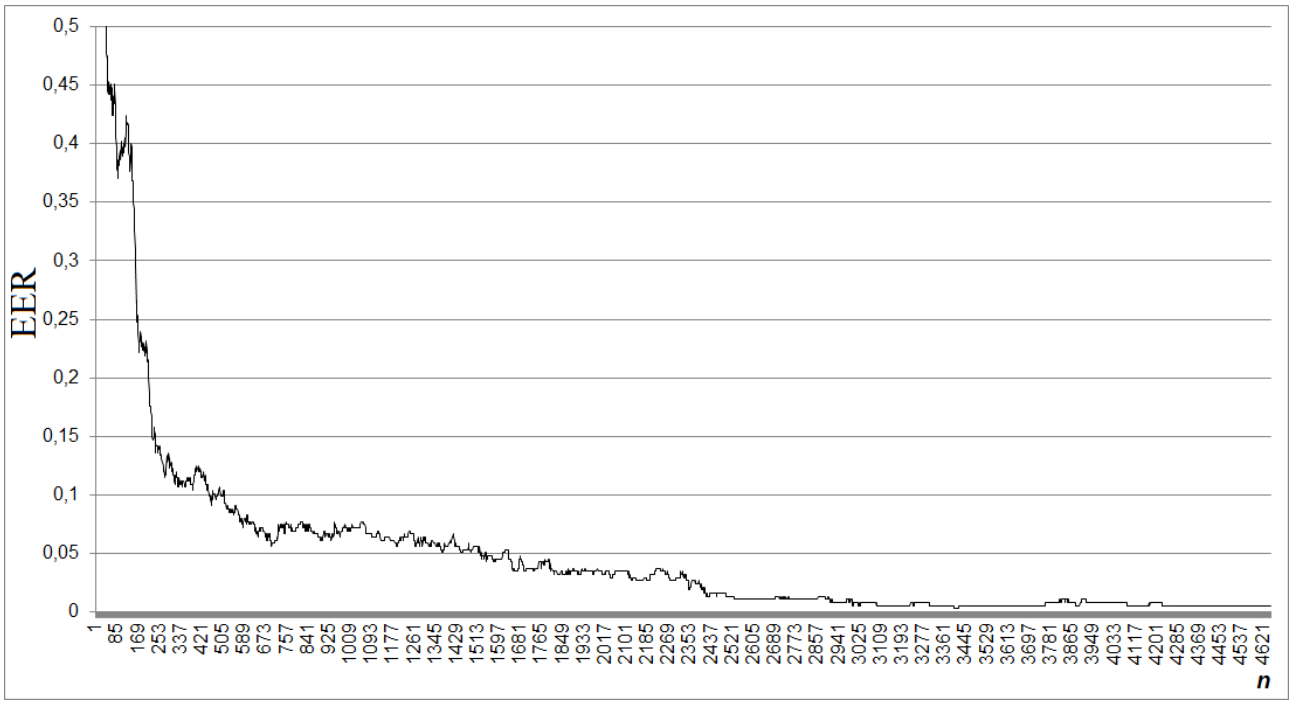


Рисунок 4.11 – Вероятности ошибок идентификации 75 испытуемых по параметрам двух ушей в зависимости от количества кепстральных признаков (W_{type} – окно Хэмминга, W^*_{type} – прямоугольное окно)

Опыты повторялись трижды – каждый раз определялось новое множество «Чужие», не пересекавшиеся с предыдущим. Далее были рассчитаны вероятности ошибок при пороге $P_h(a_j)$=0,5 (рисунок 4.12), которые при n=4652 составили: FRR=0,1028 при FAR<0,0001 (в рамках эксперимента не было зафиксировано ни одной ошибки «ложного допуска»). Чтобы определить FAR с точностью до более высоких порядков, нужна выборка гораздо большего объема. Однако полученный результат можно назвать очень оптимистичным, так как он удовлетворяет практическим целям (для биометрических систем важно иметь FAR близкий к нулю при приемлемом количестве отказов).

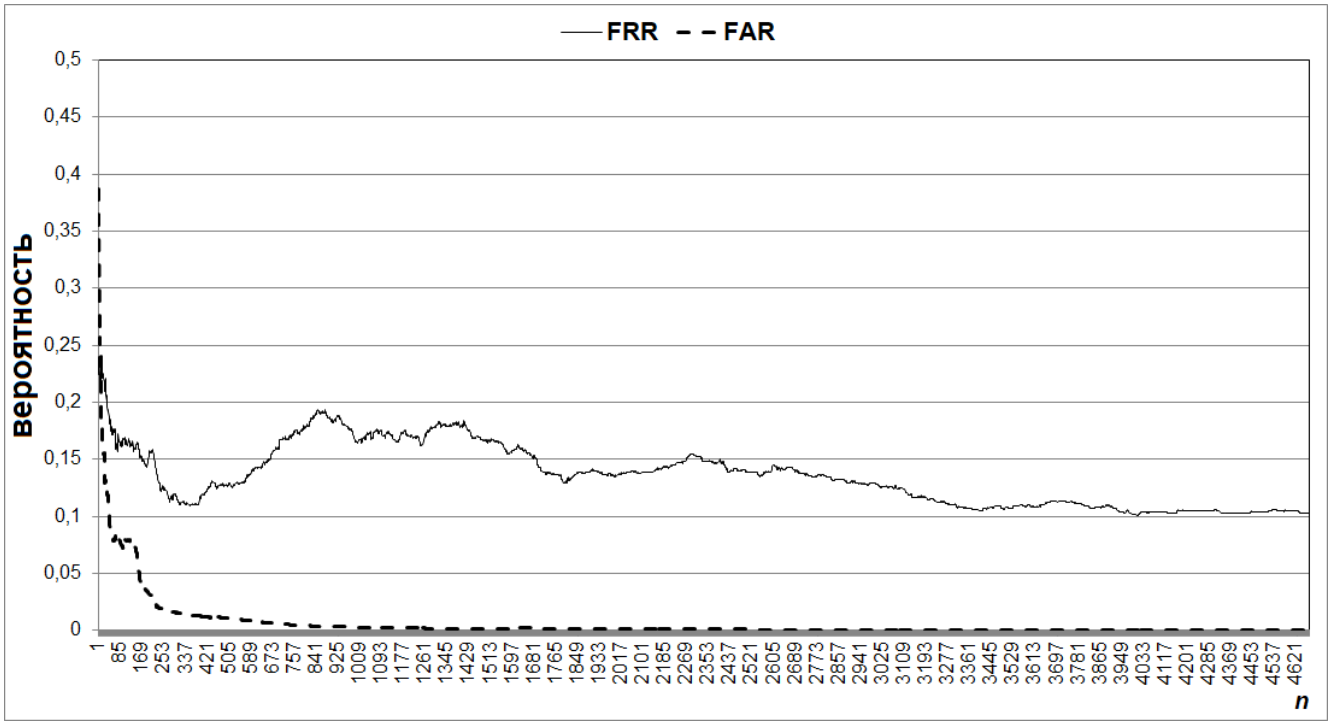


Рисунок 4.12 – Вероятности ошибок верификации образов двух ушей 75 испытуемых в зависимости от количества кепстральных признаков (W_{type} – окно Хэмминга, W^*_{type} – прямоугольное окно).

Натурный эксперимент с привлечением тех же испытуемых был проведен повторно через полгода после первоначального сбора данных. При этом использовался Байесовский классификатор, обученный на «закрытом» множестве классов полгода назад. Каждый субъект ввел биометрический образ 10 раз, после чего был идентифицирован. По результатам эксперимента зафиксировано всего 6 ошибок из 750 опытов (EER=0,008), что несущественно больше первоначального показателя (EER=0,0053). Таким образом, можно заключить, что свойства наружного слухового прохода, пригодные для идентификации личности пользователя, со временем заметно не меняются (в том числе, на них не влияет накопление серы в ушном канале).

При использовании нейронных сетей в задачах классификации количество нейронных слоев обычно стараются повысить, а число признаков сократить. Чем больше размерность входных данных, тем большим количеством параметров должна обладать сеть, способная эффективно их анализировать. Нейронные сети с большим числом параметров потенциально могут дать более высокую точность

решений, но при этом возрастает объем обучающей выборки. Построение оптимальной архитектуры – это поиск компромисса, между размерностью входа, объемом сети и объемом обучающей выборки.

Сформировано 8 различных архитектур (6 с использованием сверточных слоев, 2 без) искусственных нейронных сетей (ИНС) с разными конфигурациями (функциями активации и другими параметрами слоев), ориентированных на обработку, как усредненных спектров, так и кепстрограмм. Применение сверточных нейронных сетей обусловлено их способностью извлечения высокоинформативных признаков из графических объектов и временных рядов. Перед поступлением образов на вход в ИНС данные подготавливались:

- усредненные спектры вычислялись с параметрами W_{type} – окно Хэмминга, W_{size}=65536, W_{step}=32768, и далее интерполировались (исходная длина спектров была слишком велика) до A_{len}=300, A_{len}=600 или A_{len}=1200 амплитуд, значения амплитуд приводились к интервалу [0;1];
- кепстрограммы вычислялись с применением прямоугольного окна (W^*_{type}) с различными вариантами размера окна и шага (W^*_{size}=8, W^*_{step}=6 либо W^*_{size}=16, W^*_{step}=13 либо W^*_{size}=32, W^*_{step}=24), значения кепстральных коэффициентов приводились к интервалу [0;1].

Объем обучающей и тестовой выборок был аналогичен прошлому эксперименту (по 8 примеров для обучения и по 7 для тестирования от каждого испытуемого). При обучении ИНС использовался оптимизатор Adam. Обучение ИНС выполнялось по 25 эпох, после чего выполнялось промежуточное тестирование, при переобучении процесс останавливался и для рассматриваемой модели ИНС фиксировался наилучший результат.

Три наиболее показательные модели ИНС были выбраны среди прочих путём сравнения по точности идентификации 75 испытуемых (рисунок 4.13, таблица 4.3). Результаты представлены на рисунке 4.14.

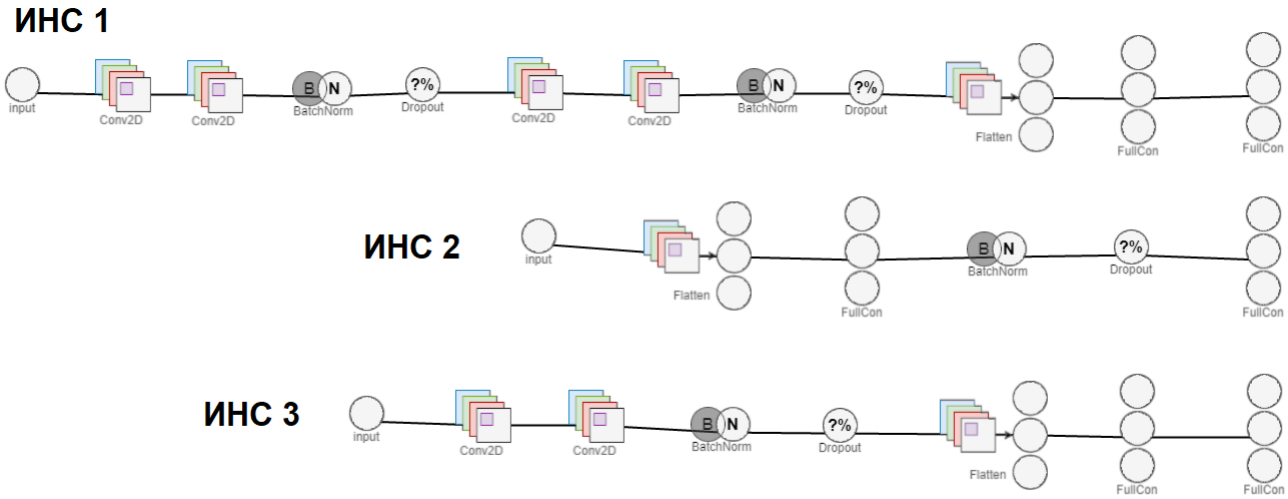


Рисунок 4.13 – Архитектуры, которые показали наилучшие результаты в рамках эксперимента.

Из представленных результатов видно, что FRR и FAR в случае ИНС легче балансировать, чем в случае Байесовского классификатора. Тем не менее, Байесовский классификатор дает меньшее число ошибочных решений в режиме идентификации (EER=0,0053 против EER=0,0266). Прежде всего, это обусловлено малым объемом обучающей выборки (8 примеров на испытуемого), которого, по всей видимости, недостаточно для эффективного обучения ИНС. Увеличивать объем обучающей выборки нецелесообразно, так как на практике система идентификации (аутентификации) пользователей должна обучаться за короткий промежуток времени. Обучение системы на 8 примерах занимает порядка 2-3 минут, при увеличении количества обучающих примеров до 16, 24, 30 и т.д. примеров время настройки биометрической системы возрастет до 5, 10 минут и более.

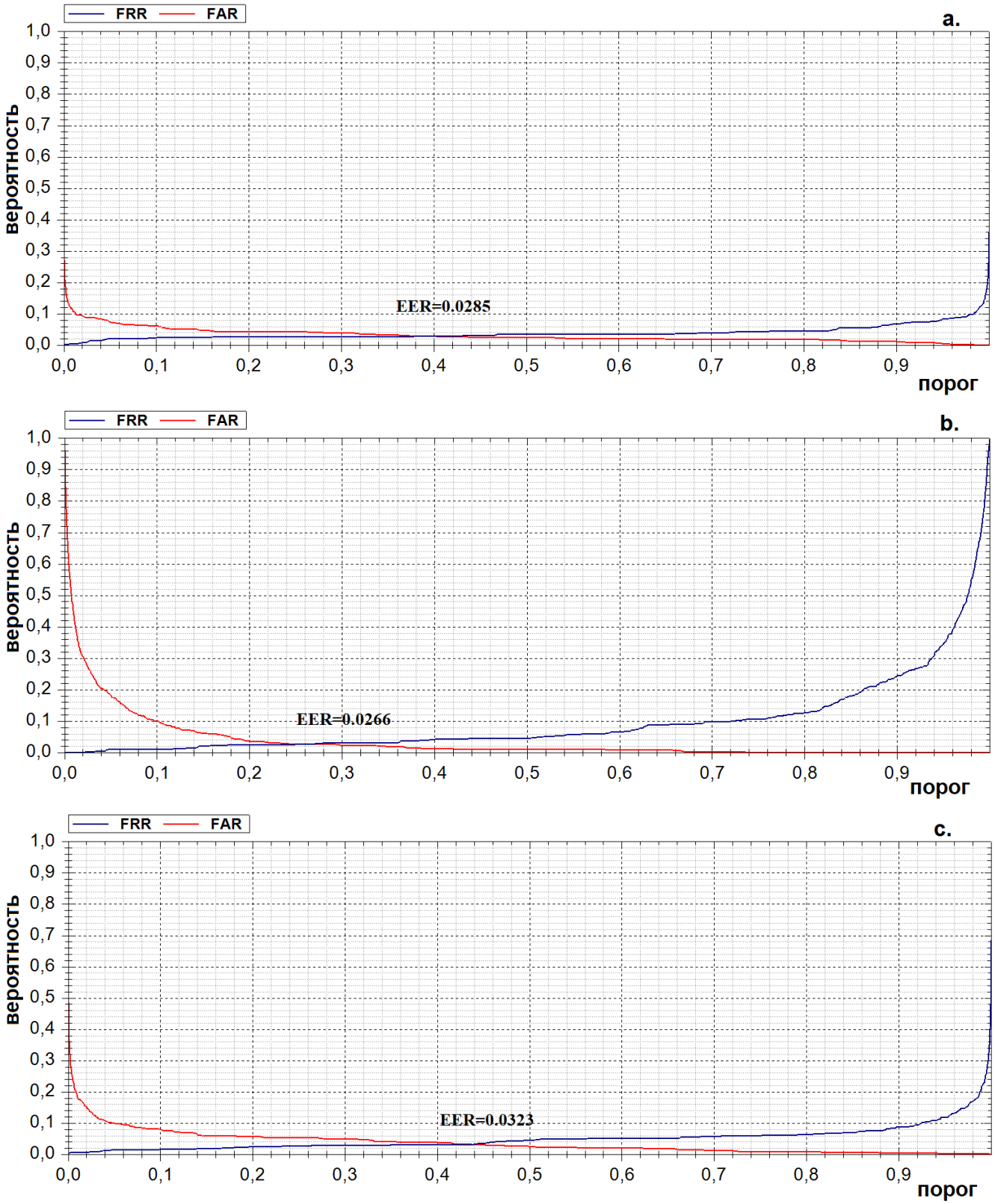


Рисунок 4.14 – ROC-кривые, построенные по результатам идентификации образов двух ушей:

а. ИНС №1 после 600 эпох обучения (кепстрограммы, W^*_{size}=8, W^*_{step}=6);

б. ИНС №2 после 50 эпох обучения (усредненные спектры, A_{len}=600);

в. ИНС №3 после 100 эпох обучения (усредненные спектры, A_{len}=600)

Таблица 4.3 – Конфигурации нейронных сетей, показавших наилучшие результаты.

ИНС №1	
Тип слоя	*Параметры слоя*
Входной	размерность = 2,4,149 ([номер уха][κ][ι])
Сверточный 2D	кол-во фильтров=4, окно свёртки=2,4, шаг свёртки=2,3
Сверточный 2D	кол-во фильтров=8, окно свёртки=2,4, шаг свёртки=2,3
Пакетная нормализация	
Dropout	rate = 0,3
Сверточный 2D	кол-во фильтров=16, окно свёртки=1,6, шаг свёртки=1,5
Сверточный 2D	кол-во фильтров=32, окно свёртки=1,6, шаг свёртки=1,5
Пакетная нормализация	
Dropout	rate = 0,3
Полносвязный слой	кол-во нейронов = 128, функция активации: relu
Полносвязный слой	кол-во нейронов = 75, функция активации: softmax
ИНС №2	
Тип слоя	*Параметры слоя*
Входной	размерность = 2,1,600 ([номер уха][1][A_{len}])
Полносвязный слой	кол-во нейронов = 150, функция активации: relu
Пакетная нормализация	
Dropout	rate = 0,1
Полносвязный слой	кол-во нейронов = 75, функция активации: softmax
ИНС №3	
Тип слоя	*Параметры слоя*
Входной	размерность = 2,1,600 ([номер уха][1][A_{len}])
Сверточный 2D	кол-во фильтров=4, окно свёртки=1,6, шаг свёртки=1,5
Сверточный 2D	кол-во фильтров=8, окно свёртки=1,6, шаг свёртки=1,5
Пакетная нормализация	
Dropout	rate = 0,1
Полносвязный слой	кол-во нейронов = 150, функция активации: relu
Полносвязный слой	кол-во нейронов = 75, функция активации: softmax

Настройку классификатора Байеса легко автоматизировать, в том время, как при обучении ИНС следует следить за возникновением переобучения. Также Байесовский классификатор легче применять в режиме верификации (две гипотезы – «Свой» и «Чужие»). «Прямое» обучение нейронной сети для решения задачи верификации образа на «открытом» множестве классов затруднительно, так как обучающая выборка «Свой» будет гораздо меньшего объема, чем обучающая выборка «Чужие» (такие эксперименты тоже проводились нами, но не увенчались успехом).

4.6 Биометрическая аутентификация по акустическим параметрам уха в защищенном режиме исполнения

Апробированы модель НПБК на базе корреляционных нейронов и алгоритм ее обучения. Проведена оценка взаимной корреляционной зависимости признаков, рассчитаны коэффициенты корреляции (2.1) между каждой парой признаков для каждого испытуемого в отдельности. По этим данным на рисунке 4.15 построены гистограммы относительных частот парных коэффициентов корреляции. Как можно видеть, взаимная зависимость между признаками кепстрограмм в целом ниже, чем между признаками усредненных спектров. Таким образом, усредненные спектры могут оказаться информативнее для НПБК.

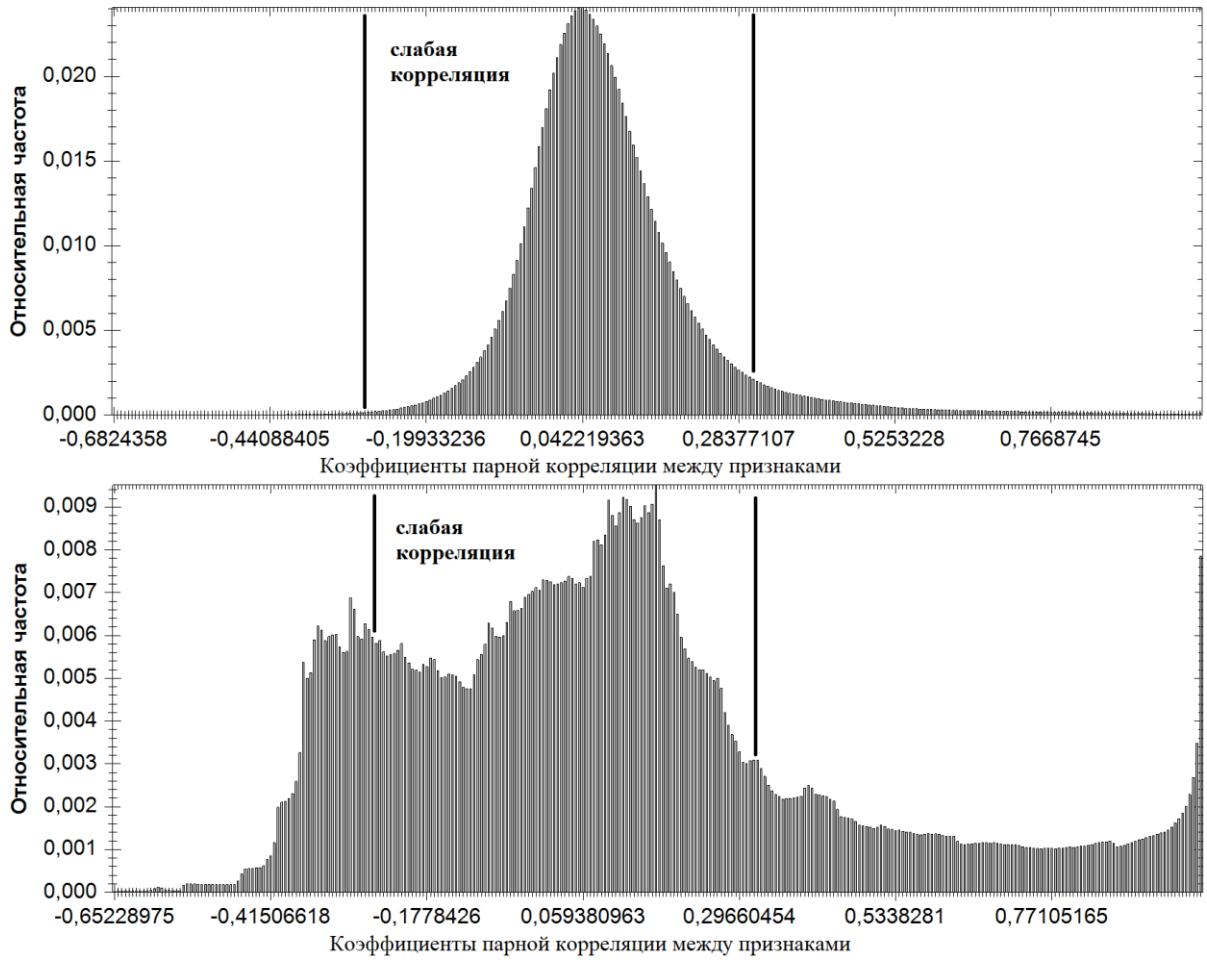


Рисунок 4.15 – Распределение коэффициентов корреляции между всеми парами признаков кепстрограмм (а) и усредненных спектров (б) для всех испытуемых

Для построения экстракторов признаков решено использовать две схожие архитектуры автокодировщиков (таблица 4.4). Для предобработки образов из акустических сигналов вычислялись спектрограммы (с использованием тех же типов окон) с размером окна W_{size}=65536 и шагом W_{step}=16384. Далее по всем окнам вычисляется спектр средних значений амплитуд в зависимости от частоты (рисунок 4.16). Из усредненного спектра удалялись первые 1500 и последние 3000 отчетов, чтобы не учитывать частоты менее 1 кГц (это начальная частота ЛЧМ-сигнала) и более 20 кГц (эти частоты не был способен регистрировать микрофон). Частоты 14-20 кГц учитывались, так как полезная информация может проявляться в обертонах. Далее полученные усредненные спектры «сжимались» до 2048 отчетов при помощи алгоритма линейной интерполяции сигналов. Перед подачей в нейронную сеть образы приводились к интервалу значений [0; 1].

В зависимости от применяемой оконной функции извлекались различные признаки. В случае использования сетей корреляционных нейронов выгодно использовать множество экстракторов признаков, которые дают сильно коррелированные, но не идентичные признаки.

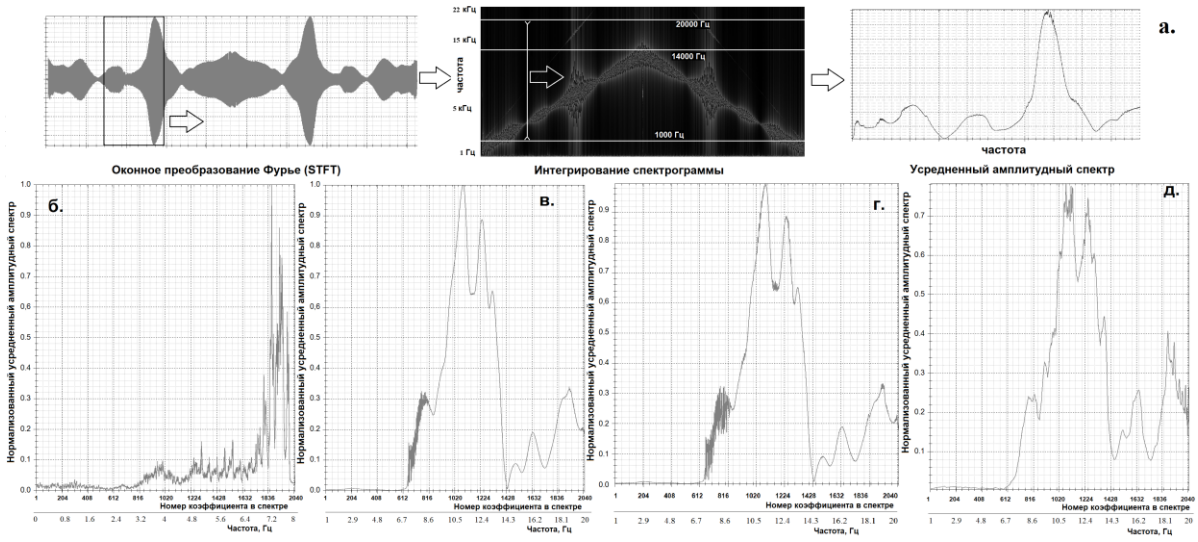


Рисунок 4.16 – Усредненный спектр:

а. процесс вычисления; б. извлеченный из голосового образа (из базы TIMIT) на базе прямоугольного окна; в. извлеченный из акустического образа уха на базе окна Хэмминга; г. извлеченный из акустического образа уха на базе прямоугольного окна; д. извлеченный из акустического образа уха на базе окна Хэмминга и восстановленный автокодировщиком.

Таблица 4.4. Конфигурации автокодировщиков

Architecture №1		**Architecture №2**	
Кодировщики			
Layer type	**Layer parameters**	**Layer type**	**Layer parameters**
Input	shape = 2048	Input	shape = 2048
Conv1D	filters=8, kernel_size=12, strides=4, activation=relu, initializer=glorot	Conv1D	filters=8, kernel_size=9, strides=2, activation=relu, initializer=he
Conv1D	filters=16, kernel_size=3, strides=2, activation=relu, initializer=glorot	Conv1D	filters=16, kernel_size=3, strides=2, activation=relu, initializer= he
Batch normalization		Batch normalization	
Conv1D	filters=16, kernel_size=4, strides=2, activation=relu, initializer=glorot	Conv1D	filters=16, kernel_size=4, strides=2, activation=relu, initializer= he
Conv1D	filters=32, kernel_size=3, strides=2, activation=relu, initializer=glorot	Conv1D	filters=32, kernel_size=3, strides=2, activation=relu, initializer= he
Batch normalization		Batch normalization	
Conv1D	filters=32, kernel_size=3, strides=2, activation=relu, initializer=glorot	Conv1D	filters=32, kernel_size=3, strides=1,2, activation=relu, initializer= he
Conv1D	filters=64, kernel_size=3, strides=2, activation=relu, initializer=glorot	Conv1D	filters=64, kernel_size=3, strides=2, activation=relu, initializer= he
Batch normalization		Batch normalization	
Conv1D	filters=64, kernel_size=3, strides=2, activation=relu, initializer=glorot	Conv1D	filters=64, kernel_size=3, strides=2, activation=relu, initializer= he
Conv1D	filters=64, kernel_size=3, strides=2, activation=relu, initializer=glorot	Conv1D	filters=64, kernel_size=3, strides=2, activation=relu, initializer= he
Batch normalization		Batch normalization	
Conv1D	filters=128, kernel_size=3, strides=2, activation=relu, initializer=glorot	Conv1D	filters=128, kernel_size=3, strides=2, activation=relu, initializer= he
Conv1D	filters=256, kernel_size=3, strides=2, activation=relu, initializer=glorot	Conv1D	filters=160, kernel_size=3, strides=2, activation=relu, initializer= he
Fully connected	units=128, activation=linear, initializer=glorot	Batch normalization	
		Conv1D	filters=192, kernel_size=3, strides=2, activation=relu, initializer= he
		Fully connected	units=128, activation=linear, initializer=glorot
Декодировщики			
Layer type	**Layer parameters**	**Layer type**	**Layer parameters**
Input	shape = 128	Input	shape = 128
Conv1D Transpose	filters=160, kernel_size=8, strides=4, activation=relu, initializer=glorot	Conv1D Transpose	filters=160, kernel_size=8, strides=4, activation=relu, initializer= he
Conv1D Transpose	filters=128, kernel_size=3, strides=2, activation=relu, initializer=glorot	Conv1D Transpose	filters=128, kernel_size=3, strides=2, activation=relu, initializer= he
Batch normalization		Batch normalization	
Conv1D Transpose	filters=64, kernel_size=5, strides=2, activation=relu, initializer=glorot	Conv1D Transpose	filters=64, kernel_size=5, strides=2, activation=relu, initializer= he
Conv1D Transpose	filters=32, kernel_size=3, strides=2, activation=relu, initializer=glorot	Conv1D Transpose	filters=32, kernel_size=3, strides=2, activation=relu, initializer= he
Batch normalization		Batch normalization	
Conv1D Transpose	filters=32, kernel_size=3, strides=2, activation=relu, initializer=glorot	Conv1D Transpose	filters=32, kernel_size=3, strides=2, activation=relu, initializer= he
Conv1D Transpose	filters=16, kernel_size=3, strides=2, activation=relu, initializer=glorot	Conv1D Transpose	filters=16, kernel_size=3, strides=2, activation=relu, initializer= he
Batch normalization		Batch normalization	
Conv1D Transpose	filters=8, kernel_size=3, strides=2, activation=relu, initializer=glorot	Conv1D Transpose	filters=8, kernel_size=3, strides=2, activation=relu, initializer= he
Conv1D Transpose	filters=8, kernel_size=3, strides=2, activation=relu, initializer=glorot	Conv1D Transpose	filters=8, kernel_size=3, strides=2, activation=relu, initializer= he

Продолжение таблицы 4.4.

Batch normalization		Batch normalization	
Conv1D Transpose	filters=4, kernel_size=3, strides=2, activation=relu, initializer=glorot	Conv1D Transpose	filters=4, kernel_size=3, strides=2, activation=relu, initializer= he
Conv1D Transpose	filters=1, kernel_size=3, strides=2, activation=sigmoid, initializer=glorot	Conv1D Transpose	filters=1, kernel_size=3, strides=2, activation=sigmoid, initializer=glorot

Любая нейронная сеть, может извлекать признаки, отличающиеся от признаков, извлеченных при помощи другой нейронной сети, обученной независимо от первой, даже если обучающие выборки идентичны. Коррелированность признаков, извлекаемых разным нейронными сетями, в данном случае является желательным свойством.

Как известно, для обучения многослойных нейронных сетей необходима выборка большого объема. Однако собранный набор данных состоит всего из 2250 образов, этого явно недостаточно для обучения автокодировщиков и последующего тестирования НПБК. Так как акустический образ уха имеет существенную схожесть с голосовым сигналом, как и их усредненные спектры (рисунки 4.16 б, в, г), то решено использовать следующую схему переноса обучения. Из речевых наборов данных TIMIT и VoxCeleb1 [351] в совокупности было извлечено 71264 голосовых образов дикторов. Размеры образов составляли от 25 до 250 кбайт (при таком размере звуковых файлов их усредненные спектры имеют визуальное сходство с усредненными спектрами эхограмм ушного канала). Эти образы также были преобразованы в усредненные спектры из 2048 амплитуд (с параметрами W_{size}=4096 и W_{step}=2048). При этом использовалось только 4 оконные функции: Хэмминга, Блэкмана, треугольная, прямоугольная. Таким образом, получено 285056 примеров усредненных голосовых спектров, которые использовались для обучения двух автокодировщиков (таблица 4.4). Нейронные сети обучались при использовании алгоритма оптимизации Adam с возрастающим объемом батчей: 256 примеров – 10 эпох, 512 примеров – 10 эпох, 1024 примера – 3 эпохи, 2048 примеров – 3 эпохи, 4096 примеров – 2 эпохи, 8192 примера – 2 эпохи. При этом осуществлялся промежуточный контроль качества восстановления данных (рисунок 4.1д). Применение различных оконных функций

для получения схожих, но отличающихся примеров можно рассматривать как аугментацию данных обучающей выборки [42].

Проведен вычислительный эксперимент по верификации личности испытуемых. Для этого набор акустических данных ушей был разделен случайным образом на две части: 50 испытуемых (100 ушей) «Зарегистрированные субъекты» и 25 испытуемых (50 ушей) «Неизвестные Чужие». Все образы были обработаны обоими кодировщиками. На первом этапе эксперимента верификация личности выполнялась по одному уху (образы левого и правого уха были разделены на два класса, как будто это два разных пользователя, таблица 4.5, 4.6). На втором этапе признаки, извлекаемые из обоих ушей были объединены в один образ (таблица 4.7). Опыты проведены трижды, каждый раз осущесвлялось новое разбиение 50/25 (выбирались новые «Чужие»).

Таблица 4.5 – Вероятности ошибок верификации личности испытуемого по образу одного уха при использовании признаков, извлеченных кодировщиком на базе архитектуры 1 и 2 (при C_+=0,5, C_-=0,5, AUC_{MAX}=0,3).

тип спектров	***L***	***N***	***n***	***η***	**EER_1**	**EER_2**
Блэкмана	128	64	128	4	0,0863	0,08828
Хэмминга	128	64	128	4	0,08747	0,07332
треугольный	128	64	128	4	0,09133	0,08995
прямоугольный	128	64	128	4	0,08122	0,07686
Гаусса	128	64	128	4	0,09047	0,08039
гауссиана параметрическая	128	64	128	4	0,08465	0,07629
Лапласа	128	64	128	4	0,07452	0,08589
прямоугольный + Хэмминга	128	64	256	4	0,04351	0,04272
прямоугольный + Хэмминга + треугольный	256	128	384	4	0,04245	0,04218
прямоугольный + Хэмминга + треугольный + Лапласа + Блэкмана	512	256	640	4	0,04061	0,03812
Все оконные функции	512	256	896	4	0,04031	0,03574

Для обучения каждого НПБК формировались выборки «Свой» и «Чужие». В выборку «Свой» входило от 4 до 8 примеров ($8 \geq K_G \geq 4$) образа конкретного пользователя из множества «Зарегистрированные субъекты», остальные примеры испытуемого использовались для тестирования и определения FRR. Выборка «Чужие» формировалась из примеров образов других испытуемых из множества

«Зарегистрированные субъекты», учитывалось по одному образу каждого уха (K_I=99 на первом этапе эксперимента и K_I=49 на втором). Образы из множества «Неизвестные Чужие» использовались только для тестирования и определения FAR. Результаты эксперимента представлены в таблицах 4.5-4.7.

Таблица 4.6 – Вероятности ошибок верификации личности испытуемого по образу одного уха при объединении признаков, извлеченных обоими кодировщиками (при C_+=0,5, C_-=0,5, AUC_{MAX}=0,3).

тип спектров	***L***	***N***	***n***	***η***	**EER**
Хэмминга	128	64	256	4	0,03591
прямоугольный + Хэмминга	256	128	512	5	0,03823
прямоугольный + Хэмминга + треугольный + Лапласа + Блэкмана	512	256	1280	5	0,03474
Все оконные функции	512	256	1792	5	0,03955
Все оконные функции	1024	512	1792	5	0,0366
Все оконные функции	1024	512	1792	20	0,03834
Все оконные функции	*2048*	*1024*	*1792*	*5*	*0,03466*
Все оконные функции	2048	1024	1792	10	0,03635
Все оконные функции	4096	2048	1792	5	0,03573
Все оконные функции	4096	2048	1792	10	0,04124

Очевидно, что векторы признаков, извлекаемые из усредненных спектров одного и того же сигнала, но на базе разных окон (рисунок 4.16 в, г), должны быть сильно коррелированы (их визуальный анализ таких спектров показал значительное сходство). Использование сильно коррелированных и очень близких по своим значениям признаков обычно не дает каких-либо преимуществ. Однако из таблицы 4.5 можно видеть, что для НПБК на базе корреляционных нейронов это позволяет снизить вероятность ошибки более чем в 2 раза, а также в разы повысить длину связанного ключа за счет появления большего числа сильно коррелированных пар признаков. Также мы можем видеть, что объединение признаков, получаемых с применением немного отличающихся по своей архитектуре кодировщиков (таблица 4.4), также дает положительный эффект (таблица 4.6). Например, объединяя признаки, извлеченные обоими кодировщиками из спектров Хэмминга, можно снизить вероятность ошибки

примерно в 2 раза (таблица 4.5, 4.6). А при объединении признаков, извлекаемых кодировщиками из спектров на базе окна Хэмминга и прямоугольного окна, вероятность ошибок снижается более чем на 10% при одновременном повышении длины ключа в 2 раза (таблицы 4.5, 4.6). Полученные результаты говорят о том, что для повышения производительности НПБК на базе корреляционных нейронов можно применять множество схожих блоков извлечения признаков (однотипные, незначительно отличающиеся нелинейные преобразования по отношению к образу). Такой прием, как правило, не дает столь ощутимого эффекта в сочетании с другими методами машинного обучения (для большинства классификаторов эти наборы признаков будут близки по информационному содержанию). Но НПБК на базе корреляционных нейронов извлекает информацию из корреляционных связей, которые будут отличаться для разных классов образов, благодаря чему, осуществляется снижение вероятностей ошибок. Конечно, предел снижения ошибок существует, однако этот вопрос заслуживает отдельного исследования.

Из таблицы 4.6 можно видеть, что повышать количество нейронов в целом выгоднее, чем количество их входов. Наилучшие результаты для используемого набора данных достигаются при η=5. Из таблицы 4.7 можно заключить, что существенно изменять границы интервалов (C_-; C_+) не стоит, оптимальным является C_-=-0,5, C_+=0,5. Вероятно, это можно объяснить тем, что на малых обучающих выборках «Свой» коэффициенты корреляции рассчитываются с большими погрешностями. При увеличении C_- и C_+ по модулю множество пар коррелированных признаков, несущих информацию, не попадают в заданные интервалы и не используются. При снижении модулей C_- и C_+ появляется множество «ложных» пар, в действительности несущих мало информации. Также существует оптимум параметра AUC_{MAX}≈0,3 (по крайней мере, для используемого набора данных). Слишком большое значение AUC_{MAX} негативно влияет на результат, так как появляется больше нестабильных нейронов, слишком малое значение приводит к тому, что создается малое количество нейронов, в результате приходится расширять интервал (C_- и C_+) и искать новые сочетания признаков для построения нейронов.

Таблица 4.7 – Вероятности ошибок верификации личности испытуемого по двух ушей при объединении признаков, извлеченных обоими кодировщиками для всех оконных функций.

L	N	n	η	EER	C_+	C_-	K_G	AUC_{MAX}
4096	2048	3584	5	0,02865	0,5	-0,5	8	0,3
6144	3072	3584	5	0,02811	0,5	-0,5	8	0,3
8192	4096	3584	7	0,02956	0,5	-0,5	8	0,3
8192	4096	3584	6	0,02584	0,5	-0,5	8	0,3
8192	*4096*	*3584*	*5*	*0,02552*	*0,5*	*-0,5*	*8*	*0,3*
8192	4096	3584	4	0,02561	0,5	-0,5	8	0,3
8192	4096	3584	3	0,0273	0,5	-0,5	8	0,3
10240	5120	3584	5	0,02729	0,5	-0,5	8	0,3
12288	6144	3584	5	0,02725	0,5	-0,5	8	0,3
8192	4096	3584	5	0,03025	0,5	-0,5	8	0,4
8192	4096	3584	5	0,02878	0,5	-0,5	8	0,35
8192	4096	3584	5	0,02703	0,5	-0,5	8	0,25
8192	4096	3584	5	0,03673	0,5	-0,5	8	0,2
8192	4096	3584	5	0,02619	0,3	-0,3	8	0,3
8192	4096	3584	5	0,026	0,4	-0,4	8	0,3
8192	4096	3584	5	0,02563	0,45	-0,45	8	0,3
8192	4096	3584	5	0,02754	0,55	-0,55	8	0,3
8192	4096	3584	5	0,02855	0,6	-0,6	8	0,3
8192	4096	3584	5	0,03274	0,7	-0,7	8	0,3
8192	4096	3584	5	0,02785	0,5	-0,5	7	0,3
8192	***4096***	***3584***	***5***	***0,0238***	***0,5***	***-0,5***	***6***	***0,3***
8192	4096	3584	5	0,03868	0,5	-0,5	5	0,3
8192	4096	3584	5	0,04098	0,5	-0,5	4	0,3

Снижение объема выборки «Свой» не очень сильно влияет на вероятность ошибок (таблица 4.7). Модель хорошо обучается на 8, 7 и 6 примерах. Таким образом, использование больших выборок для обучения НПБК на базе корреляционных нейронов не требуется.

Из рисунка 4.17 видно, что если на выходе НПБК исправлять 16% ошибочных бит (что можно сделать с помощью кодов Безяева А.В. [10, 12, 314], как это упоминалось выше), то можно настроить систему аутентификации на следующие показатели вероятности: FAR<0,0001 при FRR=0,093, что в целом может быть удовлетворительным для практических целей. Коррекция неверных бит ключа может выполняться классическими методами помехоустойчивого кодирования (но в отличие от нечетких экстракторов, это не будет влиять на длину ключа). Полученные показатели, вероятно, не являются предельными, так

как можно попробовать улучшить результат, добавив другие оконные функции или несколько автокодировщиков с иной архитектурой (в этой работе не ставились подобные цели). В любом случае предлагаемые модели являются весьма перспективными для дальнейшей разработки.

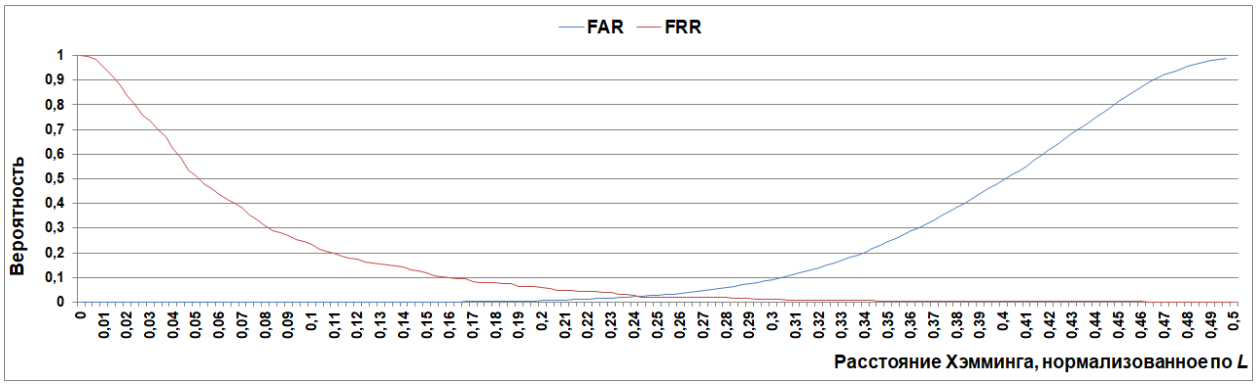


Рисунок 4.17 – Результаты верификации личности субъектов по двум ушам при C_+=0,5, C_-=0,5, AUC_{MAX}=0,3, K_G=6, N=4096, η=5.

Сравнительные показатели по существующим методам распознавания личности на основе уха представлены в таблице 4.8.

Таблица 4.8 – Достигнутые результаты в области распознавания личности по параметрам уха

Метод извлечения признаков	Метод классификации	Набор данных	Надежность, %
2D изображения уха			
Фильтр Габора	Модифицирован-ный AdaBoost	59 испытуемых по 6 образов, всего 354 образов	EER = 4 [359]
Свёрточные нейронные сети	Метод опорных векторов	60 испытуемых по 3 образа, всего 180 образов	EER = 4,2 [274]
Speeded Up Robust Features, SURF	Мера Евклида, метод k ближайших соседей	300 испытуемых по 7 образов, всего 2066 образов	EER = 2,5 [311]
Метод главных компонент + Метод силовых линий	Расстояние Евклида, k-NN	63 субъекта по 4 изображения, всего 252 изображения	EER = 13,5 [270]
Логарифмический фильтр Габора (Log-Gabor Filter)	Расстояние Хэмминга, k-NN	113 субъектов, 265 изображений тренировочной выборки, 185 изображений тестовой выборки	EER = 10,5 [270]
Метод активного контура	Расстояние Хэмминга, k-NN	28 субъектов по 145 изображений, всего 4060 изображений	EER = 5,6 [270]
3D изображения уха			
Surface patch histogram of indexed shapes	Метод k ближайших соседей	500 испытуемых по 4 образа, всего 2000 образов	EER = 2,2 [362]
Key Point Detection	Итеративный алгоритм ближайших точек	415 испытуемых по 2 образа, всего 830 образов	EER = 4,1 [223]
Locale Surface Variation			EER = 2,3 [172]

Продолжение таблицы 4.8.

Локальное поверхностное исправление (Local Surface Patch, LSP)	Средняя квадратичная ошибка (Root Mean Square, RMS)	302 субъекта, всего 942 изображения		EER = 2,3 [201]
Собственный метод	Итеративный алгоритм ближайших точек (Iterative Closest Point, ICP)	415 субъектов, всего 1386 изображений		EER = 1,2 [355]
Акустические образы уха				
Фильтр на основе метода перевалов	Определение коэффициента корреляции между эталоном и образом	Мобильный телефон: 17 испытуемых по 8 образов		5,5≤EER≤18 [182]
		Вставные наушники: 31 испытуемый по 8 образов		1≤EER≤6 [182]
		Накладные наушники: 31 испытуемый по 8 образов		0,8≤EER≤8 [182]
Оконное преобразование Фурье	k ближайших соседей	20 испытуемых по 600 образов, всего 11900 образов		5≤FRR≤10 7≤FAR≤15 [222]
	Деревья решений			9≤FRR≤15,2 9≤FAR≤14,5 [222]
	«Наивный» Байес			3≤FRR≤8 9≤FAR≤27,5 [222]
	Многослойный персептрон			3≤FRR≤9,8 4≤FAR≤8 [222]
	Машина опорных векторов			4≤FRR≤7 3≤FAR≤7 [222]
Авторская технология EarEcho				EER=4,84 [222]
MFCCs, LDA	Косинусное подобие	25 испытуемых		EER=4,47 [284]
Нечёткий экстрактор + метод k ближайших соседей		45 субъектов по 8 замеров		FRR = 0,8 FAR = 0,7 [313]
		65 субъектов по 8 замеров		FRR = 2,4 FAR = 4,4 [313]
Кепстрограммы	«Наивный» Байес	AIC-ears-75, собранная в рамках настоящего исследования 75 испытуемых по 15 образов для каждого уха, всего 2250 образов	Обучающая выборка «Свой» из 8 примеров	EER=0,53 FRR=10,28 при FAR<0,01
	Сверточные нейронные сети			EER=2,85
Усредненный спектр	Полносвязные неглубокие нейронные сети			EER=2,66
Усредненный спектр, обучение автокодировщиков на голосовых образах	Классический НПБК, обучаемый по ГОСТ Р 52633.5-2011			EER=3,136 FRR=23,42 при FAR<0,01
	НПБК на базе корреляционных нейронов		Обучающая выборка «Свой» из 6 примеров	EER=2,38 FRR=9,3 при FAR<0,01

Как можно видеть, даже при использовании относительно слабых блоков извлечения признаков на базе автокодировщиков, в основе которых лежат одномерные свертки, и которые обучены на образах иного типа (голос вместо эхограмм уха), НПБК показывает достаточно высокие результаты.

Разработанный метод позволяет реализовать концепцию защищенного исполнения нейросетевых алгоритмов искусственного интеллекта в приложениях

биометрии, используя идею, так называемой, аннулируемой биометрии [317]. Ее суть состоит в устранении существенного недостатка классической биометрии, связанного с невозможностью изменять физиологический биометрический признак в случае его компрометации.

Распознавание личности по параметрам внутреннего строения уха является достаточно новым направлением. Геометрическое строение уха человека не является динамическим биометрическим образом (как рукописный или голосовой пароли), поэтому нельзя изменить строение уха. Однако акустический образ уха, связанный с НПБК все равно может быть изменен пользователем по собственному усмотрению. Акустический образ зависит от ключевого стимула – звукового сигнала $\upsilon(t)$, который распространяется в ушном канале. При изменении зондирующего сигнала $\upsilon(t)$, эхограмма ушного канала меняется кардинально. Изменив параметры $\upsilon(t)$, пользователь может заменить биометрический образ, связанный с криптографическим ключом или паролем. Если $\upsilon(t)$ рассматривать как секретную информацию, известную только легитимному субъекту (по аналогии с паролем), то сигналы $\upsilon(t)$ и $u_{i,k}(t)$ в совокупности будут являться идентификатором и аутентификатором. Такой подход потенциально может усилить защитные свойства предлагаемого метода аутентификации по акустическим образам уха.

Даже при известном ключевом стимуле, вероятность «ложного допуска» по результатам проведенных в главе 4 экспериментов существенно ниже 0,01% (фактически ошибок 2-го рода зарегистрировано не было, но в биометрии не существует нулевых вероятностей ошибок, поэтому такая оценка дана в 4 главе, исходя из количества проведенных опытов). Таким образом, при неизвестном злоумышленнику ключевом стимуле вероятность FAR снизится на порядки. К примеру, если предоставить пользователю выбор из 100 возможных ключевых стимулов при аутентификации, то количество ошибок «ложного допуска» составит уже менее 0,001%, так как взломщику придется перебирать в 100 раз больше биометрических образов для получения допуска.

Следуя простым расчетам можно примерно обозначить возможное количество ключевых стимулов, если использовать для их синтеза ЛЧМ. Пусть звуковой 10 секундный сигнал, играющий роль стимула, можно разделить на участки длительностью от 1 до 10 секунд, на каждом из которых частота сигнала либо возрастает, либо убывает в диапазоне от 1 кГц до 14 кГц. В этом случае количество стимулов равно 2^{10}=1024, соответственно каждый стимул можно описать числом или двоичным 10-ти битным кодом. Например, используемый в главе 4 стимул можно записать как 1111100000 (где, «1» свидетельствует о возрастании частоты, «0» свидетельствует о ее убывании). При таком количестве стимулов несложно рассчитать: FAR<0,0001%, при этом FRR для каждого пользователя и стимула может незначительно разниться. С достоверностью 99% можно утверждать, что не стоит ожидать значительных отклонений (более чем на ±3%) реальной частоты ошибок «ложного отказа» от полученной оценки FRR=9,3%. При доверительном интервале 0,03 и количестве опытов 675 (75 человек совершило по 9 попыток входа) доверительная вероятность того, что оценка вероятности «ложного отказа» не выйдет за пределы интервала [0,063; 0,123] составляет более 0,99. Для FAR≈0,0001 доверительная вероятность составляет 0,96 при доверительном интервале 0,00009 (75·15·50=56250 опытов).

Также проведен аналогичный вычислительный эксперимент, но с использованием модели НПБК, обучаемой по ГОСТ Р 52633.5. Наилучшими результатами, полученными с помощью ГОСТ Р 52633.5, является следующий: EER=0,03136 (FRR=0,2342 при FAR<0,0001) при длине ключа L=716 и объеме обучающих выборок «Свой» K_G = 8 и «Чужие» K_I=49.

Как можно видеть, предложенная модель НПБК на основе корреляционных нейронов превосходит базовую модель ГОСТ Р 52633.5-2011 по длине ключа более, чем в 10 раз (716 бит против 8192 бит, таблица 4.7), а также по точности (EER=0,0238 против EER=0,03136). Однако следует также оценить энтропию кодов, генерируемых НПБК при поступлении на его входы образов «Чужих». Это важный показатель, так как энтропия связана с FAR: чем ниже FAR, тем выше энтропия [161] и тем меньше утечек конфиденциальности знаний НПБК.

Приблизительную оценку энтропии E без учета коррелированности разрядов выходов НПБК можно получить, вычислив собственную информацию события “ложного допуска”:

$$E(\mathrm{FAR}) \approx -\log_2 \mathrm{FAR}$$

Точный расчет многомерной энтропии длинных бинарных последовательностей прямым численным экспериментом является технически нерешаемой задачей, как и расчет сверхнизких показателей FAR. Чтобы получить E(FAR)=256 бит, требуется, чтобы FAR<10^{-77}. Для проверки такой экстремально низкой вероятности прямым численным экспериментом не хватит населения планеты. Однако можно прибегнуть к грубому расчету так называемой точечной энтропии для НПБК, вычисляемой по формуле [114]:

$$H("a_{001}, a_{002},...., a_L") = -\log_2 \left\{ \frac{1}{\sigma("b") \cdot \sqrt{2\pi}} \int_0^1 \exp\left(\frac{-(m("b") - x)^2}{2(\sigma("b"))^2} \right) \cdot dx \right\}$$

где $m(.)$ и $\sigma(.)$ – функции расчета математических ожиданий и среднеквадратичных отклонений.

Также об энтропии кодов на выходе НПБК косвенно можно судить по другой метрике – средней стабильности ответов НПБК на образы «Чужих»:

$$\gamma_k = \sum_{l=1}^{L} 2\left|P_l(1) - 0{,}5\right|, \qquad (4.2)$$

где k – номер мспытуемого («Чужого»), L – длина ключа , l – номер нейрона, $P_l(1)$ – вероятность (или относительная частота) появления “единицы” (можно заменить на $P_l(0)$) в l-м разряде ответа НПБК на примеры образа k-го “Чужого”. Оценка относительных частот $P_l(1)$ и $P_l(0)$ может проводиться на основании нескольких образов испытуемого (в настоящей работе использовалось по 10 примеров от каждого «Чужого»).

Вычислены показатели стабильности ответов НПБК на образы «Чужих». По результатам эксперимента для классической модели НПБК показатель стабильности (4.2) для каждого “Чужого” оказался зависим от среднего количества ошибочных бит ключа (рисунок 4.18а). Таким образом, даже в отсутствии явной индикации близости выходов НПБК к ключу пользователя

хакер может осуществить направленный перебор синтетических образов, скрещивая примеры рукописных паролей разных “Чужих”, которые дают наиболее стабильный ответ (этот недостаток нивелируется применением спецификации [162]). Из рисунка 4.18а также видно, что образы “Чужие” обладают, так называемой, *«симметрией» стабильности ответов относительно образа “Свой”* (свойством «симметрии»). Это означает, что стабильность ответов НПБК при предъявлении образов «Чужих» возрастает, но не только если ответы близки (в метрике Хэмминга) к ключу пользователя, но и если они близки к инверсии ключа (инверсный код возникает, если все биты ответа ПБК являются ошибочными). Инверсный код можно обратить и получить ключ пользователя. Данное свойство позволяет ускорить процедуру направленного перебора биометрических образов в 2 раза (осуществляя одновременно поиск наиболее близкого и наиболее дальнего образа «Чужого» относительно образа «Свой»).

Для предложенной модели НПБК на базе корреляционных нейронов ситуация несколько иная. Средние показатели стабильности выходов НПБК при поступлении образов “Чужих” ниже ($m(\gamma)$=0,24) и можно видеть, что свойство «симметрии» отсутствует (рисунок 4.18б).

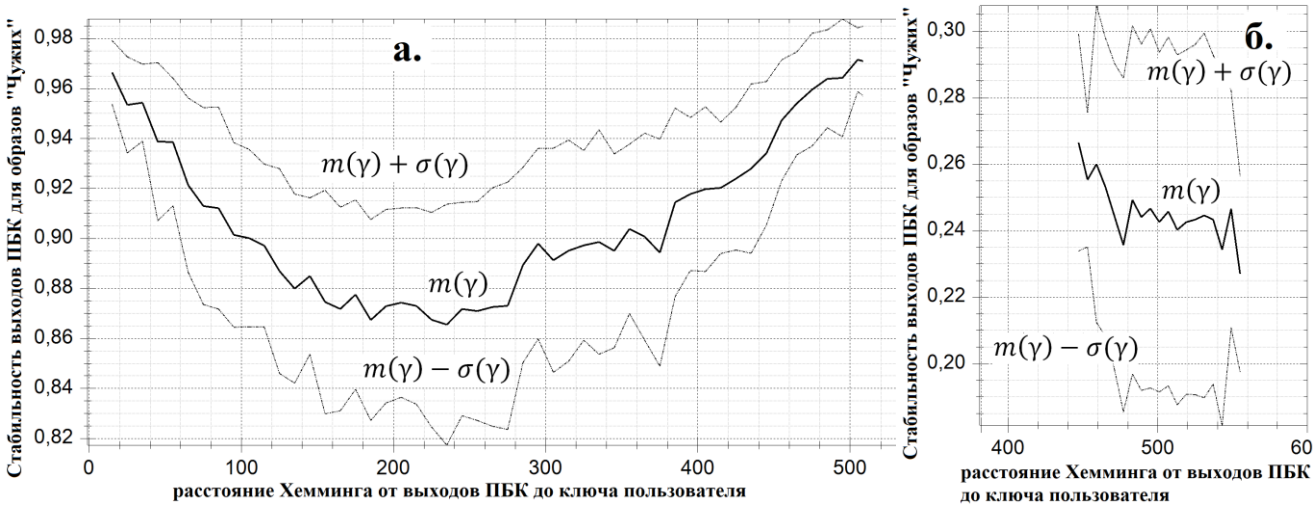


Рисунок 4.18 – Стабильность битовых последовательностей на выходе НПБК при поступлении на их входы образов «Чужих»: а. классическая модель НПБК, обучаемая по ГОСТ Р 52633.5-2011 (без применение дополнительной криптографической защиты в соответствии со спецификацией [162]); б. предложенная модель НПБК на базе корреляционных нейронов.

Таким образом, есть все основания полагать, что дополнительная криптографическая защита таблиц нейросетевых функционалов корреляционных нейронов не потребуется (точное подтверждение этого вывода находится в компетенции криптографов и выходит за рамки настоящего исследования).

При необходимости битовая последовательность, возникающая на выходе НПБК, может быть преобразована в ключ нужной длины с применением криптографической хеш-функции (после исправления ошибочных бит).

4.7 Формирование набора данных рукописных и голосовых образов

Как уже упоминалось ранее в главе 3, оцифрованный рукописный образ состоит из функций координат *x_coord(t)*, *y_coord(t)* и давления (силы нажатия) пера на планшет *pressure(t)*, где *t* – это время в дискретной форме, как показано на рисунке 4.19. Помимо базовых функций вычисляется также функция скорости пере на планшете $v_{xy}(t)$. Для устройств начального уровня поддержки давления нет (только индикация касания). В ISO/IEC 19794-7 предусмотрена возможность учета угла наклона и азимута пера. Эти данные регистрируются профессиональными устройствами, которые не имеют широкого распространения.

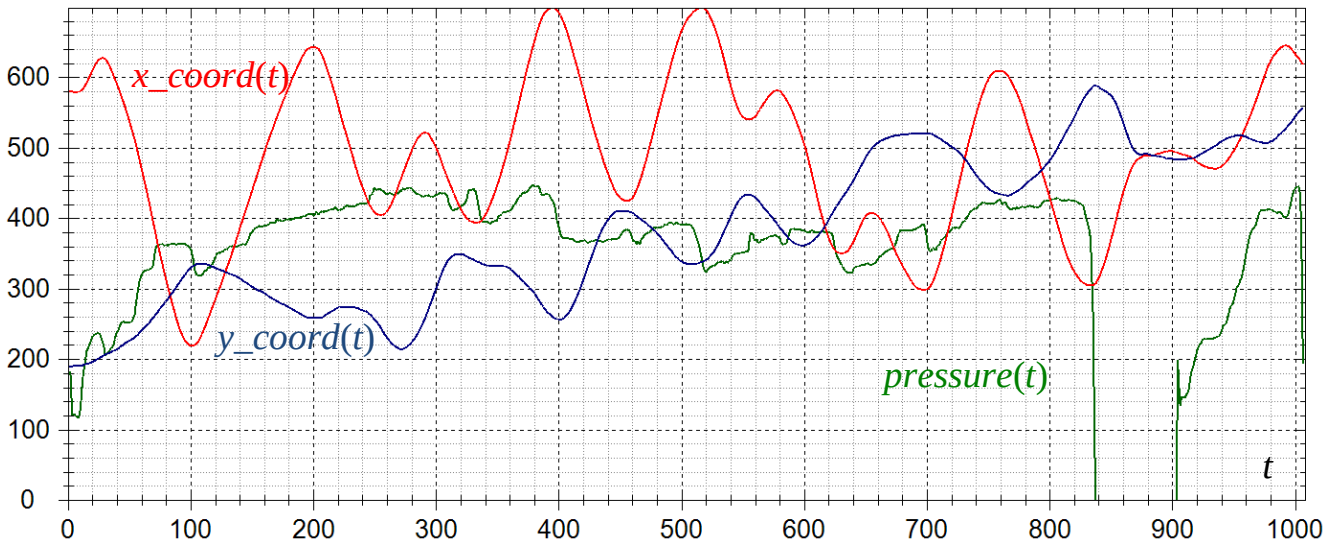


Рисунок 4.19 – Пример рукописного образа

Голосовой пароль в компьютерном представлении – это звуковой моно сигнал *voice*(*t*), как показано на рисунке 4.20, записанный в файл формата wav (без сжатия). Мнения относительно параметров оцифровки сигнала (с сохранением информации о дикторе), в разных работах расходятся. Наиболее информативные признаки диктора можно получить путем оценки индивидуальной фундаментальной частоты его голоса (частоты основного тона, ЧОТ), которая обусловлена строением гортани и проявляется при произношении гласных фонем. У мужчин ЧОТ варьируется от 80 до 150 Гц, у женщин – от 120 до 400 [39]. Однако информацию о дикторе несут и более высокочастотные составляющие (обертоны). Частоты, отвечающие за разборчивость речи, сконцентрированы в основном в диапазоне 300 и 3400 Гц. Поэтому считается, что речь человека при F=8 кГц различима. В [39] сообщается, что при вейвлет-анализе речи удается получить хорошее частотно-временное разрешение при F=8 кГц. Такая частота характерна для низко информативных каналов передачи данных, таких как телефонная сеть. Тем не менее, обычно используется частота дискретизации F=16 кГц [325] или выше [1, 339] (при увеличении F уменьшаются ошибки квантования). Касательно уровней квантования существенных расхождений нет – размер семпла в 16 бит считается достаточным для «машинного восприятия» [1, 39, 339].

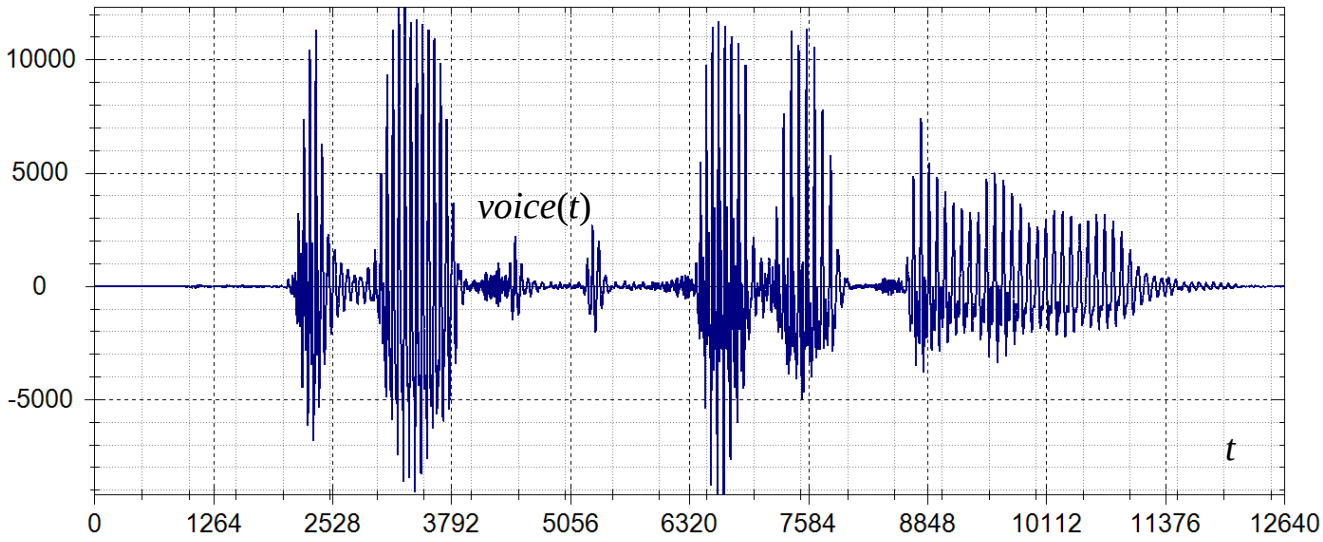


Рисунок 4.20 – Пример голосового образа

Не все устройства могут обеспечить качественную оцифровку звука. Встроенные в мобильные гаджеты средства (низкого ценового сегмента) не всегда способны охватить даже диапазон частот 80-3400 Гц. Для более качественных «голосовых» микрофонов среднего уровня реальный диапазон частот записи обычно составляет от 70 Гц до 12 кГц. По теореме Котельникова достаточно ограничиться F=24 кГц.

В открытых источниках не было найдено наборов данных голосовых и рукописных паролей, соответствующих всем требованиям репрезентативности [1]. Поэтому в данной работе собрана проприентарная обезличенная база рукописных и голосовых паролей [329], которая была долнена данными из открытых источников. Открытые голосовые базы [341] в основном состоят из фрагментов произвольной речи (для тестирования FRR и обучения нужны примеры произношения пароля или фразы). Открытые базы рукописных образов [174] разнородны (разные устройства ввода, мало данных динамики и давления пера, не достаточно примеров «Свой»). Поэтому из открытых источников взяты примеры (с параметрами записей не ниже используемых в работе), но только для расширения тестовой выборки «Неизвестные Чужие». В качестве основных источников использованы: TIMIT, VoxCeleb1 [311], VoxForge [http://www.voxforge.org] (русский набор включает 630 дикторов), обезличенная база рукописных образов веб-сервиса SignToLogin [277].

Рукописные образы формировались с использованием планшета Wacom (частота опроса 200 Гц, 1024 уровня давления), голосовые – с использованием микрофонов Pioneer, Sony (диапазон частот 70-12000 Гц).

В соответствии с ISO/IEC 19795-3 при формировании базы были учтены следующие факторы:

- “старение эталона”: база включает рукописные и голосовые образы испытуемых, полученные в разные дни (с интервалом до нескольких недель), а также образы, полученные в различных ПФС;

- пол и возраст испытуемых из базы «Свои» распределены равномерно на интервале от 18 до 35 лет;

- усилия злоумышленника: образ пароля каждого испытуемого пытались повторить еще пять субъектов по 2 раза, предварительно изучив внешний вид рукописного пароля на экране монитора и имея представление о темпе почерка его владельца, для голосовых паролей проводилось однократное заслушивание. Согласно ISO/IEC 19795-3 такой вид подделки соответствует 4 степени фальсификации.

Всего для экспериментов подготовлено:

- выборка «Зарегистрированные субъекты» («Все Свои»): 260 подписантов и 260 дикторов, каждый воспроизвел образ своего рукописного и голосового паролей по 90 раз, данные собраны в три этапа с интервалом в несколько недель, на каждом этапе испытуемый ввел по 30 примеров. На первых двух этапах испытуемые пребывали в нормальном ПФС, на третьем – в сонном;

- выборка «Неизвестные Чужие»: 6500 примеров воспроизведения рукописных и 6500 примеров голосовых образов (фраз, паролей), воспроизведенных другими субъектами, не вошедшими в базу «Зарегистрированные субъекты» (в том числе, заимствованные из открытых источников);

- выборка «Подделки»: 2600 подделок рукописных паролей и 2600 подделок голосовых паролей (когда злоумышленник знал, какую фразу произносить).

4.8 Извлечение признаков из голосовых и рукописных паролей и оценка их информативности

Наиболее распространенным методом получения признаков рукописного образа является дискретное преобразование Фурье, позволяющее выделить амплитудно-частотные параметры, характеризующие «поведение» сигнала за все время его существования. Преобразование Фурье не учитывает локализацию частот во времени, поэтому его используют для анализа стационарных сигналов. Рассматриваемые функции не являются стационарными сигналами и могут

содержать особенности, проявляющиеся на определенном участке, которые часто являются достаточно информативными с точки зрения распознавания образов, но не могут быть обнаружены путем спектрального анализа с использованием Фурье. Быстрое оконное преобразование Фурье (short time fourier transform, STFT) обычно используется для более длинных сигналов, где важным показателем является производительность, например, звуковых. При анализе коротких сигналов вейвлет-анализ дает преимущества [340].

В качестве кандидатов на признаки рассматривались детализирующие коэффициенты, полученные при разложении биометрических сигналов с помощью быстрого алгоритма Малла по следующим базисам. В качестве базисов использовались следующие: Хаара, Добеши (D4, D4, D6, D8, D10, D12, D14, D16, D18, D20), Симлета (порядков с 4 по 10), Койфлет (порядков 6, 12, 18, 24, 30), Морле, Майера, Шенона [289]. Кратко рассмотрим особенности указанных базисов, подходящих для быстрого вейвлет-преобразования.

Преобразование Хаара часто используется для сжатия входных сигналов, компрессии изображений, в основном цветных и черно-белых с плавными переходами (например, рентгеновских снимков). Вейвлеты Добеши обладают следующим свойством: аппроксимирующие и детализирующие коэффициенты обладают двойной избыточностью. Симлеты отличаются от базисов Добеши, начиная с четвертого порядка. При разложении по данным базисам наблюдается большее «размытие» сигнала, по сравнению с вейвлетами Добеши. Койфлеты являются частным случаем вейвлетов Добеши с нулевыми моментами скейлинг-функции. Вейвлеты Морле отличаются тем, что являются комплексными функциями, у которых вещественные и мнимые части – это модулированные гауссианой гармоники.

Для подписи и рукописного пароля диапазон анализируемых частот составляет 0,1-10 Гц [76]. По результатам анализа установлено, что наиболее информативными являются детализирующие вейвлет коэффициенты по базису Хаара, полученные на 4 уровнях разложения, соответствующих самым низким анализируемым частотам, эти признаки решено использовать.

В задаче идентификации диктора, как правило, наиболее информативные признаки характеризуют ЧОТ. Однако ЧОТ вариативна. Для ее оценки применяются методы вейвлет-анализа [39], анализа периодичности автокорреляционных функций [58], частоты экстремумов сигнала (переходов через «ноль» [246]). Также для поиска признаков анализируют спектры и кепстры сигнала. Для этого принято использовать кратковременное быстрое преобразование Фурье (STFT). Распространен подход с расчетом мел-кепстральных коэффициентов (MFCC) [325] или логарифмических энергий (MFEC) [346]. Вейвлет-анализ редко применяется в задачах биометрической аутентификации по голосу из-за низкой скорости работы (голосовой образ содержит в десятки и даже сотни раз больше отчетов, чем рукописный). В [329] установлено, что использование вейвлет параметров не дает ощутимых преимуществ в задаче аутентификации диктора по голосу по сравнению с другими признаками, вычисляемыми на основе Фурье-анализа.

В настоящей работе использована компиляция нескольких подходов к вычислению признаков рукописных и голосовых образов (таблица 4.9), включая спектральный, корреляционный и вейвлет-анализ. Апробировано 2 набора рукописных признаков: для устройств с поддержкой давления (N_{HXYP}=782) и без (N_{HXY}=521), и 2 набора признаков голосовых образов: для устройств с хорошим качеством записи звука (частота дискретизации 24 кГц или выше, N_{V24}=570) и с низким качеством записи звука (8 кГц, N_{V8}=350). Подробнее об используемых методиках вычисления признаков, представленных в таблице 4.9, можно ознакомиться в публикациях [23, 36, 37, 84, 113, 144, 146, 147, 148, 171].

Опираясь на имеющиеся знания о задачах верификации образов, введем условные (обобщенные) уровни информативности признаков (таблица 4.10), по аналогии со шкалов Чеддока для корреляционной зависимости. Сверхинформативные признаки не нуждаются в дополнительном обогащении, т.е. какие-либо функционалы для их интегрирования не требуются, т.к. можно верифицировать образ непосредственно по данным плотностей вероятности либо функций распределения значений одного признака. Для верификации образов

привысокоинформативных признаках можно воспользоваться почти любыми мерами близости (можно обойтись без ИНС или других сложных моделей ИИ). Весьма информативные признаки нуждаются в более существенном обогащении, и для их обработки уже требуется хотя бы однослойная нейросеть. Признаки со средней информативностью могут обрабатываться большими нейронными сетями или нейро-иммунными моделями ИИ на базе корреляционных, классических, квадратичных или иных нейронов в зависимости от специфики задач. Чем меньше число доступных признаков – тем больше требуется типов нейронов, т.е. недостаток информации может компенсироваться количеством способов ее обработки. По мере снижения информативности признаков может возрастать число слоев и/или нейронов в слое, а также количество типов нейронов. На данный момент не найдено архитектур нецронов, которые могли бы принимать надежные решения при наличии почти не информативных признаков (I_{bit}<0,001) [140, 159, 331, 334].

Таблица 4.9. Краткое описание выбранных методик извлечения признаков, используемых для биометрической аутентификации

№	Краткое описание группы признаков (1 – рукописный, 2 – голосовой пароль)	Число признаков
1.1	образ делится на 16 равных по числу точек отрезков, строится матрица расстояний между их краями в 2-х и 3-х мерном пространстве (*pressure*(*t*) – третье измерение)	240 120 – без *p*(*t*)
1.2	вычисление коэффициентов корреляции между *x_coord*(*t*), *y_coord*(*t*), *pressure*(*t*), *x_coord'*(*t*), *y_coord'*(*t*), *pressure'*(*t*) и функцией скорости пера $v_{xy}(t)$, производной от *x_coord*(*t*), *y_coord*(*t*)	21 10 – без *p*(*t*)
1.3	вычисление параметров внешнего вида образа – угол наклона, отношение длины к ширине, центр в 2-х (3-х) мерном пространстве, описываемый 2 или 3 координатами	5 4 – без *p*(*t*)
1.4	вычисление средних значений фрагментов функций *pressure*(*t*), *x_coord'*(*t*), *y_coord'*(*t*), $v_{xy}(t)$ (образ делится на 5 равных по числу точек отрезков)	20 15 – без *p*(*t*)
1.5	вычисление детализирующих коэффициентов быстрого вейвлет преобразования Хаара (алгоритм Малла), полученных на 4 уровнях разложения (низкие частоты) для *x_coord*(*t*), *y_coord*(*t*), *pressure*(*t*), $v_{xy}(t)$ (функции сначала приводились к 128 отчетам (интерполяция))	240 180 – без *p*(*t*)
1.6	вычисление усредненного амплитудного спектра с помощью STFT (размер окна – 128 отчетов, шаг – 16 отчетов) для *x_coord*(*t*), *y_coord*(*t*), *pressure*(*t*), $v_{xy}(t)$	256 192 – без *p*(*t*)

Продолжение таблицы 4.9

2.1	вычисление части усредненного по всем окнам амплитудного спектра речевого сигнала (коэффициенты нижних частот), вычисленного с помощью STFT (размер окна – 2048(512) при $_F$=24 (8) кГц, шаг – 16). Предварительно речевой сигнал нормируется по энергии, удаляется тишина.	40 – $_F$=24 кГц 20 – $_F$=8 кГц
2.2	вычисление коэффициентов нижних частот кепстра, который берется от полного усредненного по всем окнам амплитудного спектра, получаемого в соответствии с 2.1	40 – $_F$=24 кГц 20 – $_F$=8 кГц
2.3	вычисление коэффициентов нижних частот кепстра, который берется от полного логарифмированного усредненного амплитудного спектра, получаемого в соответствии с 2.1	40 – $_F$=24 кГц 20 – $_F$=8 кГц
2.4	вычисление кепстра второго порядка от полных кепстров 2.1 и 2.2 (кепстры 2.1 и 2.2 повторно подвергаются прямому преобразованию Фурье)	256 – $_F$=24 кГц 128 – $_F$=8 кГц
2.5	подсчет частоты переходов сигнала через нулевое деление окном (размер окна – 2048(512) при $_F$=24 (8) кГц, шаг – 16), грубо характеризуют ЧОТ (нулевую форманту)	64 – $_F$=24 кГц 32 – $_F$=8 кГц
2.6	вычисление полного амплитудного спектра функции автокорреляции речевого сигнала	128
2.7	вычисление частоты переходов через «ноль» и экстремумов функции автокорреляции	2

Таблица 4.10. Предлагаемая шкала информативности признаков

$AUC(\Phi_G(a_j), \Phi_I(a_j))\leq 0{,}001$	$9{,}966\leq I_{bit}$	Сверхинформативные
$0{,}001< AUC(\Phi_G(a_j), \Phi_I(a_j))\leq 0{,}1$	$3{,}322< I_{bit} \leq 9{,}966$	Высокоинформативные
$0{,}1< AUC(\Phi_G(a_j), \Phi_I(a_j))\leq 0{,}3$	$1{,}737< I_{bit} \leq 3{,}322$	Весьма информативные
$0{,}3< AUC(\Phi_G(a_j), \Phi_I(a_j))\leq 0{,}5$	$1< I_{bit} \leq 1{,}737$	Средней информативности
$0{,}5< AUC(\Phi_G(a_j), \Phi_I(a_j))\leq 0{,}7$	$0{,}515< I_{bit} \leq 1$	Малоинформативные
$0{,}7< AUC(\Phi_G(a_j), \Phi_I(a_j))\leq 0{,}9$	$0{,}152< I_{bit} \leq 0{,}515$	Крайне малоинформативные
$0{,}9< AUC(\Phi_G(a_j), \Phi_I(a_j))\leq 0{,}999$	$0{,}001< I_{bit} \leq 0{,}152$	Почти не информативные
$0{,}999< AUC(\Phi_G(a_j), \Phi_I(a_j))$	$I_{bit}<0{,}001$	Неинформативные

Признаки из таблицы 4.9 имеют закон распределения, близкий к нормальному, реже к Лапласа или логнормальному, что проверялось на основании критерия хи-квадрат. Как можно видеть из таблицы 4.11 динамические биометрические образы по большей части (72,5%) состоят из малоинформативных признаков, также они включают существенную часть признаков средней информативности и незначительное количество весьма информативных, а также крайне малоинформативных признаков. Для сравнения

можно видеть, что образ лица человека (статический биометрический образ) содержит гораздо больше информации, т.к. имеет высокоинформативные признаки и существенную часть весьма информативных параметров [159]. Наиболее информативные статические образы (отпечаток пальца, радужка, трехмерные модели лица и черепа) состоят преимущественно из высокоинформативных признаков.

Таблица 4.11. Распределение биометрических признаков по уровням информативности

Уровень информативности	Подпись и рукописные пароли	Голос	Клавиатурный почерк	Динамические образы	Лицо
$I \leq 0{,}001$	0%	0%	0%	0%	0%
$0{,}001 < I \leq 0{,}1$	0%	0%	0%	0%	3%
$0{,}1 < I \leq 0{,}3$	1%	3%	1%	2,5%	13%
$0{,}3 < I \leq 0{,}5$	45%	9,5%	5%	23%	53%
$0{,}5 < I \leq 0{,}7$	50%	87%	84%	72,5%	31%
$0{,}7 < I \leq 0{,}9$	4%	0,5%	10%	2%	0%
$0{,}9 < I \leq 0{,}999$	0%	0%	0%	0%	0%
$0{,}999 < I$	0%	0%	0%	0%	0%

4.9 Биометрическая аутентификация по голосовым и рукописным паролям с обеспечением устойчивости к дрейфу биометрических данных

Проведен эксперимент по распознаванию личности испытуемых на основе рукописных и голосовых паролей с использованием предложенной адаптивной нейро-иммунной модели ИИ. Для каждого из 260 испытуемых из выборки «Зарегистрированные субъекты» создавалась адаптивная модель ИИ, которая обучалась:

- на 15 примерах «Свой» соответствующего испытуемого (по алгоритму на рисунок 3.10), полученных в нормальном ПФС;
- на 259 примерах «Чужой» из этой же выборки (каждый из 260 испытуемых является «Чужим» по отношению ко всем остальным).

Для оценки FRR использовались образы испытуемых из выборки «Зарегистрированные субъекты», не вошедшие в обучающую выборку (т.е. по 80 примеров на испытуемого).

Для оценки FAR использовался подход, на котором базируется ГОСТ Р 52633.3-2011. Согласно стандарту в эксперименте необходимо использовать не только естественные образы “Чужой”, но и синтетические, генерируемые на основе скрещивания естественных (по методике ГОСТ 52633.2-2010). Если для естественных образов «Чужих» не наблюдается ошибок 2-го рода («ложного совпадения ключа»), то естественные образы скрещиваются. При скрещивании следует выбирать пары из “Чужих”, которые дают ответы ПБК, наиболее близкие в метрике Хэмминга к ключу пользователя. Далее по аналогичному принципу могут скрещиваться синтетические образы. Каждая новая популяция синтетических образов «Чужих» все ближе к образу “Свой”. Процесс тестирования ПБК прекращается, когда для очередной популяции будут зафиксированы ошибки 2-го рода или синтезировано определенное число популяций «Чужих». В процессе тестирования сокращается количество попыток предъявления конкурирующих примеров, а точность оценки FAR увеличивается на порядки при сохранении статистической значимости.

В проведенном эксперименте применялся данный подход, но вместо метрики Хэмминга использовалась метрика среднего реакций детекторов нейро-иммунной модели ИИ:

$$\ddot{u} = \Phi(\overline{D}^* = \{D_1^*, \ldots D_N^*\}, \overline{a}) = \frac{1}{N} \cdot \sum_{i=1}^{N} \phi(D_i^*, \overline{a}) = \frac{1}{N} \cdot \sum_{i=1}^{N} u_i$$

В качестве тестовых выборок использовались выборка «Неизвестные Чужие» и выборка «Подделки» (для второй выборки FAR вычислялась как отношение количества неверных решений к общему числу опытов, так как для полноценного тестирования в соответствии с ГОСТ Р 52633.3-2011 было недостаточно данных). При двухфакторной аутентификации векторы признаков рукописного и голосового пароля объединялись в итоговый вектор путем конкатенации. Проведена оценка FRR и FAR для каждого набора признаков (из

таблица 4.9), как для однофакторной аутентификации, так и для двухфакторной. Результаты можно видеть в таблице 4.12.

Таблица 4.12. Наилучшие полученные показатели надежности (без учета подделок)

	Голосовой пароль		Рукописный пароль		Два фактора	
	24 кГц	8 кГц	с учетом силы нажатия	без учета силы нажатия	24 кГц и с учетом силы нажатия	8 кГц и без учета силы нажатия
Вероятности ошибок	FRR=0,101 FAR=10^{-5}	FRR=0,14 FAR=10^{-4}	FRR=0,05 FAR=10^{-5}	FRR=0,073 FAR=10^{-4}	FRR=0,03 FAR=10^{-10}	FRR=0,046 FAR=10^{-8}
Параметры адаптивной модели	$I_{ВИ}$=500 $N_{ВИ}$=256 $I_{ПИ}$=4 $N_{ПИ}$=256	$I_{ВИ}$=400 $N_{ВИ}$=128 $I_{ПИ}$=5 $N_{ПИ}$=128	$I_{ВИ}$=600 $N_{ВИ}$=384 $I_{ПИ}$=3 $N_{ПИ}$=128	$I_{ВИ}$=500 $N_{ВИ}$=256 $I_{ПИ}$=4 $N_{ПИ}$= 128	$I_{ВИ}$=700 $N_{ВИ}$=512 $I_{ПИ}$=3 $N_{ПИ}$=256	$I_{ВИ}$=600 $N_{ВИ}$=512 $I_{ПИ}$=3 $N_{ПИ}$=256

При тестировании с использованием подделок, исходя из объемов тестовой выборки подделок, можно говорить о FAR=0,001 (0,1%) для низкокачественных образов и FAR<0,001 (0,1%) для высококачественных (во втором случае не было обнаружено ошибок второго рода). Сравнение полученных результатов с достигнутыми ранее представлено в таблицах 4.13 и 4.14.

Таблица 4.13. Сопоставительные данные по эфективности известных методов распознавания личности по рукописным образам

Краткое описание метода	FRR, %	FAR,%	С учетом подделок
Многослойный персептрон + метод главных компонент [303]	6,4	7,4	+
Авторский метод реализации ПБК [250]. Ключ 256 бит. Защита повышает число ошибок	38,75	13,45	-
Сверточная искусственная нейронная сеть + метод опорных векторов [237]. Обучающая выборка “Свой” 30 примеров	2,17	13	+
Сверточная искусственная нейронная сеть [324]. Обучающая выборка “Свой” 30 примеров	1,48 2,63	1,48 2,63	+ +
Сверточная искусственная нейронная сеть [174]	2,42	2,42	+
Рекуррентная искусственная нейронная сеть [34]	2,37	2,37	+
Fuzzy commitment [245]	9	9	-
Линейный классификатор, метод главных компонент [263], тестирование на базе SVC2004	2,8	2,8	
Скрытые маркоские процессы, ИНС [243]	2	2	
Вейвлет коэффициенты Добеши D6, ИНС [302]	6	1,07	
Нечеткий экстрактор на базе кодов Адамара [281]	14,8	5	
Fuzzy Vault	7,85	6,91	
Предложенный метод на базе адаптивной нейро-иммунной модели ИИ. Обучающая выборка «Свой» 15 примеров	5	≈0,001	-
	5	< 0,1 (≈0)	+

Таблица 4.14. Сопоставительные данные по эфективности известных методов распознавания личности по голосу

Краткое описание метода	FRR, %	FAR,%
Метод, разработанный в ЦРТ (обобщённый метод моментов, машина опорных векторов, совместный факторный анализ, вариационный байесовский анализ) [98]	3	2
Нечеткий экстрактор [209]	20	20
Сверточные нейронные сети (с 3d-свертками) [346]	21,1	21,1
Сверточные нейронные сети [325]	3	3
Сверточные нейронные сети [320]	10,5	10,5
Набор данных RedDots и базовая модель ИИ для распознавания дикторов [343]	7,32	7,32
Тестирование различных признаков и моделей на наборе данных RedDots Нарушитель знал пароль. Лучший результат [349]	1,53	3,6
Тестирование различных признаков и моделей на наборе данных RedDots Нарушитель не знал пароль. Лучший результат [349]	0,09	< 0,1 (≈0)
Метод d-vector (модифицированный) [221]	7,61	7,61
Метод i-vector (модифицированный) [261]	2,5	2,5
Метод i-vector, скрытые марковские процессы. Набор данных RedDots (мужчины). Нарушитель знал пароль [261]	1,6	1,6
Метод i-vector, скрытые марковские процессы. Набор данных RedDots (мужчины). Нарушитель не знал пароль [255]	0,25	0,25
Метод i-vector, скрытые марковские процессы. Набор данных RedDots (женщины). Нарушитель знал пароль [255]	2,75	2,75
Метод i-vector, скрытые марковские процессы. Набор данных RedDots (женщины). Нарушитель не знал пароль [255]	0,32	0,32
Предложенный метод на базе адаптивной нейро-иммунной модели ИИ. Обучающая выборка «Свой» 15 примеров. Нарушитель знал пароль	10,1	< 0,1 (≈0)
Предложенный метод на базе адаптивной нейро-иммунной модели ИИ. Обучающая выборка «Свой» 15 примеров. Нарушитель не знал пароль	10,1	≈0,001

Как можно видеть, предложенный метод превосходит почти все представленные в таблицах аналоги по точности решений в режиме однофакторной аутентификации. Отметим, что в известных работах не учитывалось изменение ПФС субъекта, а также использовались иные наборы данных, что не позволяет сделать точное прямое сравнение. Однако из таблиц можно сделать вывод, что полученные результаты соответствуют мировому уровню, а разработанный метод в значительной степени устойчив к дрейфу биометрических данных.

Достоверность для оценок FAR≈10^{-5} и FRR=0,03 при доверительных интервалах ±0,000009 и ±0,01, соответственно, составляет более 0,99 (1690000 и 15600 опытов). Для более высоких FRR и FAR достоверность не будет ниже, а для более низких вероятностей FAR рассчитать достоверность затруднительно ввиду экстремально низких показателей.

В заключение параграфа следует добавить, что в рамках диссертационного исследования также проводились работы по оценке информативности тайных и открытых голосовых и рукописных биометрических образов, а также образов клавиатурного почерка, результаты которой отражены в публикациях [13, 329]. Экспериментально установлено, что для биометрических признаков, извлекаемых из голосовых, рукописных и клавиатурных паролей (индивидуальных для каждого пользователя) характерна более высокая информативность и как следствие такие образы в среднем дают более низкий процент ошибок в прямом численном эксперименте, чем, если в качестве ключевого стимула использовать фиксированную (единую для всех испытуемых) фразу. Это легко объяснить тем, что динамический биометрический образ содержит информацию, как о личности владельца, так и о содержании пароля, поэтому его уникальность выше.

Исследования тайных голосовых образов также проводились в работе [117], что позволило применить нейродинамический подход к обработке голосовых сигналов (когда к сигналу применяются методы аугментации не на стадии обучения, а на стадии распознавания образа).

4.11 Анализ результатов. Выводы

Ушной канал можно рассматривать как резонансную систему. Если воздействовать на него звуковыми волнами, то волны, отражаясь от стенок канала, меняют амплитудно частотные характеристики. В результате записанный на микрофон отраженный сигнал может характеризовать индивидуальные особенности строения уха человека. На основе данного принципа разработано устройство, с помощью которого собрана база акустических образов ушей 75 испытуемых. Сформированный набор данных находится в открытом доступе [333] (AIC-ears-75).

В качестве признаков апробированы параметры усредненного амплитудного спектра отраженного сигнала, а также параметры кепстрограмм. Комбинируя

разные типы окон на этапе вычисления усредненного спектра и кепстрограммы на его основе можно получить больше информации об особенностях строения ушного канала испытуемых. Установлено, что среди рассмотренных оконных функций прямоугольной, Барлетта (треугольной), гауссианы (классической и параметрической), Лапласа, Блэкмана, Хэмминга лучшие результаты получаются при сочетании окна Хэмминга с прямоугольным окном.

Апробировано два подхода к распознаванию образов ушей: на базе формулы гипотез Байеса и искусственных нейронных сетей. Байесовский классификатор показал меньший процент ошибок идентификации: EER=0,0053, в то время как наилучший результат для нейронных сетей составил EER=0,0266. Для нейросетевых классификаторов объем обучающей выборки в 8 примеров на испытуемого является сравнительно небольшим. Малый объем выборки обусловлен требованиями практики: процесс обучения биометрической системы должен быть быстрым и автоматическим. Настройку классификатора Байеса легче автоматизировать, так как при обучении ИНС следует следить за возникновением переобучения.

Натурный эксперимент с привлечением тех же испытуемых через полгода после первоначального сбора данных показал несущественное отклонение в количестве неверных решений (EER=0,008), из чего сделан вывод: идентификационные свойства наружного слухового прохода со временем заметно не меняются (в том числе, на них не влияет накопление серы в ушном канале).

Также рассчитаны вероятности ошибок в режиме верификации образов на основе Байесовского классификатора: FRR=0,1028 при FAR<0,0001 (в рамках эксперимента не было зафиксировано ни одной ошибки «ложного допуска»).

Предложен метод биометрической аутентификации на основе акустических образов уха и разработанной модели НПБК на базе корреляционных нейронов. Результаты эксперимента показали высокую эффективность предложенного метода: EER=0,0238 (FRR=0,093 при FAR<0,0001) при длине ключа L=8192 бит и объеме обучающих выборок «Свой» K_G = 6 и «Чужие» K_I=49. Очевидно, что при изменении параметров зондирующего сигнала $\upsilon(t)$, меняются и параметры эхо-

сигнала $u_{i,k}(t)$. Поэтому синтез нового сигнала $\upsilon(t)$ позволяет создавать новые акустические образы уха человека. Таким образом, чем выше энтропия возможных параметров сигнала $\upsilon(t)$, тем ниже вероятность ошибки «ложного допуска» (чем больше вариаций параметров сигнала $\upsilon(t)$, тем большее количество биометрических образов уха следует перебирать злоумышленнику, чтобы получить доступ, и тем ниже FAR).

В главе представлены методики извлечения рукописных и голосовых признаков и результаты корреляционного, спектрального и вейвлет анализа динамических биометрических образов. Проведена оценка информативности динамических биометрических признаков и образов. Проведен масштабный эксперимент по распознаванию 260 испытуемых, при этом учитывалось, что со временем биометрический образ человека устаревает (интервал между сбором обучающих и тестовых выборок составлял несколько недель). Тайный динамический биометрический образ является более информативным, чем открытый динамический образ [329]. Секрет, содержащийся в биометрическом образе, не только усиливает его защитные свойства (компрометация пароля не ведет к компрометации биометрического образа), но и повышает уникальность голосового или рукописного образа.

Разработан метод двухфакторной биометрической аутентификации по голосовому и рукописному паролям. Для обработки образов предложено два варианта извлечения признаков – образов низкого (FRR=0,046, FAR=10^{-8}) и высокого качества (FRR=0,03 при FAR=10^{-10}). На базе метода ГОСТ Р 52633.3 проведен эксперимент по точной оценке FAR для ГПБК в режиме однофакторной и двухфакторной аутентификации. Для этого предложена методика расширения выборки «Чужих».

Предложен алгоритм высоконадежной трехфакторной аутентификации (рисунок 4.1), комплексирует разработанные методы аутентификации по акустическим образам уха, рукописным и голосовым паролям. Для комплексного метода показатели ошибок принимают следующие значения (с высоким качеством голосовых и рукописных образов):

$$FRR = FRR_1 + (1 - FRR_1)\cdot FRR_2 = 0{,}093 + 0{,}907\cdot 0{,}03 = 0{,}12021\ (\approx 12\%\ \text{ошибок}),$$

$$FAR < FAR_1\cdot FAR_2 = 10^{-4}\cdot 10^{-10} = 10^{-14}\ (<10^{-12}\%\ \text{ошибок})$$

Достигнутые вероятности соответствуют высоконадежной системе аутентификации ($FAR<10^{-12}$). Полученный результат соответствует мировому уровню, удовлетворяет практическим целям и превосходит известные аналоги.

Оценим ориентировочный объем долговременной памяти компьютера, которая требуется для реализации процедуры биометрической аутентификации. Блок извлечения признаков акустических образов уха состоит из двух кодировщиков, которые включают 194648 и 243192 параметров, соответственно (включая веса нейронов). При сохранении структуры и параметров обученных нейронных сетей, объем файлов составляет 926448 и 1132881 байт, т.е. всего около двух мегабайт. Один корреляционный нейрон после обучения представлен таблицами весовых коэффициентов (если каждый вход представлен типом double, то для 5 входов получается 8 х 5 = 40 байт), связей (5 х 2 = 10 байт) и порогов (3 х 8 = 24 байта), а также необходимо хранить номер хеширующей таблицы (1 байт). Общий объем памяти одного нейрона составляет 75 байт. Один биометрический эталон, представленный в виде параметров обученного НПБК на базе 4096 корреляционных нейронов с 5 входами каждый составляет 307200 байт.

5 Технология автоматического синтеза и обучения доверенного искусственного интеллекта и ее применение

Технология — это совокупность методов и инструментов для достижения желаемого результата. В более широком смысле — это совокупность научных знаний (концепцией, моделей, методов и алгоритмов), применяемых для решения практических задач.

Исходя из такой формулировки, разработанная технология автоматического синтеза и обучения нейросетевых моделей доверенного ИИ включает следующие основные элементы и научно-технические положения:

1. Концепцию защищенного исполнения нейросетевых алгоритмов искусственного интеллекта от исследования и компрометации знаний, которая является идеологической основой технологии.
2. Модель корреляционных нейронов и алгоритм их автоматического синтеза и обучения на малых выборках данных.
3. Модель двухслойного нейросетевого преобразователя образов в код на основе корреляционных и классических нейронов для задач классификации образов (НПБК является частным случаем такого преобразователя). *Нейросетевой преобразователь образов в код* – преобразователь, основанный на неглубокой нейронной сети, способный преобразовывать вектор нечетких, неоднозначных признаков образа «Свой» в четкий однозначный код криптографического ключа или пароля. Преобразователь, откликающийся случайным выходным кодом на воздействие образов «Чужой». Преобразователь образов в код обучается автоматически, после чего позволяет разделять входные данные на два класса – «Свой» и «Чужой», генерируя стабильный фиксированный код в одном случае и нестабильный случайный код во втором случае. В результате обучения преобразователя формируются знания, которые представляют собой срез информации, необходимый для его функционирования. Знания

преобразователя кодируются специфическими параметрами, компрометация которых не приводит к раскрытию знаний. Нейросетевые преобразователи образов в код можно встроить в цепочку обработки данных (рисунок 5.1) для повышения защищенности систем ИИ (СИИ) от компьютерных атак, а также знаний ИИ от компрометации.

4. Способ автоматического синтеза и обучения двухслойного преобразователя образов в код. Первый слой корреляционных нейронов настраивается в соответствии с алгоритмом, предложенным воо второй главе. Второй слой классических нейронов настраивается в соответствии с ГОСТ Р 52633.5-2011.
5. Адаптивная нейро-иммунная модель ИИ и алгоритмы ее обучения с учителем и с подкреплением, позволяющие предупредить или снизить влияние концептуального дрейфа.

Разработанная технология позволяет реализовать *отчуждение ИИ от принятия решений* – архитектурный принцип, усиливающий функциональную безопасность систем ИИ, при котором ИИ не управляет объектами из реального мира напрямую, а делает это через посредника – нейросетевой преобразователь образов в код (в биометрических приложениях – через НПБК).

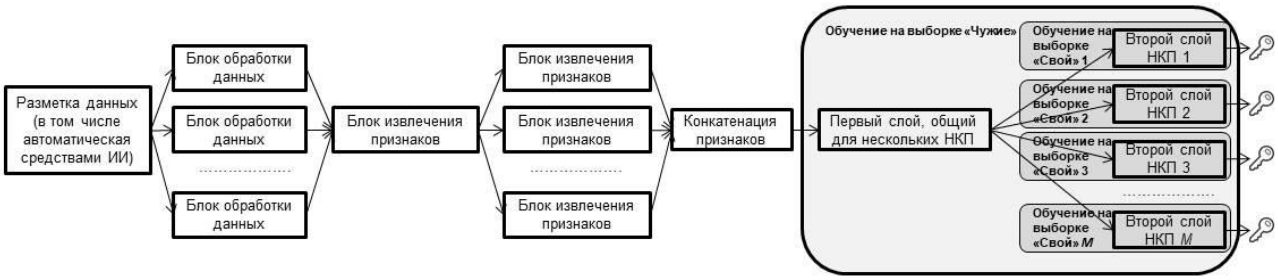


Рисунок 5.1– Пример схемы построения ИИ для задачи идентификации М классов и встраивания преобразователей образов в код в цепочку обработки данных

Основной акцент в настоящей работе сделан на приложениях высоконадежной биометрической аутентификации. Поэтому в данной области имеются внедрения на предприятиях ООО «Системы информационной безопасности» (СИБ), ООО «Открытый код» и ООО "КРАФТ ЛАБ".

На базе технологии под руководством Сулавко А.Е. на базе ОмГТУ разработана первая редакция национального стандарта ГОСТ Р «Искусственный интеллект. Нейросетевые алгоритмы в защищённом исполнении. Автоматическое обучение нейросетевых моделей на малых выборках в задачах классификации», стандарт поставлен в план национальной стандартизации на 2022 год техническим комитетом № 164 «Искусственный интеллект». Стандарт в первой редакции не является отраслевым и ориентирован на применение на объектах критической информационной инфраструктуры при разработке ответственных приложений ИИ. Это первый стандарт в мире, который регламентирует особенности создания и обучения нейросетевых моделей ИИ, исполняемых в защищенном от исследования режиме.

Разработанная технология применима для широкого класса приложений ИИ, основанных на процедурах классификации образов. Она использована при разработке линейки программных продуктов AIConstructor (AIC) [aiconstructor.ru]. Научным руководителем разработки продуктов AIC является Сулавко А.Е.

AIC desktop – это программный комплекс, ориентированный на исследования по машинному обучению в области ИБ, анализ биомедицинских данных и классификацию образов.

AIC ModelOps Platform – это веб-сервис для цифровой трансформации предприятий с обеспечением надежности, прозрачности и безопасности процессов разработки и внедрения ИИ. Продукт разрабатывается в рамках проектно-конструкторской деятельности ООО «АИ ЗИОН» для дальнейшей реализации.

На базе предложенной технологии разработана библиотека автоматического машинного обучения AIC library для быстрого создания защищенного искусственного интеллекта в компьютерных приложениях, основанная на корреляционных нейронах и их сетях. Библиотека является одним из инструментов, которые интегрированы с AIC ModelOps Platform.

Также результаты настоящей работы использованы:

- в рамках инструментально-лабораторных обследований и тестирования пациентов на аппаратно-программном комплексе в Центре здоровья поликлиники №1 бюджетного учреждения здравоохранения Омской области «Медико-санитарная часть № 4»);
- в учебныом процессе (ФГАОУ ВО СПбГЭТУ «ЛЭТИ» и ФГАОУ ВО «ОмГТУ»).

Соответствующие акты внедрения представлены в Приложении Б.

5.1 Границы применимости разработанных методов и технологии

Для сохранения рукописного и голосового образа в виде вектора признаков потребуется немного больше памяти, так как для функционирования адаптивной нейро-иммунной модели необходимо хранить обучающую выборку (по крайней мере, ту часть, которая требуется для настройки вновь синтезируемых детекторов приобретенного иммунитета, а именно валидационную). Один рукописный образ включает от 521 до 782 признаков (в зависимости от поддержки давления), а голосовой - от 350 до 570 признаков. Каждое значение признака может быть описано типом double (8 байт), соответственно один рукописный пример составляет от 4168 до 6256 байт, а голосовой – от 2800 до 4560 байт. При объеме валидационной выборки в 10 примеров (что является более чем достаточным), максимальный объем данных пользователя составит не более 106 кбайт. Также следует расчитать объем данных нейро-иммунной модели. Один детектор представляет собой набор параметров, не превышающих 2 кбайта (в зависимости от типа меры близости, лежащей в его основе и количества признаков, с которыми он соединен). Хотя в среднем это значение составляет менее 500 байт. При $N_{ВИ}$=512 и $N_{ПИ}$=256 объем всех данных детекторов составляет не более 1536 кбайт. С учетом гиперпараметров нейро-иммунной модели и объемом данных эталона уха, максимально возможный объем данных на одного пользователя можно округлить до 2 мбайт.

Столь незначительный объем памяти позволяет реализовать разработанные методы и алгоритм многофакторной биометрической аутентификации даже на низкопроизводительных мобильных устройствах. В системе со 100000 пользователей потребуется не более 196 гбайт для хранения всех эталонов.

Разработанные технология и методы могут применяться в многопользовательских системах десятками и сотнями тысяч пользователей. Архитектура системы аутентификации может быть распределенной. Модуль извлечения признаков можно расположить на выделенном сервере либо реализовать в виде приложения непосредственно на клиентских устройствах в виде десктопного или мобильного приложения (последнее рекомендуется с точки зрения усиления безопасности). Классификаторы на базе НПБК либо гибридные классификаторы, которые предполагает схема алгоритма многофакторной аутентификации (рисунок 4.1), могут быть размещены как на выделенных серверах, так и на клиентских устройствах. Адаптивные нейро-иммунные модели без дополнительной криптографической защиты или в отрыве от схемы, представленной на рисунке 4.1, следует размещать только на клиентских устройствах.

При размещении классификаторов на сервере следует учитывать, что ориентировочное число пользователей, которые может быть обслужено одним процессорным ядром при очень высокой плотности запросов, по предварительным оценкам составляет от 50 до 500, в том числе в зависимости от используемой схемы аутентификации (трехфакторной или однофакторной). Оценка произведена на Intel(R) Core(TM) i9-9900K CPU @ 3.60GHz. При реализации технологии HyperThreading количество пользователей на ядро может быть увеличено до 2 раз

Со временем точность может снижаться (на 20%-30), если пользователь надолго перестанет пользоваться системой аутентификации. Поэтому пользователям рекомендуется периодически (хотя бы раз в полгода) проходить процедуру аутентификации, чтобы адаптивная нейро-иммунная модель могла учитывать эти изменения.

5.2 Проект национального стандарта

Разработанный стандарт ГОСТ Р «Искусственный интеллект. Нейросетевые алгоритмы в защищённом исполнении. Автоматическое обучение нейросетевых моделей на малых выборках в задачах классификации» [69] дополняет и в значительной степени расширяет уже имеющийся стандарт ГОСТ Р 52633.5-2011. Данный стандарт отличается от базового стандарта ГОСТ Р 52633.5-2011 тем, что:

- он распространяется на нейросетевые преобразователи не только биометрических, но и любых иных образов данных в код;
- он распространяется на нейросетевые преобразователи образов в код, основанные на использовании корреляционных нейронов;
- он распространяется на корреляционные нейроны, имеющие два выходных состояния (один нейрон дает два бита выходного кода);
- он распространяется не только на задачи бинарной классификации образов (когда имеется два класса образов «Свой» и «Чужой»), но и идентификации образов на открытом множестве (когда существует множество известных классов образов «Свой», а также образы «Чужой», к ним не относящиеся).

Новые свойства нового стандарта в сравнении с базовым стандартом ГОСТ Р 52633.5-2011 позволяют:

- повысить длину бинарного кода (пароля, криптографического ключа), связываемого с целевым классом образов (классом «Свой»);
- снизить вероятности ошибок 1-го и 2-го рода преобразования образов целевого класса (класса «Свой») в бинарный код (пароль, криптографический ключ);
- улучшить хэширующие (перемешивающие) свойства нейросетевого преобразователя для примеров образов данных, не принадлежащих целевому классу (т.е. принадлежащих классу «Чужой»);
- повысить уровень защищенности знаний нейросетевого преобразователя от компрометации, как при условии применения технической спецификации

26.2.002-2020 «Системы обработки информации. Криптографическая защита информации. Защита нейросетевых контейнеров с использованием криптографических алгоритмов», так и без ее применения;

– сделать невозможным или, по крайней мере, усложнить несанкционированное управление СИИ путем манипуляций с моделями машинного обучения, анализ логики работы ИИ с целью повлиять на его решения, извлечение знаний из памяти ИИ (в том числе путем зондирования модели), а также их интерпретацию;
– снизить вероятность успеха состязательных атак, реализуемых злоумышленником путем наложения различных шумов на исходный образ данных «Свой» или путем создания синтетических примеров образа данных «Чужой».

Разрабатываемый стандарт регулирует требования к нейросетевым моделям машинного обучения, исполняемым в защищенном режиме, предназначенным для решения задач классификации образов по набору признаков и использования в ответственных приложениях ИИ, а также на объектах критической информационной инфраструктуры. Стандарт направлен на повышение безопасности ИИ путем защиты от ряда угроз информационной безопасности, влияющей на безопасность СИИ в целом.

Стандарт построен на необходимости применения нейро-корреляционных преобразователей (НКП) – нейросетевых преобразователей образов в код на основе сети разностных корреляционных нейронов Байеса-Минковского. НКП принимает на вход образы, представленные вектором признаков. Потенциальная длина ключа для такого преобразователя значительно превосходит длину ключа для нейросетевого преобразователя биометрия-код, обученного по ГОСТ Р 52633.5-2011.

В соответствии со стандартом в ответственных приложениях ИИ и на объектах критической информационной инфраструктуры системы ИИ следует строить так, чтобы можно было выделить блоки извлечения признаков и НКП, реализующий защищенное исполнение процедуры классификации и отчуждение

ИИ от принятия решений. НКП должен встраиваться в цепочку обработки данных, как показано на рисунке 5.1.

В стандарте приведена классификация корреляционных нейронов по уровню защиты данных от компрометации и по принципу их функционирования, указаны требования к законам распределения признаков, к обучающим данным и независимости тренировочных примеров, к тестированию качества НКП, описан алгоритм их обучения.

При разработке стандарта учтены:

– международные стандарты (ISO/IEC), разработанные техническими комитетами по стандартизации ISO/IEC JTC 1/SC 42 «Artificial intelligence», ISO/IEC JTC 1/SC 37 «Biometrics» и ISO/IEC JTC 1/SC 27 «Information security, cybersecurity and privacy protection»;
– национальные стандарты России (ГОСТ Р) и технические спецификации, разработанные в технических комитетах ТК 164 «Искусственный интеллект», ТК 362 «Защита информации»;
– научные статьи в ведущих рецензируемых журналах России и мира.

Стандарт разработан в рамках НИР «Защищенный режим исполнения искусственного интеллекта на базе автоматически обучаемых сетей автокорреляционных нейронов» (Приложение В). Проведенная НИР включала:

– выполнение аналитико-синтетического обзора научных работ, а также международных и национальных стандартов в области искусственного интеллекта и защиты информации на тему проблемы безопасности искусственного интеллекта;
– обоснование необходимости защиты искусственного интеллекта от ряда угроз информационной безопасности и разработки национального нового стандарта;
– разработку план-проспекта (Приложение Г) и проекта предлагаемого стандарта.

К настоящей работе прилагается акт внедрения, свидетельствующий о разработке стандарта на базе ОмГТУ (Приложение Б), выдержки из плана работы ТК 164 на 2022 год (Приложение Д), подтверждающие включение разработанного стандарта в план национальной стандартизации на 2022 год, официальное письмо от ТК 164 с предложением включить Сулавко А.Е. в состав экспертов от России Международного технического комитета ISO/IEC JTC 1/SC 42 «Artificial intelligence» (Приложение Е).

5.3 Библиотека автоматического машинного обучения и программный комплекс на ее основе

Назначение библиотеки AIC library – это синтез и автоматическое обучение нейросетевых и адаптивных нейро-иммунных моделей ИИ на малых выборках данных для решения задач классификации образов (идентификации, верификации) при разработке компьютерных приложений. AIC library может использоваться как отдельно, так и в качестве дополнения к существующим библиотекам глубокого обучения многослойных нейронных сетей.

Поддерживаемые модели и алгоритмы обучения:

- однослойные и многослойные сети корреляционных нейронов с поддержкой защищенного режима исполнения и алгоритмы их автоматического послойного обучения (с учителем);
- однослойные и многослойные гибридные на базе классических, квадратичных и корреляционных нейронов и алгоритмы их автоматического послойного обучения (с учителем);
- адаптивные нейро-иммунные сети и алгоритмы их обучения с учителем и онлайн-обучения с подкреплением.

За счет поддержки защищенного режима исполнения нейросетевых алгоритмов классификации образов создаваемый с помощью AIC library искусственный интеллект будет обладать повышенной устойчивостью к ряду атак

(«ключ под ковриком, «атака на решающий бит», состязательные атаки, «извлечение знаний»).

Библиотека AIC library реализована в двух вариантах: для использования при программировании на языке C#, и Python, однако планируется реализовать ее на других популярных языках программирования, в том числе, активно применяемых в задачах анализа данных (например, R).

AIC desktop [128, 129] – это программный комплекс для проведения научных исследований по машинному обучению. Подойдет для небольших научных групп и отдельных исследователей (студентов, аспирантов, постдоков). Может быть использован в образовательных целях, а также для работы над малыми и средними научными проектами (диссертациями, отчетами и т.д.). Ориентирован на задачи информационной безопасности, анализ биомедицинских данных, классификацию образов. К основным функциональным возможностям можно отнести:

– Анализ данных и распознавание образов. Анализ данных с применением признанных методов математической статистики для выявления закономерностей. Реализовано множество известных и новых методов машинного обучения (нейронные сети, Байесовские сети, иммунные сети и др.), а также возможность анализа промежуточных этапов принятия решений классификатором для выявления причинно-следственных связей и повышения объяснимости решений. Имеется возможность построения различных графиков, для визуализации результатов анализа;
– Конструкторы нейронных сетей. Визуальный редактор позволяет создавать сложные архитектуры многослойных искусственных нейронных сетей (сверточных, полносвязных, гибридных). Сети можно объединять в ансамбли и комплексировать с другими (в том числе с не сетевыми) моделями. Можно решать задачи классификации и извлечения признаков;
– Высокая достоверность оценок. Вероятности ошибок 1-го и 2-го рода, точность классификации можно оценить разными способами, проводя вычислительные эксперименты различной сложности с учетом множества

параметров, что позволяет получать результаты с высокой достоверностью и оценивать доверительную вероятность полученных оценок;

– Защита информации, биометрические технологии. Поддержка нечетких экстракторов, нейросетевых преобразователей биометрия-код (НПБК), обучаемых по ГОСТ Р 52633.5, НПБК на базе корреляционных нейронов, а также гибридных НПБК с возможностью оценки стабильности выходных разрядов с помощью нескольких метрик;

– Форматы данных. Информацию об образах можно загружать в виде векторов признаков (xml, txt) и в виде сырых (размеченных и частично размеченных) данных (bmp, wav, edf, csv и др.), относящихся к любой предметной области (звук, изображения, термограммы, эхограммы, ЭКГ, ЭЭГ, биометрические и биомедицинские образы и др.). После загрузки данные преобразуются в унифицированный формат. Имеется возможность экспорта результатов исследований.

В таблице 5.1 представлен перечень функционала, реализованного в AIC desktop. Архитектура AIC desktop в общем виде представлена на рисунке 5.2. На рисунках 5.3 и 5.4 представлены экранные формы программного комплекса.

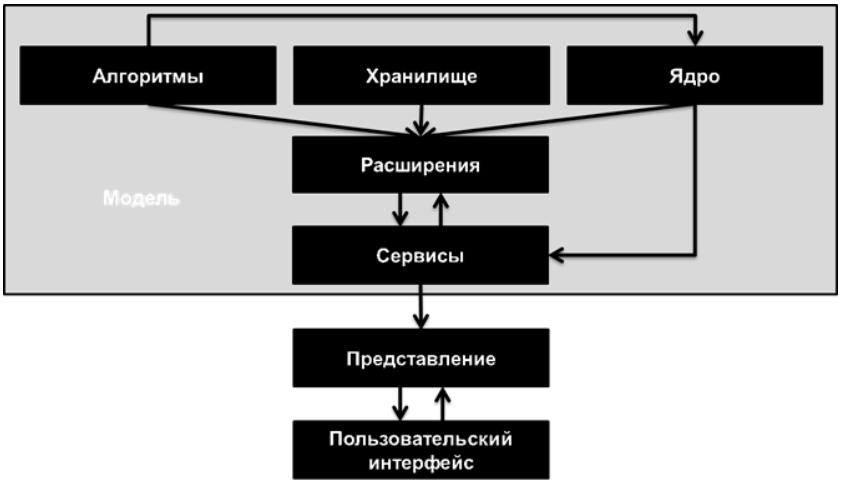


Рисунок 5.2 – Архитектура программного комплекса AIC desktop

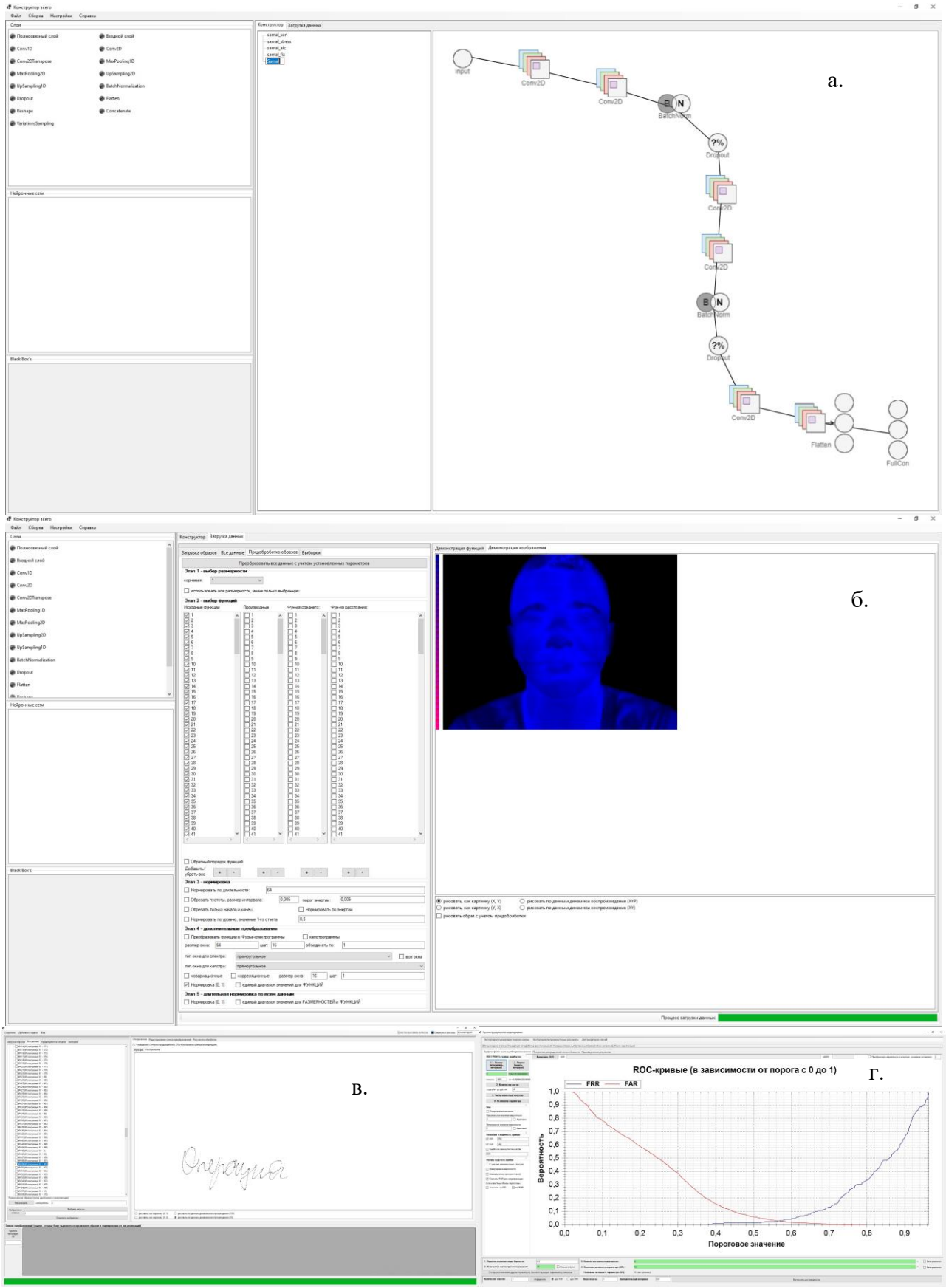


Рисунок 5.3– AIC desktop: а. конструктор ИНС; б. анализ термограмм лица; в. анализ рукописных образов; г. оценка вероятностей ошибок FRR и FAR

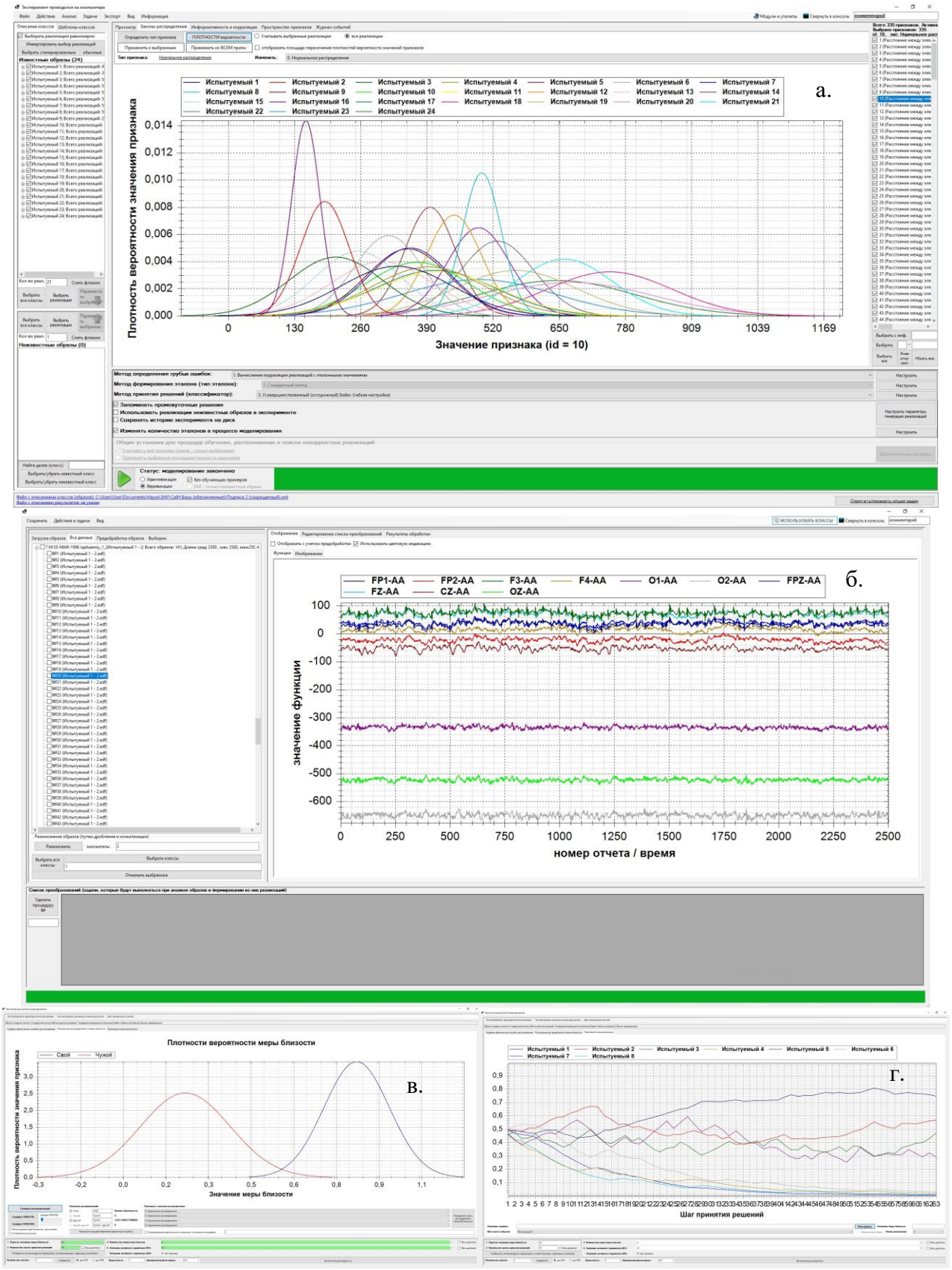


Рисунок 5.4– AIC desktop: а. анализ плотностей вероятности признаков; б. анализ ЭЭГ; в. оценка вероятностей ошибок 1-го и 2-го рода; г. анализ промежуточных решений классификатора для выявления причинно-следственных связей и объяснения решения

Таблица 5.1. Реализованные и разрабатываемые функции AIC desktop

Функционал	Реализовано	В разработке
Многослойные нейронные сети	сверточные и полносвязные	рекурентные
Глубокое обучение	поддержка обучения на процессоре и видеокарте, алгоритмы: градиентный спуск и его оптимизаторы (Adam и др.)	
Типы задач	классификация (идентификация, верификация), понижение размерности (извлечение признаков классическими методами анализа и нейронными сетями - автокодировщиками), повышение размерности (переход в спрямляющее пространство мета-признаков Байеса-Минковского), порождение объектов (частично - только на базе метода Монте-Карло)	генеративно-состязательные сети (GAN), вариационные автокодировщики
Преобразователи биометрия-код (ПБК), защита ИИ	нечеткие экстракторы, нейросетевые ПБК (ГОСТ 52633.5), ПБК на базе сетей Байеса-Минкоского, гибридные ПБК, оценка энтропии и стабильности выходных кодов НПБК	
Автоматически обучаемые сетевые модели	искусственные иммунные сети, гибридные нейронные сети, широкие нейронные сети, сети радиально-базисных функций, сети нейронов Байеса-Минковского и др.	
Статистический подход	семейство Байесовских классификаторов ("наивный", "осторожный" и др.), проверка гипотез о законе распределения (8 законов для описания признаков), оценка коррелированности и информативности признаков	
Другие классификаторы	простые параметризованные классификаторы (на базе мер Минковского/Махаланобиса и Байеса-Минковского), множество статистических критериев для проверки гипотез и др.	деревья решений, машина опорных векторов
Ансамбли моделей	стеккинг, усреднение модели, бэггинг и бустинг (входят в функционал сетевых моделей), другое (переход в пространство мета-признаков Байеса-Минковского, объединение пространств признаков)	алгоритмы бустинга (AdaBoost и др.), случайный лес, нейродинамика
Анализ сигналов	спектрограммы, кепстрограммы (на базе STFT, разные оконные функции), корреляционный и вейвлет анализ (разные базисы, алгоритм Малла) и др.	
Тестирование	прямой численный эксперимент (оценка точности, вероятности ошибок 1-го и 2-го рода, Equal Error Rate), тестирование по ГОСТ 52633.3 (частично), оценка достоверности через доверительных интервал	тестирование по ГОСТ 52633.3 со скрещиванием образов
Форматы данных	загрузка "сырых" данных из CSV, WAV, PCM, EDF+, файлов термограмм и др., загрузка обработанных данных (признаков) из xml, txt	поддержка большего числа графических форматов
Модули	статистического анализа признаков и классификации, конструктор многослойных нейронных сетей,генератор шаблонов классов, модуль оценки энтропии кодов ПБК, модуль анализа "сырых" данных, модуль оценки ошибок, экспорт данных, консольный модуль	создание сложных ансамблей из различных методов анализа, классификаторов и нейронных сетей, генератор научных отчетов, мастер

Схема взаимодействия программных модулей и классов иллюстрируется на рисунке 5.5. AIC library интегрирована с AIC desktop для расширения функциональности программного комплекса.

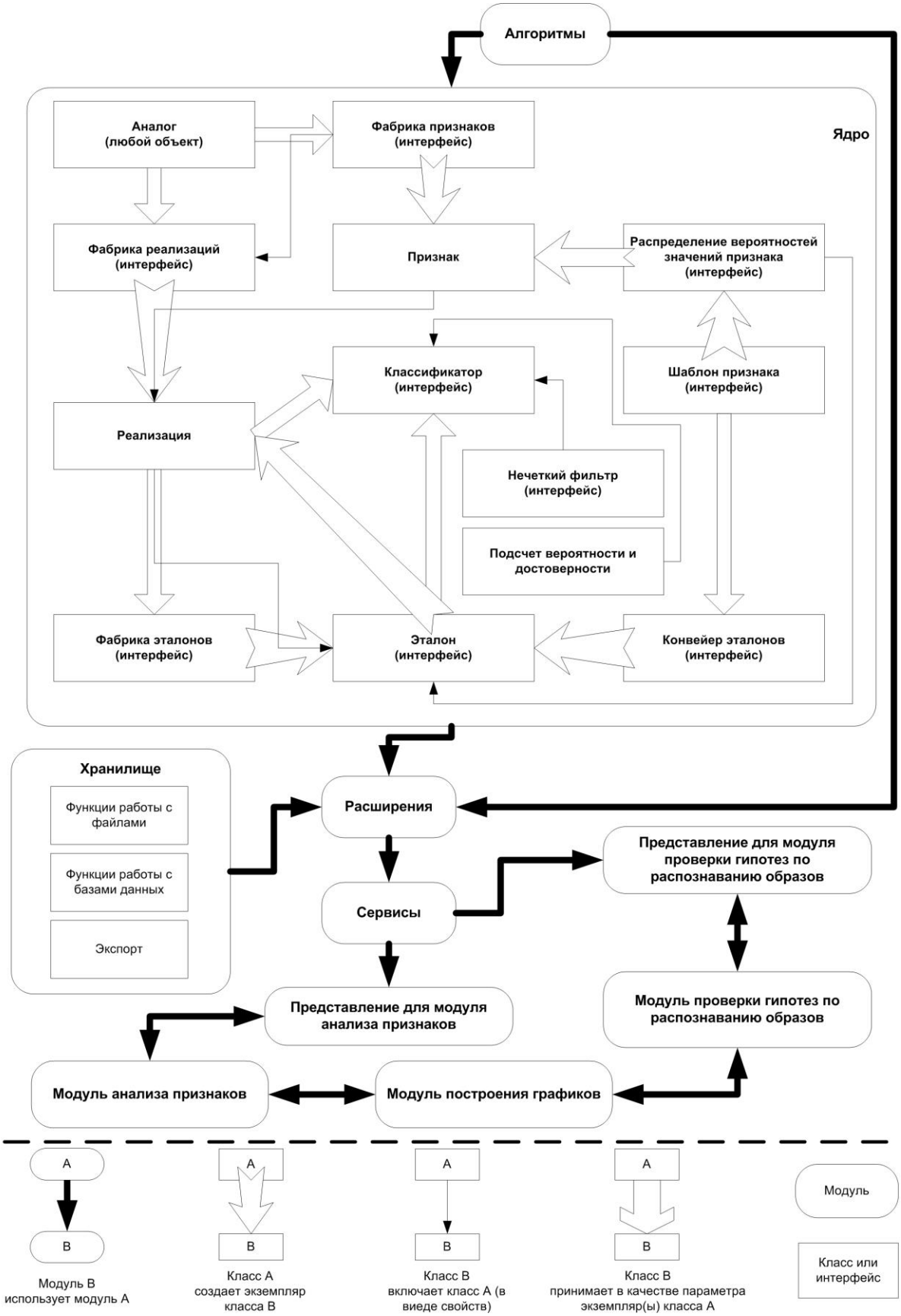


Рисунок 5.5 – Структура программного комплекса AIC desktop

AIC desktop поддерживает ряд форматов для загрузки "сырых" (размеченных, частично размеченных) данных (csv, edf+, wav, бинарный и другие). Возможность загрузить эти данные имеется в модуле анализа "сырых" данных (модуле извлечения признаков) и конструкторе нейронных сетей. Так как каждая предметная область исследований имеет свою специфику, собрать набор "сырых" данных первоначально можно только с помощью сторонних программ, сохранив их в одном из поддерживаемых форматов. Также имеется возможность пересохранить данные из специализированных форматов в универсальный формат (*.shv), который специально создан для быстрой загрузки любых данных.

Описания образов в виде векторов признаков можно хранить в одном из универсальных форматов. Загрузка производится из основного модуля (модуля статистической обработки). Поддерживается 4 формата: полный и сокращенный форматы на базе xml позволяет создавать сложные описания классов образов в любом пространстве признаков, независимо от предметной области (поддерживается не только загрузка, но и сохранение в этот формат, описание формата и пример приведены в листинге ниже), в сокращенном формате названия xml-тегов пишутся в сокращенном виде, а также два формате на базе txt, которые предложены другими исследователями.

Листинг. Структура xml-файла

```
<?xml version="1.1" encoding="UTF-8" ?>
<!-- убедитесь, что кодировка указана верно -->
<Classes lang=”ru”>
<!-- lang - локализация, если в качестве разделителя запятая, он равен “ru”, если точка - “en” -->
<!-- Specification - текстовое описание признаков, полученных при различных условиях или от физически разных факторов -->
<Specification description="Эксперимент проводился в обычных условиях">
<!-- в теге Feature есть необязательный атрибут unused, который может быть равен любому значению, если он есть, то признак не используемый (выключен)-->
<Feature id="1" description="Расстояние между глазами" />
<Feature id="2" description="Цвет глаз" />
<Feature id="3" description="Отношение между размерами головы и лба" />
.....
</Specification >
<Features>
<!-- Class - описание пользователя -->
<Class name="Пользователь 1">
<!-- в теге Realization есть также необязательный атрибут timeId- Время получения реализации либо ее порядковый номер (в любом случае - это беззнаковое длинное целое unsigned long) -->
<Realization>
<Feature id="1" value="0,101" />
<Feature id="2" value="1" />
<Feature id="3" value="333" />
.....
</Realization>
<Realization>
<Feature id="1" value="0,254" />
<Feature id="2" value="1" />
<Feature id="3" value="342" />
```

```
.....
</Realization>
</Class>
<Class name="Пользователь 2">
<Realization>.....</Realization>
<Realization>.....</Realization>
.....
</Class>
<Class .....</Class>
.....
</Features>
</Classes>
```

Пример:

```
<?xml version="1.0" encoding="windows-1251"?>
<Classes lang="ru">
<Specification description="Эксперимент по формированию рукописных паролей проводился в обычных условиях">
<Feature id="1" description="Rxy" />
<Feature id="2" description="Ryp" />
<Feature id="3" description="Rxp" />
<Feature id="4" description="Rxx" />
<Feature id="5" description="Ryy" />
<Feature id="6" description="Rpp" />
<Feature id="7" description="Ap1" />
<Feature id="8" description="Ap2" />
<Feature id="9" description="Ap3" />
<Feature id="10" description="Ap4" />
</Specification >
<Features>
<Class name="Пользователь 1">
<Realization>
<Feature id="1" value="-0,417" />
<Feature id="2" value="0,09" />
<Feature id="3" value="0,284" />
<Feature id="4" value="0,675" />
<Feature id="5" value="0,309" />
<Feature id="6" value="0,493" />
<Feature id="7" value="0,00137828" />
<Feature id="8" value="9,798E-5" />
<Feature id="9" value="5,498E-5" />
<Feature id="10" value="4,214E-5" />
</Realization>
</Class>
<Class name="Пользователь 2">
<Realization>
<Feature id="1" value="-0,534" />
<Feature id="2" value="0,07" />
<Feature id="3" value="0,312" />
<Feature id="4" value="0,234" />
<Feature id="5" value="0,754" />
<Feature id="6" value="0,134" />
<Feature id="7" value="0,00754334" />
<Feature id="8" value="7,798E-5" />
<Feature id="9" value="9,498E-5" />
<Feature id="10" value="3,214E-5" />
</Realization>
</Class>
</Features>
</Classes>
```

Принципы построения xml-файла:

1. Каждая реализация - одно измерение значений всех (или некоторых) признаков (например, одно перемещение мыши между 2 элементами, одна подпись, снимок лица и др.), т.е. один образец данных.

2. Каждый класс - это классифицируемый объект (пользователь, человек, субъект, явление и др.).

3. В каждом классе должна быть одна или несколько реализаций.

4. В реализациях может быть различное количество признаков и различное количество значений одного и того же признака (т.е. признаков с одинаковыми id), признаки могут быть представлены в любом порядке, даже в различном от реализации к реализации.

Пример работы с AIC desktop приведен в Приложении Ж.

5.4 Система управления жизненным циклом доверенного искусственного интеллекта

В рамках проектно-конструкторской деятельности ООО «АИ ЗИОН» разрабатывается флагманский продукт линейки AIConstructor [180], в котором использовались основные результаты настоящей работы. На данный момент разработан опытный образец системы.

AIC ModelOps Platform – это корпоративная среда управления жизненным циклом ИИ и моделей принятия решений, которая может быть использована для автоматизации, отслеживания и контроля рабочих процессов на всех этапах построения модели: от исследования до внедрения в бизнес среду. Это полнофункциональный продукт для обеспечения надежности, прозрачности и безопасности процессов разработки и внедрения искусственного интеллекта.

ModelOps платформы ориентированы в первую очередь на управление жизненным циклом широкого спектра оперативных моделей искусственного интеллекта и принятия решений, включая машинное обучение, графы знаний, правила, оптимизацию, лингвистические и агент-ориентированные модели. Жизненный цикл модели ИИ включает три основных этапа: исследование и разработку, внедрение модели ИИ в практику, мониторинг модели и оценку стабильности ее функционирования. На каждом этапе возникают специфические риски. С помощью подобных решений можно снизить эти риски от внедрения искусственного интеллекта, в том числе, повысить и отслеживать надежность решений, обнаружить дрейф данных и концепций, повысить объяснимость решений моделей ИИ, их защищенность от состязательных атак и извлечения знаний.

Функции, выполнение которых должен обеспечивать AIC ModelOps Platform:

1. Веб-сервис. AIC ModelOps Platform позволит создавать архитектуры моделей ИИ, обеспечивать их масштабирование и версионирование, а также

прозрачность производительности, характеристик и использования модели с течением времени. Созданные модели можно будет переносить в бизнес-среду за считанные минуты.

2. Испытательный полигон. С помощью AIC ModelOps Platform возможно управлять множеством моделей ИИ в одном месте. Производить анализ их активности и производительности, следить за тенденциями вывода и поведением данных. В системе будет присутствовать возможность настраивать автоматические оповещения для выявления предвзятости, дрейфующих характеристик, низкой производительности и др. А также генерировать подробные отчеты жизненного цикла модели, с момента ее создания до внедрения в бизнес среду.
3. Универсальность. AIC ModelOps Platform позволит использовать множество популярных фреймворков и сред машинного обучения.
4. Инфраструктура. AIC ModelOps Platform будет иметь возможность автоматического масштабирования и развертывания по выбору заказчика - локально, облачно или гибридно с конвейерами развертывания для систем CI/CD.
5. Защита моделей ИИ. Создаваемый с помощью AIC ModelOps Platform искусственный интеллект будет устойчив к ряду атак («ключ под ковриком, «атака на решающий бит», состязательные атаки, «извлечения знаний»).
6. Конструктор моделей ИИ. AIC ModelOps Platform позволит из готовых элементов конструировать конвейеры искусственного интеллекта на базе стандартных и пользовательских компонентов и моделей, способные решать произвольные задачи, определяемые пользователем. Для типовых задач будут предложены шаблоны интеллектуальных систем.
7. Объяснимый искусственный интеллект. AIC ModelOps Platform позволит обеспечить понимание выбора модели в различных сценариях использования. Предоставить функции объяснимости для аудитории, которая плохо разбирается в тонкостях работы алгоритмов, но которым

необходимы метрики для оценки качества работы моделей принятия решений.

Основные бизнес-процессы, которые реализуются с помощью AIC ModelOps Platform, представлены на рисунках 5.6-5.8.

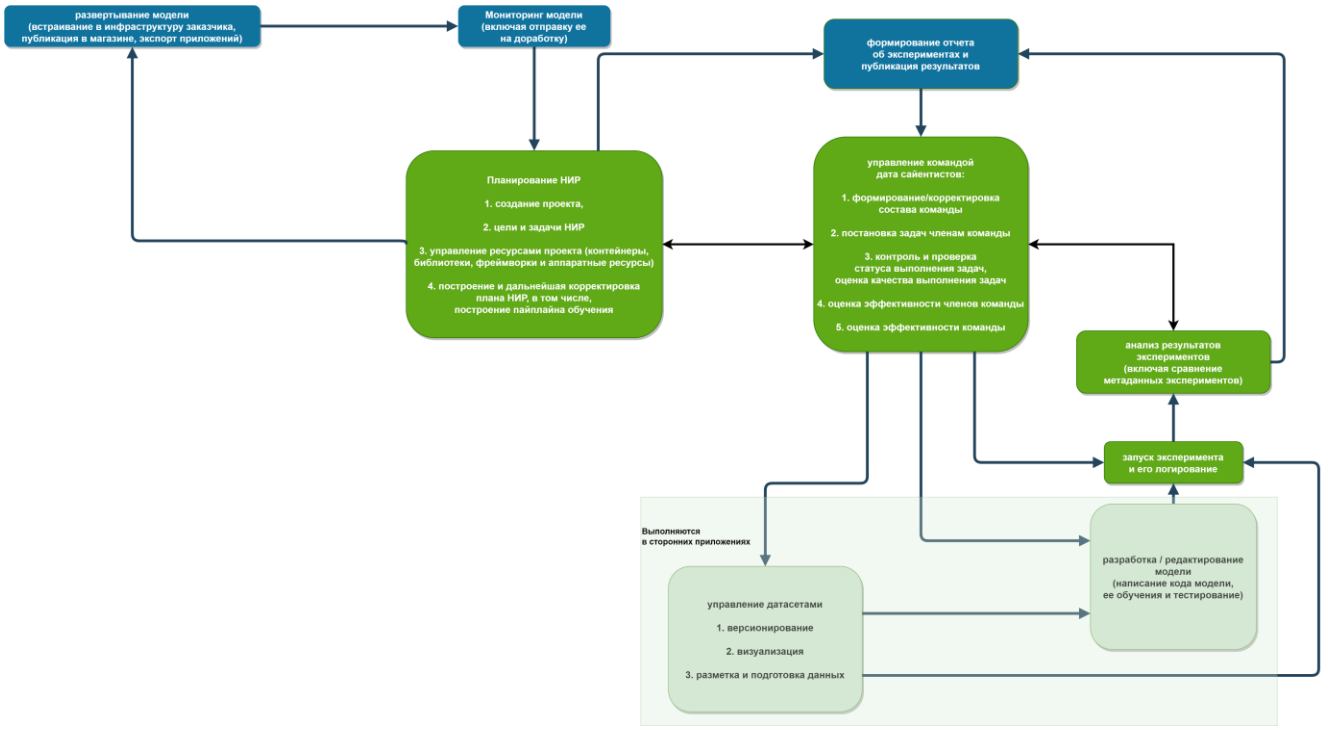


Рисунок 5.6 – Декомпозиция основных бизнес-процессов, реализуемых AIC ModelOps Platform

AIC ModelOps Platform может быть представлен в виде облачного, локального или гибридного решения, в зависимости от предпочтений и требований клиента. Облачное решение представляет собой веб-сервис, доступный в сети Интернет. Локальное решение представляет собой веб-сервис, развёрнутый и доступный исключительно в локальной сети клиента. В свою очередь гибридное решение предусматривает возможность подключения части инфраструктуры клиента к облачному веб-сервису, расположенному на серверах компании. Продукт имеет следующие особенности и принципиальные отличия от других систем ModelOps:

– возможность размещения и развертывания всей среды в инфраструктуре заказчика;

- возможность использования и оплаты только необходимого клиенту функционала;
- продукт имеет встроенные инструменты, обеспечивающие устойчивость искусственного интеллекта к состязательным атакам и зондированию моделей, не имеющие аналогов в мире. В основе лежит инновационная технология, позволяющая создать посредника между бизнес-логикой и ИИ;
- продукт имеет встроенные инструменты, позволяющие предупредить постепенный дрейф концепций при разработке искусственного интеллекта путем онлайн-обучения моделей, функционал с аналогичными свойствами отсутствует у конкурентов.

Эти новаторские функции стали возможны во многом благодаря интеграции AIC ModelOps Platform и AIC library.

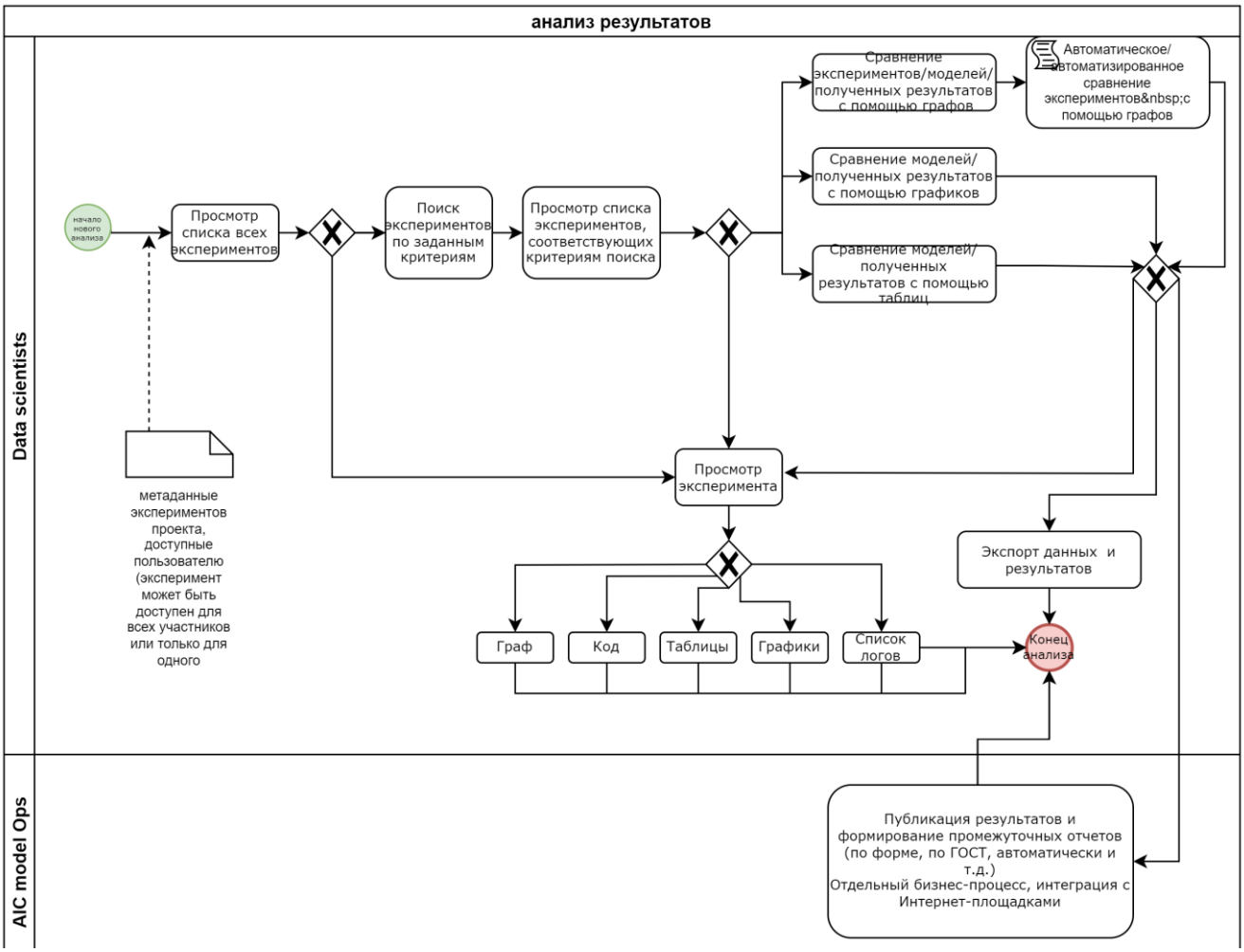


Рисунок 5.7 – BPMN диаграмма бизнес процесса «Анализ результатов экспериментов», реализуемого с помощью AIC ModelOps Platform

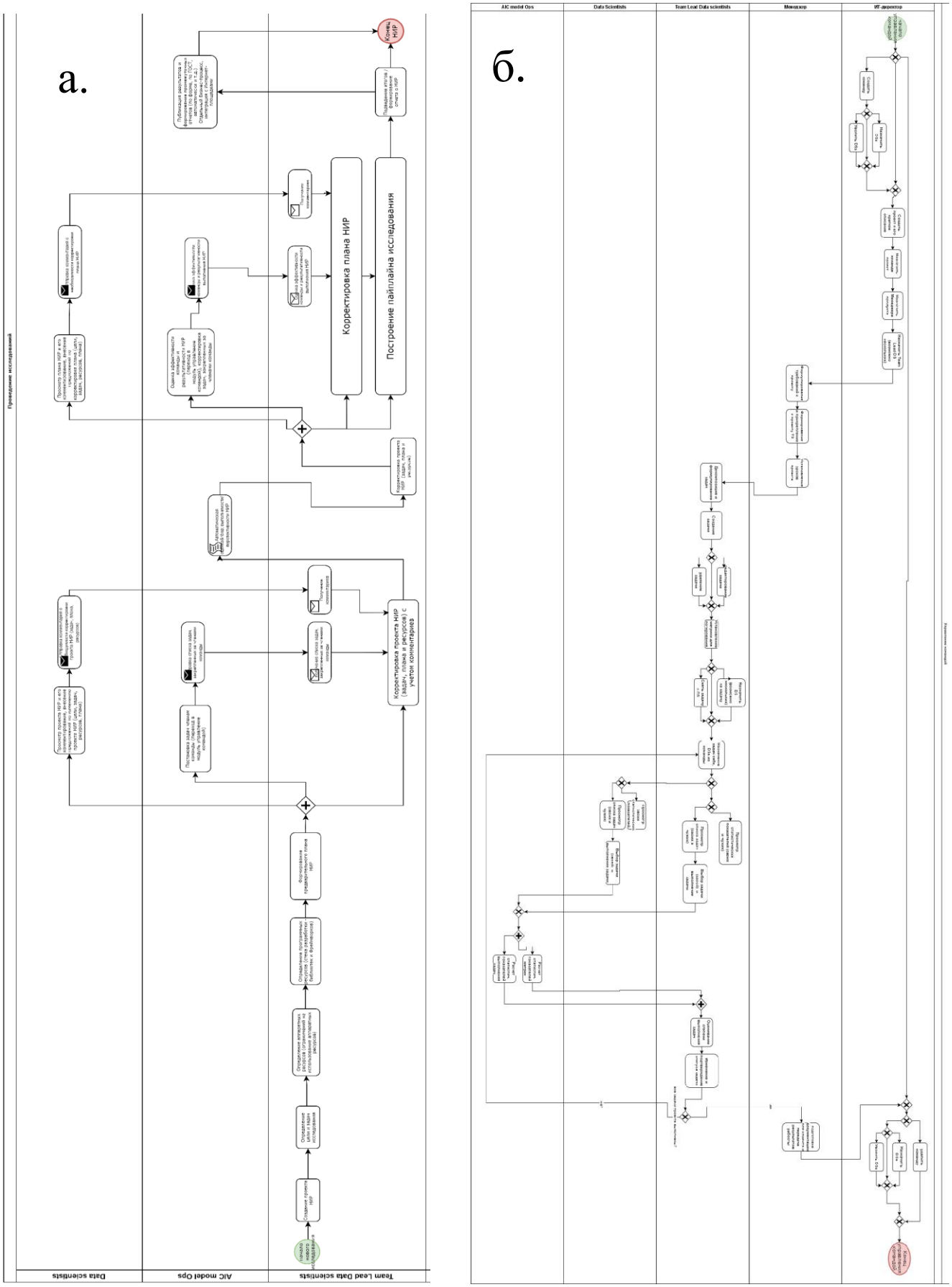


Рисунок 5.8 – BPMN диаграммы бизнес процессов, реализуемых AIC ModelOps Platform:

а. «Проведение НИР»; б. «Управление командой»

Требования к программной части: для корректной работы веб-сервиса требуется выделенный сервер или виртуальная машина с операционной системой на базе RedHat/Debian, к ним можно отнести CentOS, Ubuntu и др.

Требования к аппаратной части: процессор с архитектурой x64 (Intel с поддержкой EM64T, AMD с поддержкой AMD64), оперативная память 4096Мб и/или выше, HDD (жесткий диск) или SSD (твердотельный накопитель), объемом 80Гб и/или выше. От мощностей сервера будет зависеть скорость производимых вычислений. Здесь и далее указаны самые минимальные системные требования. Рекомендуемые для сервера — от 4 ядер и 16 ГБ ОЗУ.

Для использования AIC ModelOps Platform пользователю необходимо создать или загрузить модель ИИ и наборы данных для обучения/тестирования модели. По окончанию работы продукт позволяет пользователю получить следующие выходные данные:

1. Статистические данные об использовании вычислительных мощностей, стабильности модели, отчеты об использовании и мониторинг производительности с использованием таких метрик, как AUC, F1-score, RMSE, статистика Джини и т.д., представленные в виде информационных панелей.
2. Отчет о проведении вычислительного эксперимента (аннотированный научный отчет об исследовании).
3. История создания и изменения модели, с возможностью переключаться между определёнными ее версиями.

Новизна предлагаемых в инновационном продукт решений заключается в следующем:

1. Автоматическое онлайн и офлайн обучение неглубоких нейросетевых моделей на малых выборках.

2. Повышение доверия к ИИ и обеспечение функциональной безопасности ИИ.

Для реализации системы была выбрана архитектура «клиент-сервер», чтобы улучшить расширяемость и взаимодействие пользователя с приложением. Исходя

из этого, необходимо определиться с технологиями разработки как серверной, так и клиентской части. При выборе технологий для реализации серверной части, в первую очередь необходимо существующие языки программирования, чтобы выбрать тот, который в большем объеме удовлетворяет требованиям к системе.

В качестве ключевых факторов выбора языка программирования для серверной части были выбраны:

– популярность языка программирования;
– стоимость труда специалиста, обладающего знаниями данного языка программирования.

Для разработки серверной части приложения было решено выбрать C#.

Архитектурный стиль взаимодействия клиентской и серверной частей в приложении – REST (Representational State Transfer), который представляет собой строго согласованный набор ограниченных функциональных возможностей, которые учитываются при проектировании системы

После выбора технологии для реализации серверной части необходимо определиться с технологиями для реализации клиентской части. Современный подход к реализации веб-приложений популяризирует одностраничные клиентские приложения. Single Page Application (SPA) представляет собой веб-приложение или веб-сайт, представленный в виде единственного HTML документа в качестве оболочки для всех веб-страниц, который организует взаимодействие сервиса с пользователем с помощью динамически подгружаемых HTML, CSS, JavaScript документов.

У данного подхода имеется ряд достоинств:

– экономия времени на загрузке одних и тех же элементов на экранах приложения;
– манипулирование состоянием элементов приложения на клиентской стороне;
– экономия серверных вычислительных мощностей за счет рендера и манипуляцией с HTML-документом на стороне клиента.

На сегодняшний день все современные SPA-приложения реализованы на языке JavaScript.

Диаграммы архитектуры программного обеспечения представляют собой отличный способ сообщить, как планируется создавать программную систему или описать принципы работы уже существующей системы.

Модель C4 — это подход «сначала абстракция» к диаграммной архитектуре программного обеспечения, основанный на абстракциях, которые отражают то, как архитекторы и разработчики программного обеспечения представляют себе программное обеспечение. Небольшой набор абстракций и типов диаграмм упрощает изучение и использование модели C4.

На рисунке 5.9 представлена диаграмма системного контекста для разработанной системы.

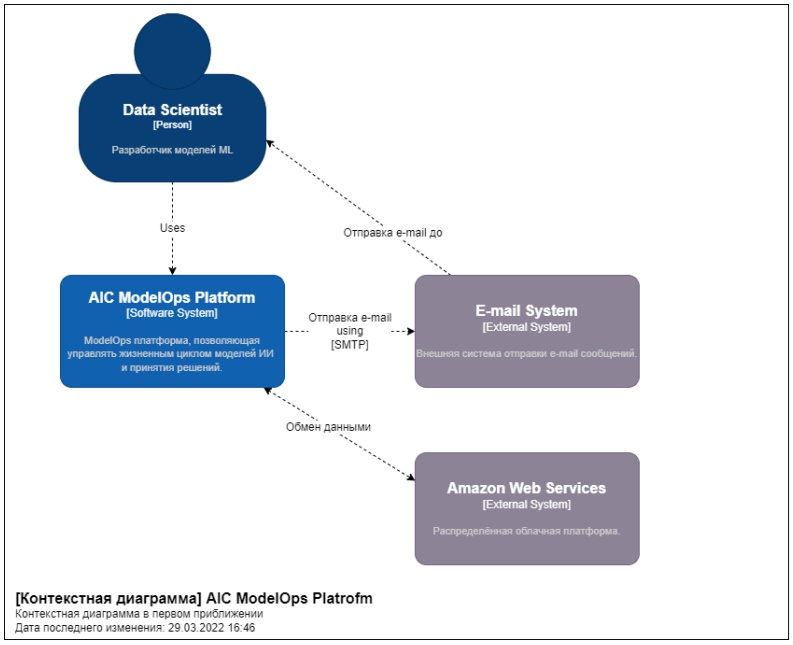


Рисунок 5.9 – Модель C4. Диаграмма системного контекста.

Архитектура AIC ModelOps Platform иллюстрируется на рисунке 5.10. AIC ModelOps Platform использует существующую распределенную облачную платформу, а также внешнюю систему отправки электронной почты для отправки уведомлений и иных информационных сообщений пользователям.

Диаграмма контейнеров (5.10) показывает составляющие элементы программной системы, а именно: контейнеры, приложения, хранилища данных,

микросервисы, одностраничные приложения и т.п. Она также показывает основные варианты технологий и то, как контейнеры взаимодействуют друг с другом. Это простая схема высокого уровня, ориентированная на технологии, которая одинаково полезна как для разработчиков программного обеспечения, так и для персонала службы поддержки и эксплуатации.

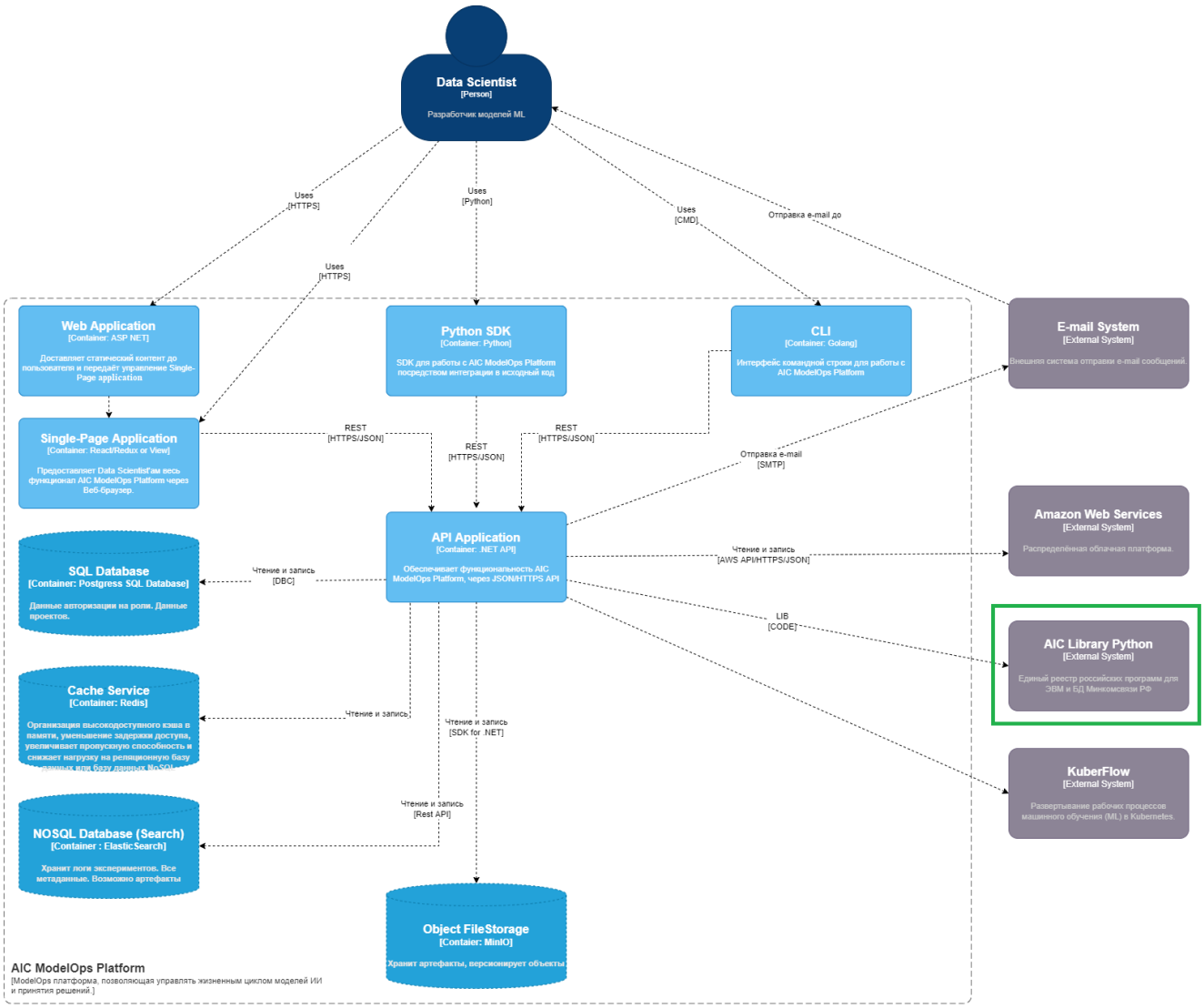


Рисунок 5.10 – Схема архитектуры AIC ModelOps Platform

Диаграмма, представленная на рисунке 5.9 показывает, что разработанная система состоит из 9-и контейнеров: веб-приложения на стороне сервера, одностраничного приложения на стороне клиента, python SDK, интерфейса для работы с командной строкой, приложения API на стороне сервера, SQL базы

данных, NoSQL базы данных, сервиса кеширования данных и объектного файлового хранилища.

Веб-приложение представляет собой ASP.NET-приложение на базе шаблона MVC, которое просто обслуживает статический контент (HTML, CSS и JavaScript), включая контент, из которого состоит одностраничное приложение. Одностраничное приложение — это приложение React/Redux/View, которое запускается в веб-браузере клиента и обеспечивает все функции системы для создания, внедрения и защиты доверенного искусственного интеллекта от угроз информационной безопасности. Кроме того, пользователи могут использовать Python SDK или CLI для доступа к некоторому функционалу программной среды. Одностраничное приложение, Python SDK и CLI используют JSON/HTTPS API, который предоставляет другое приложение ASP.NET MVC, работающее на стороне сервера. Приложение API получает информацию о пользователе из реляционной базы данных. Приложение API также связывается с NoSQL базой данных для обеспечения возможности осуществлять полнотекстовый поиск по неструктурированным данным и может взаимодействовать с объектным хранилищем, основная задача которого состоит в том, чтобы хранить данные модели, наборы данных, промежуточные результаты выполнения эксперимента. Приложение API также использует существующую систему электронной почты для отправки уведомлений пользователю. Также приложение взаимодействует с некоторыми другими внешними системами, обеспечивающими доступ к вычислительным ресурсам, а также технологиям и решениям, которые направлены на ускорение процесса разработки моделей машинного обучения и искусственного интеллекта. На рисунке 5.11 представлена диаграмма компонентов API контейнера разработанной системы.

Как видно из диаграммы API контейнер состоит из 8 контроллеров, предоставляющих точки доступа для JSON/HTTPS API, при этом практически каждый контроллер впоследствии использует другие компоненты для доступа к данным из баз данных, объектного хранилища или сервиса кеширования.

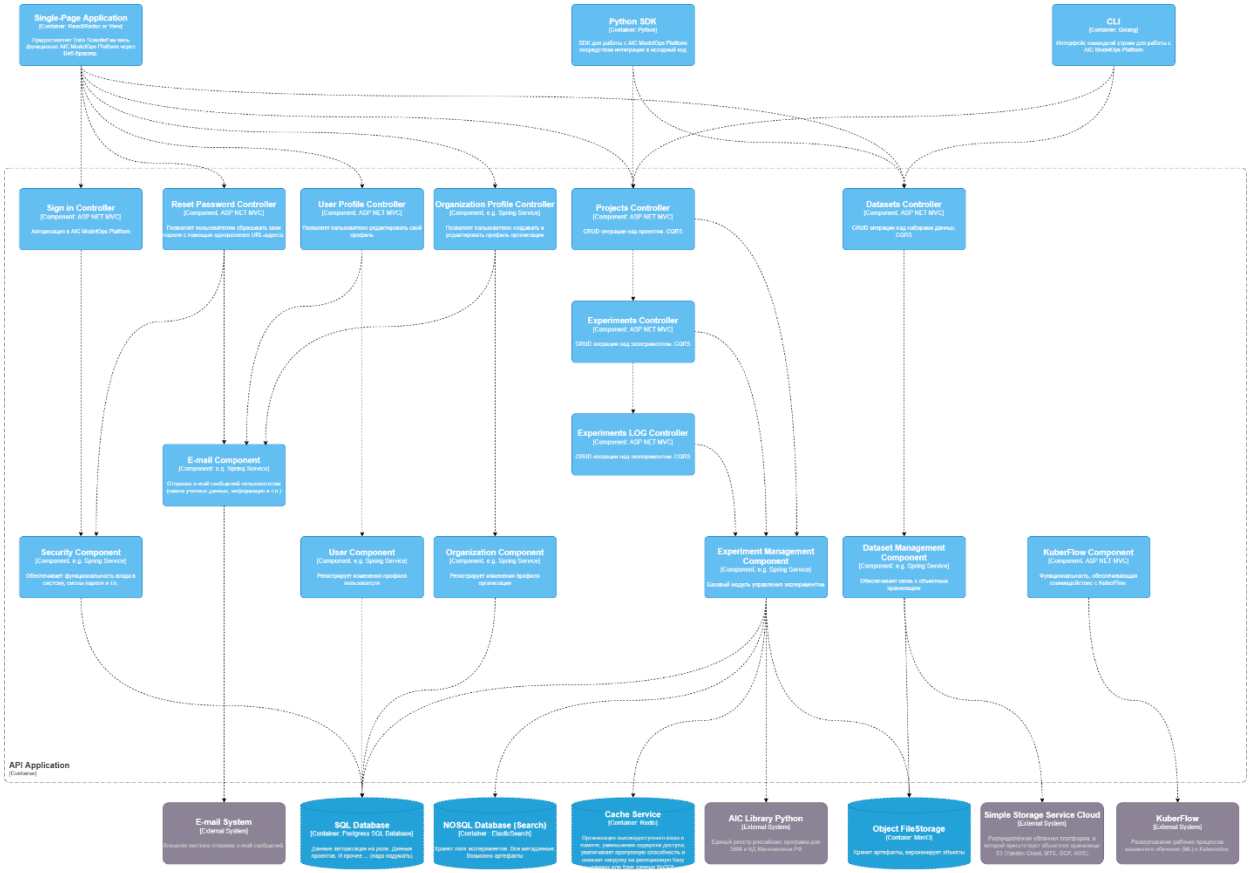


Рисунок 5.11 – Модель C4. Диаграмма компонентов. Описание API-контейнера

В качестве СУБД SQL выбрана PostgreSQL. Для описания схемы базы данных была спроектирована ER диаграмма (рисунок 5.12). ER диаграмма была спроектирована в онлайн-ресурсе draw.io. В процессе проектирования был выделен ряд основных сущностей будущей базы данных:

– organization – данная сущность содержит информацию б организации и имеет следующие поля:
  - id – код организации. Является первичным ключом (LONG ()).
  - name – имя организации (VARCHAR(45)).
  - description – описание организации (VARCHAR(1000)).
– datasets – данная сущность содержит информацию о наборах данных и имеет следующие поля:
  - id – код набора данных. Является первичным ключом
  - (LONG ()).
  - name – имя набора данных (VARCHAR(45)).

- description – описание набора данных (VARCHAR(1000)).
- size – размер набора данных (DOUBLE()).
- files_count – количество файлов в наборе данных (LONG()).
- organization_id – код организации, является внешним ключом (LONG ()).

– projects –данная сущность содержит информацию о проектах организации (один проект равносилен одному исследованию, в рамках какой-либо тематике) и имеет следующие поля:

- id – код проекта. Является первичным ключом (LONG ()).
- name – имя проекта (VARCHAR(45)).
- description – описание проекта (VARCHAR(1000)).
- privacy – является ли проект приватным (BOOL()).
- organization_id – код организации, является внешним ключом (LONG ()).

– project_settings –данная сущность содержит дополнительную информацию о проектах организации и имеет следующие поля:

- proj_setting_id – код конфигурации проекта. Является первичным ключом (LONG ()).
- project_id – код проекта, является внешним ключом (LONG ()).
- setting_name – имя конфигурации (VARCHAR(1000)).

– experiment –данная сущность содержит информацию об экспериментах проекта и имеет следующие поля:

- id – код эксперимента. Является первичным ключом (LONG ()).
- name – имя эксперимента (VARCHAR(45)).
- created_by – кем был создан эксперимент (VARCHAR(100)).
- description – описание эксперимента (VARCHAR(1000)).
- tags – список тегов эксперимента (VARCHAR(1000)).
- status_id – статус эксперимента. Является внешним ключом (LONG ()).

- created_at – дата создания эксперимента (DATE()).
- start_time – дата запуска эксперимента (DATETIME()).
- end_time – дата окончания эксперимента (DATETIME()).
- project_id – код проекта. Является внешним ключом (LONG ()).

– artefacts –данная сущность содержит информацию об артефактах проекта и имеет следующие поля:

- id – код артефакта. Является первичным ключом (LONG ()).
- experiment_id – код эксперимента. Является внешним ключом (LONG ()).
- project_id – код проекта. Является внешним ключом (LONG ()).
- minio_id – url адрес на файл в MinIO (VARCHAR(500)).

– experiment_status – данная сущность содержит информацию о статусах экспериментов и имеет следующие поля:

- id – код статуса эксперимента. Является первичным ключом (LONG ()).
- status_name – название статуса эксперимента (VARCHAR(45)).

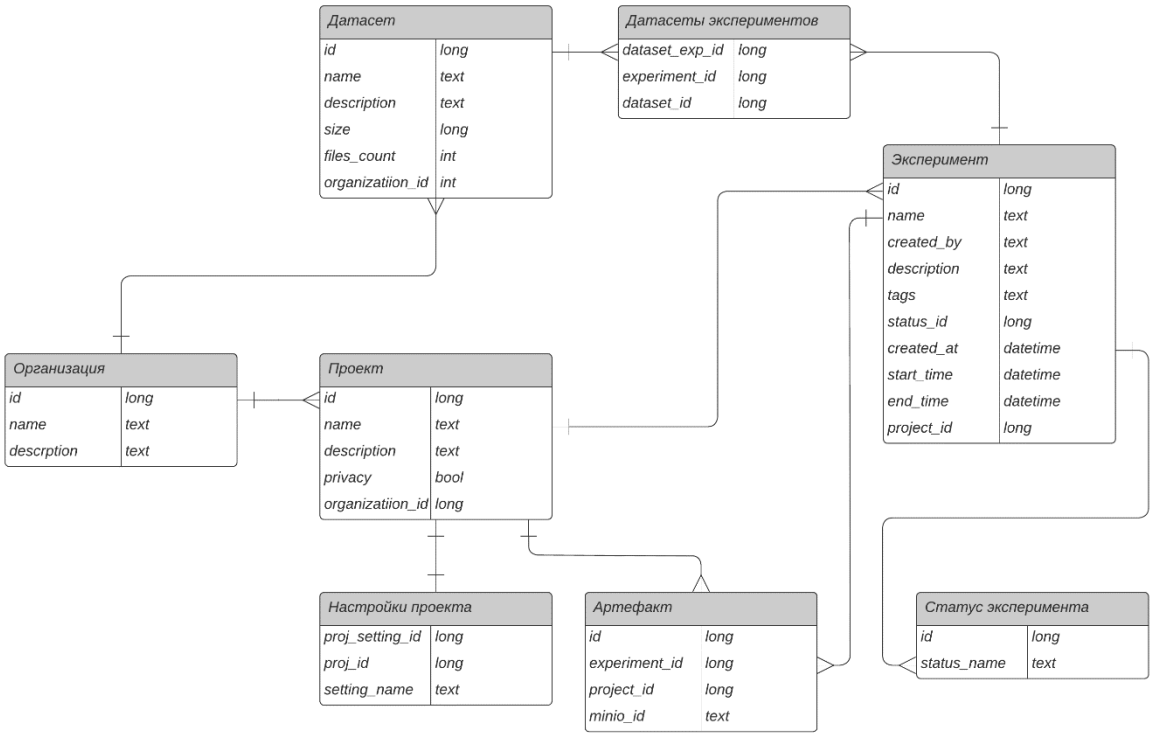


Рисунок 5.12 – ER диаграмма базы данных

Использование платформы ASP.Net MVC подразумевает использование архитектурного паттерна MVC (Model-View-Controller).

Основная цель паттерна MVC – отделение внешнего вида от собственно обработки данных, таким образом, что одна и та же модель данных может использоваться в разных представлениях. Это достигается через использование 3 типов объектов: модель данных (Model), представление (View), контроллер (Controller).

В качестве первого шага разработки была обеспечена связь между веб-интерфейсом и базой данных. В качестве СУБД использовался PostgreSQL. Для связи веб-интерфейса и БД использовался Entity Framework.

Для авторизации в системе пользователь использует форму, представленную на рисунке 5.13. В случае если пользователь еще не является зарегистрированным в системе, он может перейти на вкладку регистрации и создать аккаунт (рисунок 5.13).

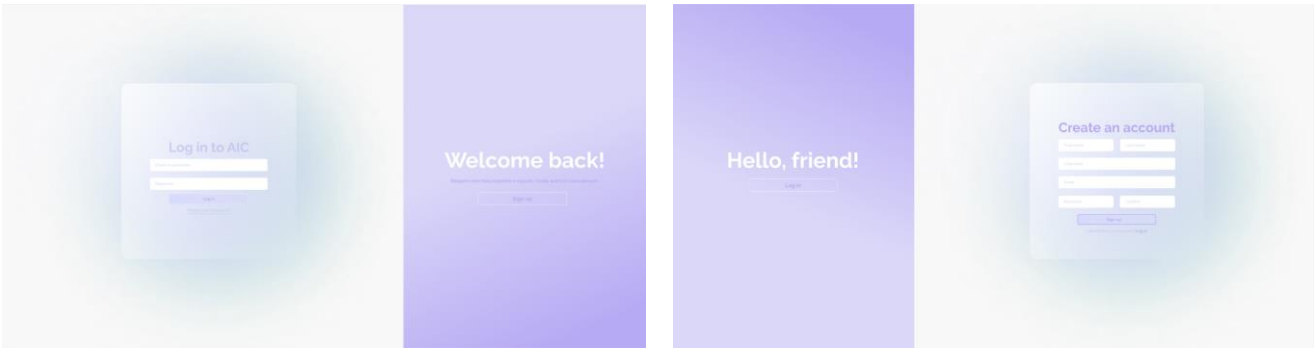


Рисунок 5.13 – Страница авторизации (слева) и страница регистрации пользователя (справа)

После авторизации в системе пользователь попадает на свою главную страницу, содержащую все рабочие пространства, в которых он задействован. После перехода в одно из рабочих пространств, пользователь попадает в список проектов, реализуемых в данном рабочем пространстве. На странице присутствует навигационная панель, которая позволяет осуществлять переход между конфигурацией и управлением различными компонентами системы, такими как:

1. Datasets (наборы данных, загруженные в систему участниками проектов).

2. Projects (страница всех проектов).

3. My results (результаты работы пользователя по тем проектам рабочего пространства, участником или владельцем которых он является).

4. Community (сообщество участников рабочего пространства).

Пользователь может перейти на его главную страницу определенного проекта (рисунок 5.14). Функционал главной страницы позволяет создавать задачи для участников проекта, отслеживать их активность, а также наблюдать последние новости по работе с данным проектом. Дополнительные функции по работе с отдельным проектом представлены в боковом меню с пятью позициями (помимо главной страницы проекта): цели проекта, таблица экспериментов (рисунок 5.15), раздел построения конвейера обработки данных (от слова «pipeline»), файлы проекта (рисунок 5.16) и настройки.

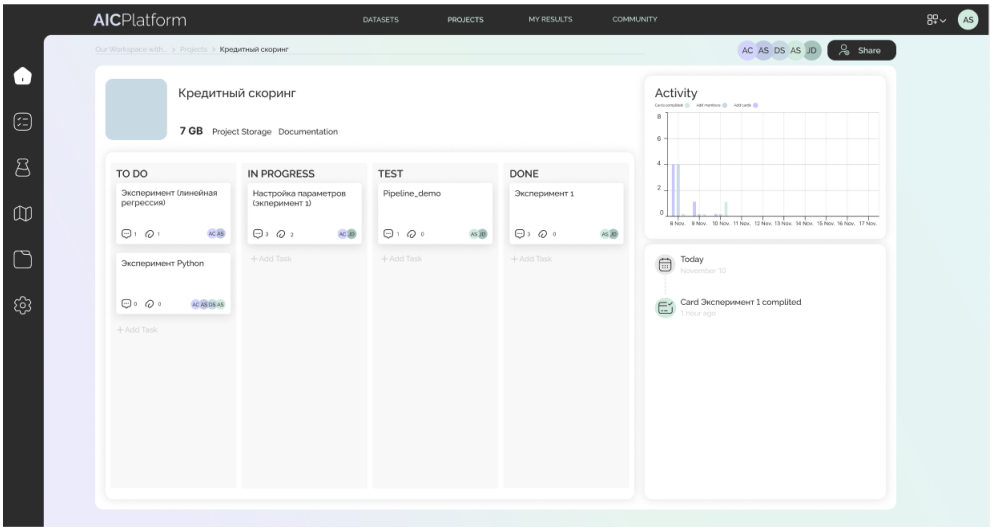


Рисунок 5.14 – Главная страница проекта

Пользователь может анализировать результаты экспериментов, связанных с выполнением проекта. При создании нового эксперимента пользователь переходит на страницу интегрированного в платформу приложения Jupyter Notebook, в сущности, являющегося средой разработки. Важным функционалом для эффективной работы является выпадающая таблица с параметрами метрик (рисунок 5.15). Данная таблица позволяет наблюдать основные метрики и гиперпараметры моделей или алгоритма обучения, а также полученные результаты. Также пользователь может воспользоваться функционалом сравнения

экспериментов по параметрам и метрикам, выбрав два или более экспериментов в таблице.

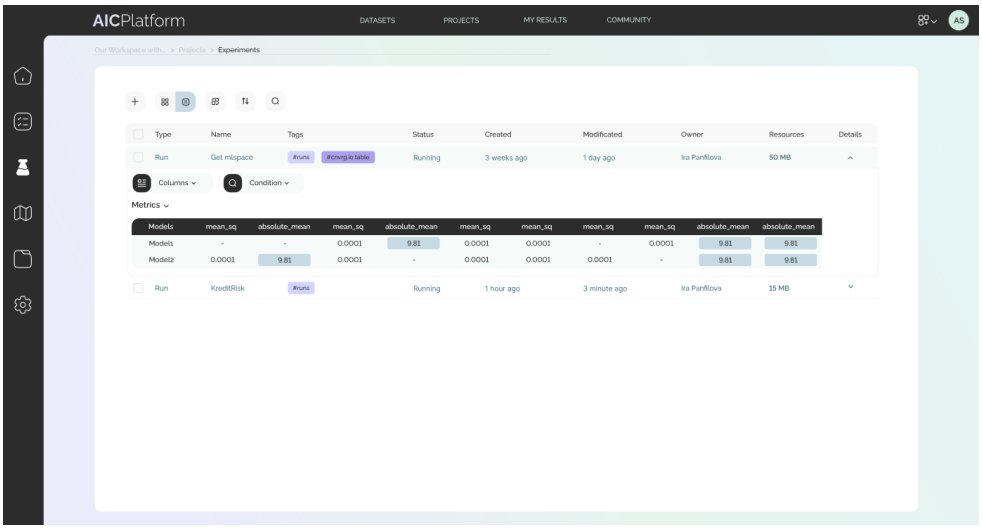


Рисунок 5.15 – Таблица экспериментов проекта

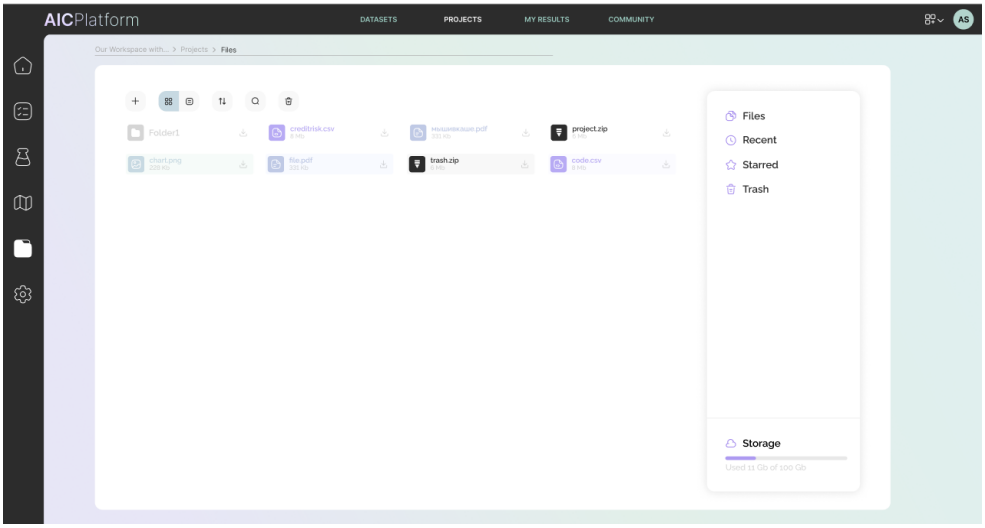


Рисунок 5.16 – Файловое хранилище проекта

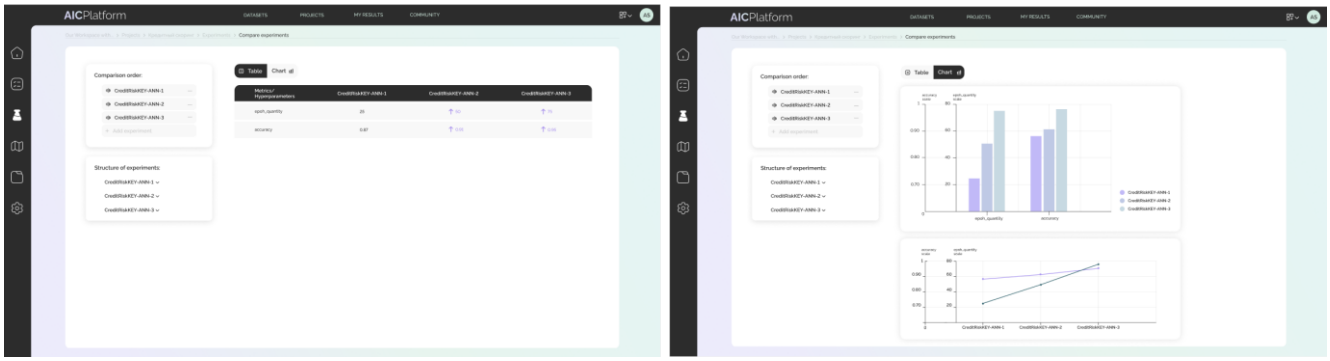


Рисунок 5.17 – Анализ результатов эксперимента: табличное (слева) и графическое (справа) сравнение экспериментов

Пользователь может переходить непосредственно к проектированию составных элементов конвейера обработки данных (исполняемых блоков кода), добавлять которые позволяет функциональное меню (рисунок 5.18). Каждый новый составной элемент конвейера обработки данных может быть представлен одним из трех блоков:

1. Data (блок работы с данными).
2. Exec (блок работы с исполняемым файлом).
3. Deploy (блок развертывания приложения).

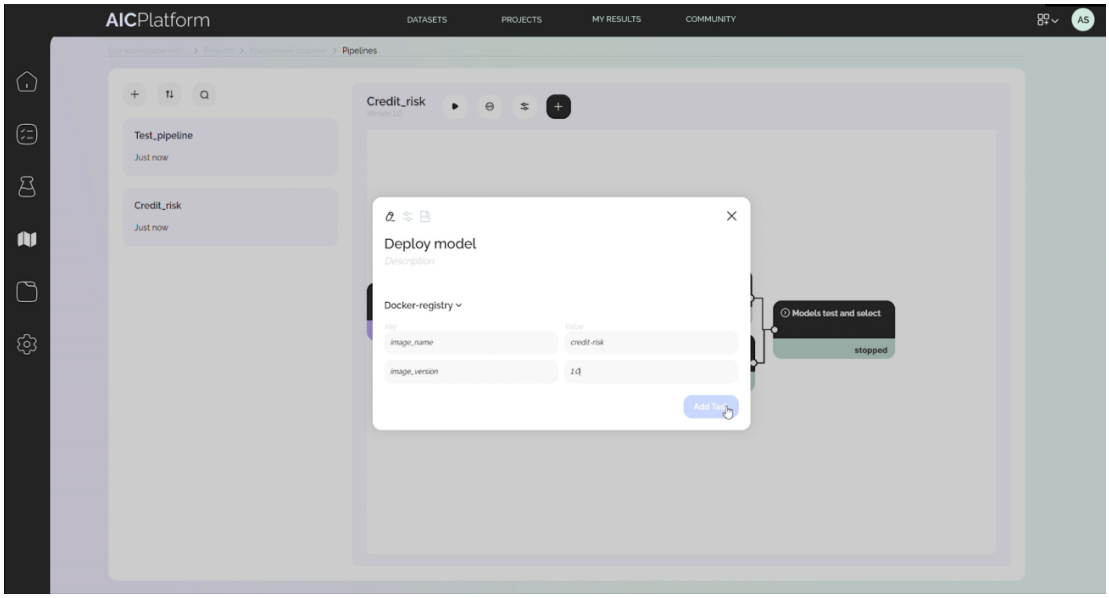


Рисунок 5.18 – Процесс добавления блока типа Deploy

Каждый добавленный в конструкторе блок можно соединять друг с другом (рисунок 5.19). Связи позволяют передавать выходные данные одного блока на вход другого, что обеспечивает сквозное выполнение кода обучения и тестирования моделей в цепочке обработки данных.

Важно отметить, что конструктор конвейеров обработки данных поддерживает возможность версионирования цепочки из блоков обработки данных, позволяя пользователям проектировать приложение, используя разные подходы к его построению.

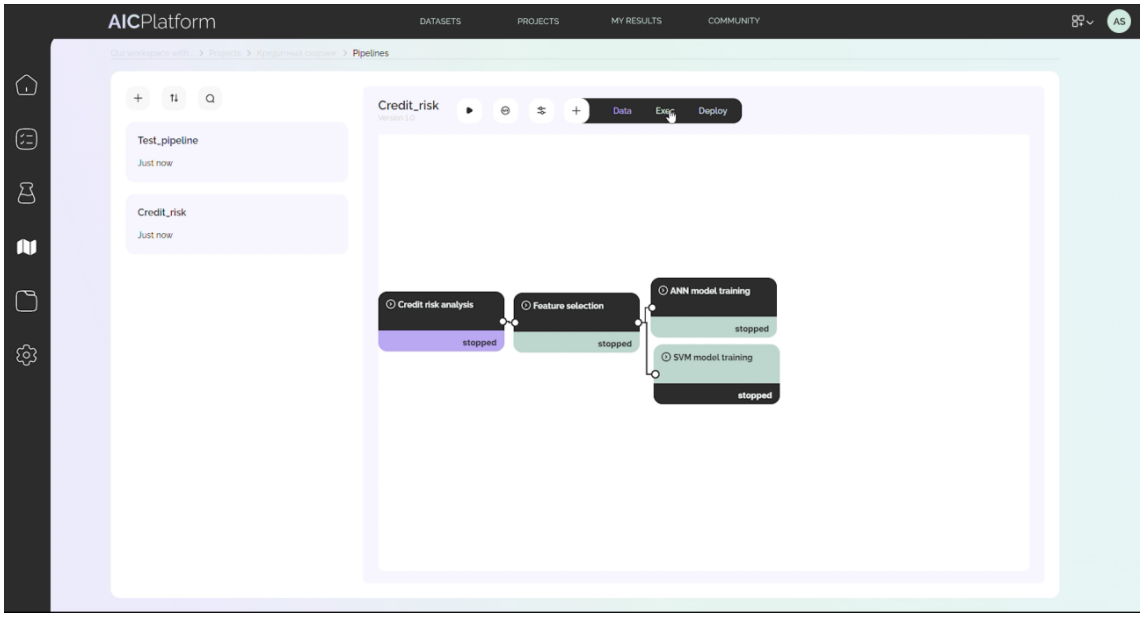


Рисунок 5.19 – Процесс конструирования конвейера обработки данных

После нажатия кнопки запуска результат исполнения конвейера обработки данных выводится на экран в виде сравнительной таблицы, как показано на рисунке 5.20. Информационное окно демонстрирует основные метрики, по которым сравниваются варианты построения моделей машинного обучения, что позволяет выбрать ту реализацию модели, которая в будущем может быть развернута в инфраструктуре заказчика.

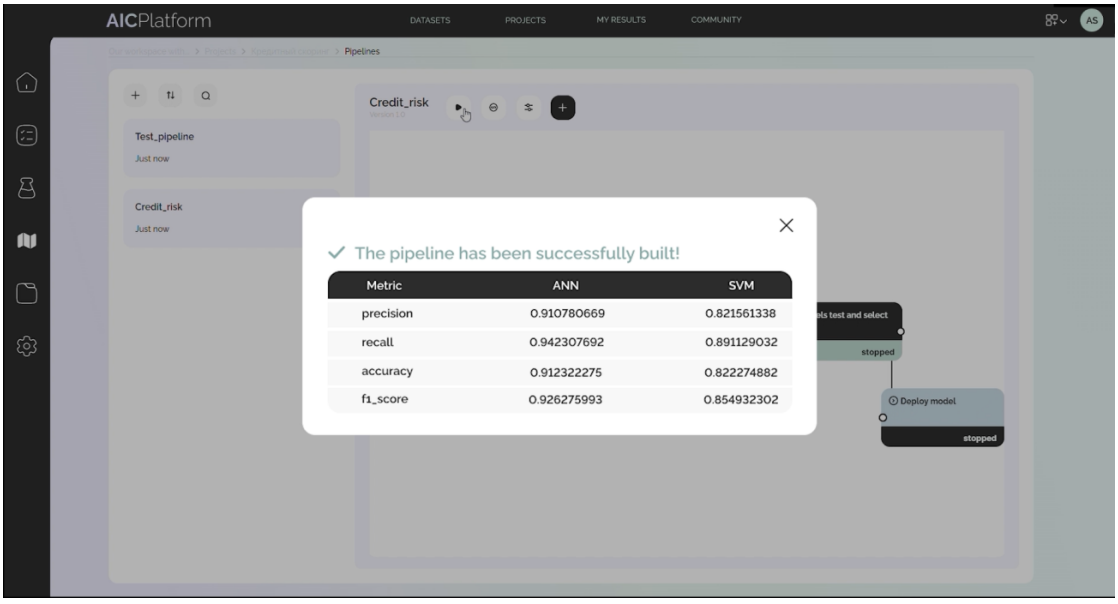


Рисунок 5.20 – Информационное окно об итогах выполнения обработки данных

Дополнительной возможностью работы с уже развернутыми реализациями моделей является таблица мониторинга моделей, представленная на рисунке 5.21. Таблица позволяет отслеживать дрейф данных, а также концептуальный дрейф, предупреждая разработчиков о возможных изменениях в структуре функционирования модели ИИ. Дополнительно предусмотрена возможность отката модели для возможной доработки в случае серьезных изменений в процессах ее функционирования.

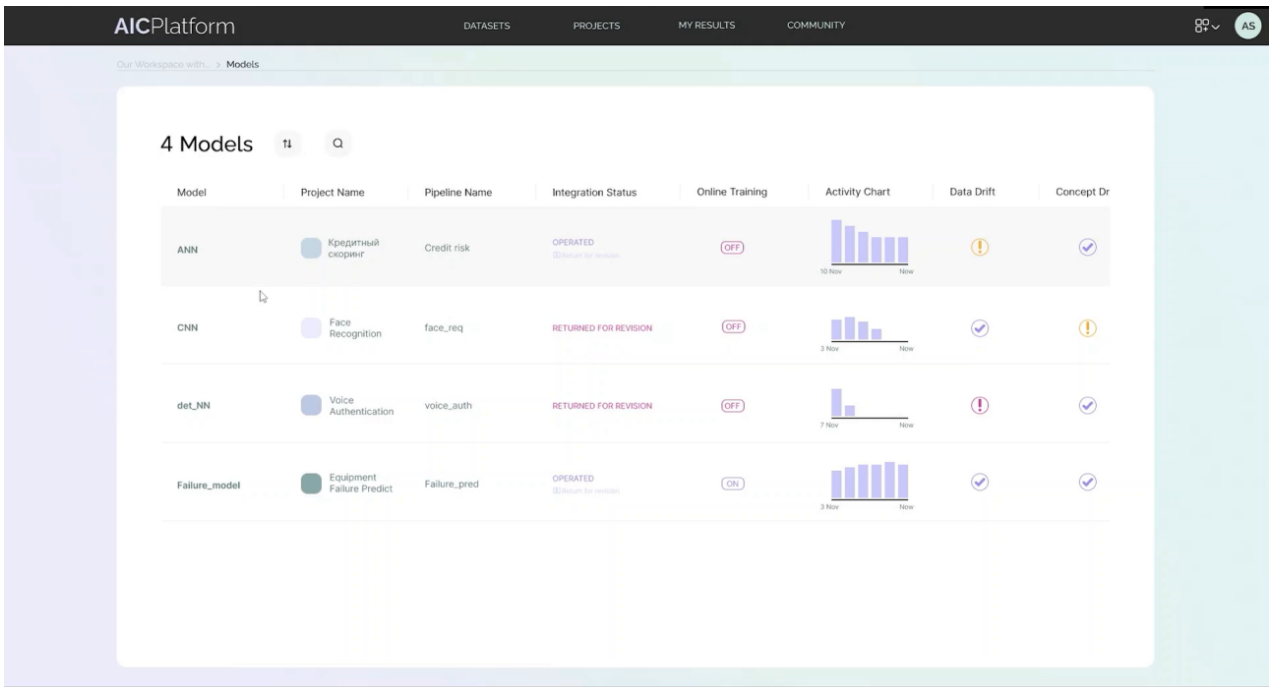


Рисунок 5.21 – Таблица мониторинга моделей ИИ

AIC ModelOps Platform поставляется в различной комплектации, в зависимости от выбранных заказчиком модулей, каждый из которых содержит различный функционал:

1. Базовый модуль “Управление экспериментом”:
    1.1. Интеграция с существующими библиотеками и фреймворками
    1.2. Функции отслеживания эксперимента (Model management + model monitoring)
    1.3. Возможность установки в сетевой инфраструктуре заказчика
    1.4. Инструменты для командной работы исследователей
    1.5. Интеграция с базами данных

1.6. Импорт из других источников данных

2. Модуль “Конструктор моделей ИИ”:
   2.1. Визуальный редактор моделей (глубоких сетей и не только)
   2.2. Low-code интерпретатор
   2.3. Переиспользование моделей
3. Модуль “Конструктор конвейеров обработки данных ”:
   3.1. CI/CD модели из эксперимента
   3.2. Визуальный конструктор конвейеров обработки данных
   3.3. Визуальный мониторинг данных и модели (принимаемые решения и входные данные, которые к нему привели)
   3.4. Определение дрейфа данных с помощью простых метрик (минимум, максимум, уникальные значения, среднее и отклонение)

4. Модуль “Диагностика”:
   4.1. Расширенный мониторинг производительности и стабильности модели с помощью множества метрик, автоматический анализ проблем
   4.2. Определение дрейфа моделей (дрейфа концепций) с помощью более сложных метрик (сравнение функций плотности распределения и статистических моментов признаков, сравнение корреляционных матриц признаков)
   4.3. Устойчивость к дрейфу моделей (автоматическое переобучение модели по расписанию, предупреждение постепенного дрейфа с помощью онлайн-обучения в процессе функционирования модели в реальных условиях)
   4.4. Непрерывное повышение надежности решений (онлайн-обучение в процессе функционирования модели в реальных условиях)

5. Модуль “Управление доверием”:
   5.1. Устойчивость к состязательным атакам и зондированию моделей
   5.2. Отчуждение ИИ от принятия решений
   5.3. Определение важности признаков или информативности данных

5.4. Объяснение причинно-следственных связей решений/данных

6. Модуль “Интерактивность”:

6.1. Образовательные кейсы для студентов

6.2. Подсказки для неопытных пользователей

AIC ModelOps Platform включает кейсы с типовыми моделями ИИ для решения определенных задач из нефтегазовой и финансовой отраслей. Основные потребители результатов – крупные, средние и проектные организации, перед которыми стоят сложные научно-технические оптимизационные и аналитические задачи, а также научно-исследовательские институты. К дополнительному сегменту относятся образовательные организации.

5.5 Разработка биометрических систем аутентификации и непрерывного мониторинга пользователей

Результаты работы использованы в проектно-конструкторской деятельности ООО «Системы информационной безопасности» при проектировании платформы информационного обмена нового поколения с функционалом по обеспечению и поддержанию заданного уровня доверия в недоверенной среде. В рамках данного проекта разработан прототип системы высоконадежной биометрической аутентификации по лицу и клавиатурному почерку для контроля доступа, обладающих повышенным уровнем устойчивости к состязательным атакам, а также высоким уровнем защищенности знаний искусственного интеллекта, в том числе, биометрических данных пользователей от компрометации. В основу системы легли следующие положения:

1. Концепции защищенного исполнения процедур высоконадежной биометрической аутентификации от исследования и компрометации знаний.

2. Модели корреляционных нейронов и модели нейросетевого преобразователя биометрия-код на их основе, а также алгоритмов их автоматического синтеза и обучения на малых выборках данных.

3. Методов и алгоритмов высоконадежной биометрической аутентификации в защищенном режиме исполнения с обеспечением защиты биометрических данных от компрометации.

В качестве признаков, извлекаемых на основе анализадвумерного изображения лица, использовались аналогичные характеристики, из тех, что описаны в работах [3, 54, 103].

Также результаты диссертации использованы в рамках проектно-конструкторской деятельности ООО «КРАФТ ЛАБ» при разработке модуля непрерывного мониторинга и распознавания личности пользователя по особенностям работы с компьютерной мышью. Процесс верификации личности пользователя основан на анализе большого количества данных о перемещениях мыши. Используемые признаки основаны на оценке тремора курсора мыши и времени перемещения курсора мыши между элементами интерфейса (рисунок 5.22), а также использовании закона Фиттса [121]. На данный момент известны методы анализа траекторий движения мыши с целью распознавания попыток удаленного доступа и подмены авторизованного пользователя, однако эти методы не позволяют связать образ пользователя с длинным паролем [29, 43, 45].

Для тестирования системы разработан симулятор, где пользователи работают с картой местности (перемещают карту, выбирают объекты, изменяют масштаб). Собран набор данных 186 пользователей с использованием идентичного программного и аппаратного обеспечения.

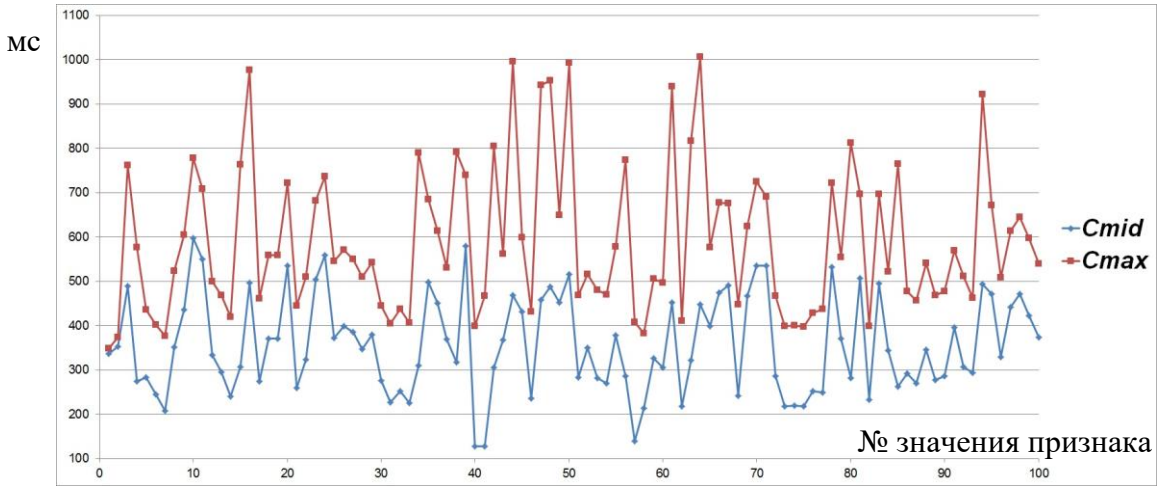


Рисунок 5.22 – Значения двух признаков C_{max} и C_{mid} одного из испытуемых в миллисекундах

Решено использовать 12 признаков, которые вычисляются из координат пути перемещения курсора между каждой парой элементов интерфейса. Корреляция между данными признаками значительно меньше единицы, но даже для наиболее информативного признака площади пересечения функций плотностей вероятности значительны, что говорит о том, что признаки мыши малоинформативны. Однако при многократном измерении признаков в процессе совершения пользователем перемещений мышью разделение известных классов все-таки выполняется (рисунки 5.23-5.24).

Рисунок 5.23 – Расчет интегральных показателей о различии пользователей в пространстве предложенных признаков на основе стратегии Байеса

При разработке программного модуля использована концепция альтернативных сценариев авторизации [51].

Использование полученных в работе результатов на предприятии ООО «Открытый код» позволило спроектировать и разработать функционал для многофакторной биометрической аутентификации при доступе в закрытые зоны с частотой ошибок «ложного отказа» в доступе менее 6% и частотой ошибок «ложного допуска» менее 0,01% по результатам предварительного тестирования, а также обеспечить защиту биометрических данных пользователей от компрометации.

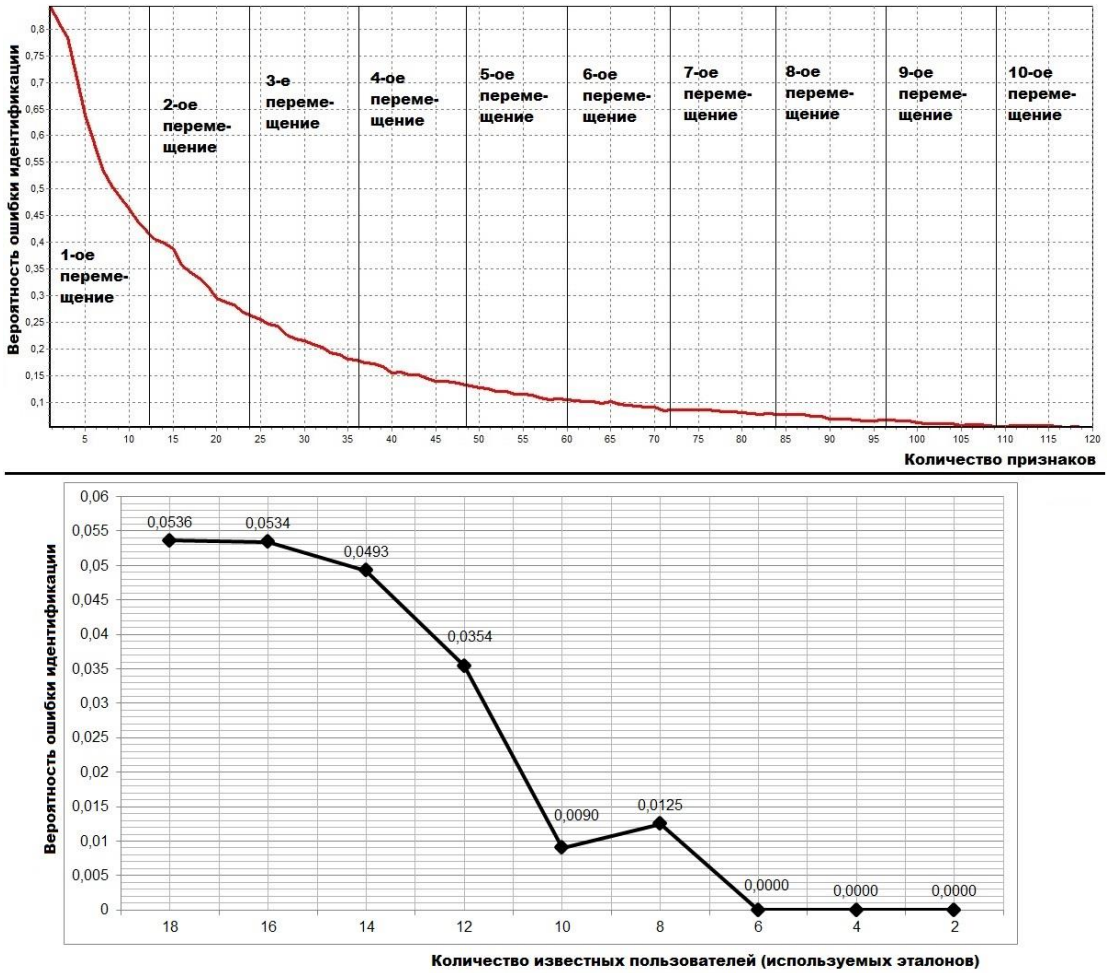


Рисунок 5.24 – Результаты эксперимента: вероятность ошибки идентификации 18 пользователей при различном количестве признаков (перемещений) – вверху; вероятность идентификации различного количества пользователей при осуществлении 10 перемещений (120 признаках) – внизу

В основу системы были положены следующие технические предложения:

– концепция защищенного исполнения процедур высоконадежной биометрической аутентификации от исследования и компрометации знаний;
– модель корреляционных нейронов, модель нейросетевого ПБК на основе корреляционных нейронов, алгоритмы автоматического синтеза и обучения на малых выборках данных нейросетевых преобразователей биометрия-код на основе корреляционных нейронов;
– методы высоконадежной биометрической аутентификации по рукописным и голосовым паролям в защищенном режиме исполнения с обеспечением защиты биометрических данных от компрометации.

5.6 Использование результатов в области медицины

Результаты работы применялись в рамках инструментально-лабораторных обследований и тестирования пациентов на аппаратно-программном комплексе в Центре здоровья поликлиники №1 БУЗОО МСЧ№4, в виде:

1. Модели корреляционных нейронов, анализирующих корреляционные связи между признаками вместо признаков, а также робастного алгоритма автоматического синтеза и обучения сетей корреляционных нейронов на малых выборках данных.
2. Адаптивной нейро-иммунной модели искусственного интеллекта и алгоритмов ее обучения на малых выборках данных.
3. Технологии автоматического синтеза и обучения нейросетевых моделей доверенного искусственного интеллекта.

На базе указанных технических предложений, а также с использованием алгоритмов кластеризации данных был разработан программный модуль для анализа результатов обследований пациентов в обезличенном виде, проводимых в Центре здоровья. Результаты обследований представляют собой данные, регистрируемые приборами Валеоскан, Кардиовизор, Медасс, а также рекомендации врача. Разработанный программный модуль позволяет:

1. Выявлять условные категории пациентов со схожими проблемами со здоровьем. На основании алгоритма кластеризации результатов обследований были составлены типовые шаблоны рекомендаций для каждой условной категории пациентов.
2. Классифицировать пациентов (с использованием положений диссертации), вновь проходящих обследование в Центре здоровья, по условным категориям и предоставлять врачу шаблон типовых рекомендаций, который врач по своему усмотрению может редактировать для каждого случая отдельно (добавлять, удалять, изменять рекомендации в зависимости от особенностей, показаний и истории болезни пациента). Это позволило

оптимизировать работу врача, сократив время, затрачиваемое врачом на подготовку рекомендации пациенту (набор текста на компьютере, формулирование заключения), и соответственно общее время приема пациента.

5.7 Внедрение в учебный процесс

Предложенные в диссертации модель, методы и алгоритмы, а также программно-аппаратный комплекс были задействованы на кафедре «Комплексная защита информации» для подготовки студентов по направлению «Информационная безопасность» (бакалавриат), а также по специальности «Безопасность информационных технологий в правоохранительной сфере» (магистратура) в рамках следующих дисциплин: «Биометрия и защита информации», «Машинное обучение в приложениях биометрии», «Защищенное исполнение искусственного интеллекта», «Доверенный искусственный интеллект», «Основы информационной безопасности», «Распознавание образов». Кроме того, результаты докторской диссертации Сулавко А.Е. активно используются в рамках выполнения научно-исследовательских работ университета, а также выпускных работ студентов.

Также указанные результаты используются при разработке программы подготовки магистратуры СПбГЭТУ «ЛЭТИ» «Безопасность и этика искусственного интеллекта» в рамках направления 01.04.01 «Информатика и вычислительная техника». Предложенные в докторской диссертации Сулавко А.Е. положения легли в основу следующих дисциплин: «Доверенный искусственный интеллект», «Защищенное исполнение искусственного интеллекта», «Машинное обучение в приложениях биометрии». Кроме этого, результаты диссертационной работы соискателя применяются при выполнении квалификационных и научно-исследовательских работ сотрудников и студентов.

5.8 Анализ результатов. Выводы

Настоящая глава посвящена практическому внедрению результатов и описанию разработанной технологии. Также в рамках главы обозначены перспективные направления для дальнейших исследований. Предложенная технология, а также ее элементы, к которым относятся концепция, модели, методы и алгоритмы, были использованы на следующих предприятиях: ООО «Открытый код», ООО «Системы информационной безопасности», ООО «АИ ЗИОН», ООО «КРАФТ ЛАБ», БУЗОО "Медико-санитарная часть № 4", где они использовались в проектно-конструкторской деятельности и также в учебный процесс ФГАОУ ВО СПбГЭТУ «ЛЭТИ» и ФГАОУ ВО «ОмГТУ». Результаты применялись при разработке национального стандарта ГОСТ Р «Искусственный интеллект. Нейросетевые алгоритмы в защищённом исполнении. Автоматическое обучение нейросетевых моделей на малых выборках в задачах классификации» под руководством соискателя на базе ОмГТУ, стандарт поставлен в план на 2022 техническим комитетом № 164 «Искусственный интеллект».

Имеются также другие разработанные программные продукты, в которых применялись отдельные элементы результатов исследования [115, 130-135].

Возможные направления будущих исследований и развития темы:

- создание и поддержка на всех этапах жизненного цикла систем доверенного искусственного интеллекта;
- разработка систем защищенного исполнения алгоритмов ИИ на объектах критической информационной инфраструктуры;
- создание объяснимого и этического искусственного интеллекта.

Заключение

В диссертационной работе на основе выполненного автором исследования решена актуальная научная повышения надежности многофакторной биометрической аутентификации и защищенности биометрических систем от компьютерных атак на основе технологии защищенного исполнения нейросетевых моделей и алгоритмов ИИ, имеющая важное хозяйственное значение с точки зрения обеспечения информационной безопасности компьютерных ресурсов и конфиденциальных данных, а также знаний, моделей и алгоритмов ИИ.

Получены следующие основные результаты:

1. Разработана концепция защищенного исполнения нейросетевых алгоритмов ИИ, которая позволяет сформировать устойчивость модели к извлечению знаний. Это достигается путем преобразования корреляционных связей между признаками в высокоинформативные мета-признаки Байеса-Минковского, которые сложно фальсифицировать. Установлено, что один мета-признак может содержать в 2-3 раза больше информации, чем содержится суммарно в паре исходных признаков, от которых он порожден. Доказано, что корреляция между признаками увеличивает количество информации о классифицируемом образе. Предложены отображения для перехода в пространства мета-признаков Байеса-Минковского, что не требует хранения какой-либо дифференциальной информации о параметрах классов образов. Свойства пространств мета-признаков исследованы экспериментально.
2. Разработаны модель корреляционных нейронов и модель нейросетевого преобразователя биометрия-код на их основе, анализирующие корреляционные связи между признаками вместо признаков, а также робастной алгоритм их автоматического синтеза и обучения на малых выборках. Это позволило повысить защищенность биометрических данных от компрометации и длину ключа, связываемого с биометрическими образами субъектов, снизить вероятность

ошибок биометрической аутентификации в защищенном режиме исполнения и повысить устойчивость ИИ к состязательным атакам.

3. Разработаны адаптивная нейро-иммунная модель и алгоритмы ее пакетного обучения с учителем и онлайн-обучения с подкреплением, позволяющие предупредить или снизить влияние концептуального дрейфа, даже если исходная обучающая выборка недостаточно репрезентативна или незначительна в объеме. Алгоритм обучения с учителем позволяет сформировать врожденный иммунитет модели, позволяющий разделять входные образы на два класса. В процессе функционирования модель адаптируется к изменению данных, используя алгоритм онлайн-обучения с подкреплением, в результате чего формируется приобретенный иммунитет, корректирующий решения модели в спорных случаях.

4. Разработаны методы и алгоритм высоконадежной многофакторной аутентификации на основе рукописных и голосовых образов, а также акустических образов уха с обеспечением защиты биометрических данных от компрометации. Новизна заключается в использовании нового типа биометрических данных – акустических параметров ушного канала, получаемых методом эхолокации, а также учете информативности, стабильности и приоритезации признаков, совместным использованием НПБК и нейро-иммунной модели, способе кепстрального анализа сигналов. Акустические образы уха не компрометируются в естественной среде, так как фотография уха неинформативна для синтеза состязательных примеров. Предложенные методы и алгоритм дают более низкий процент ошибок по сравнению с известными мировыми аналогами: FRR = 0,12 при $FAR < 10^{-14}$ и FRR = 0,03 при $FAR < 10^{-10}$.

5. Разработана технология автоматического синтеза и обучения нейросетевых моделей доверенного ИИ для высоконадежной биометрической аутентификации и других ответственных приложений ИИ. На базе технологии разработана линейка программных продуктов AIConstructor и первая редакция национального стандарта ГОСТ Р «Искусственный интеллект. Нейросетевые алгоритмы в защищённом исполнении. Автоматическое обучение нейросетевых моделей на малых выборках в задачах классификации». Это первый стандарт, который

регламентирует создание и обучение нейросетевых моделей ИИ, исполняемых в защищенном от исследования режиме. Выполнено 8 внедрений результатов на 7 предприятиях.

Перспективы дальнейшего развития темы. Все более актуальными становятся вопросы создания систем доверенного ИИ, минимизации рисков, связанных с внедрением и поддержкой систем ИИ, защиты от компьютерных атак и обеспечения функциональной безопасности ИИ. Основные перспективы развития темы состоят именно в данном направлении. Защищенное исполнение алгоритмов ИИ необходимо на объектах критической информационной инфраструктуры. Перспективным направлением для развития темы также является управление жизненным циклом ИИ. Разработанный аппарат может применяться для сокращения объемов обучающей выборки, формирования инструментов повышения объяснимости решений, для обнаружения и корректировки дрейфа моделей ИИ.

Кроме того, развитие и применение разработанных концепции, моделей и алгоритмов видится перспективным в области стандартизации ИИ в тех отраслях, в которых ставятся повышенные требования к безопасности.

Список сокращений

БЧХ – коды Боуза – Чоудхури – Хоквингема

ВИ – врожденный иммунитет

ИБ – информационная безопасность;

ИИ – искусственный интеллект;

ИИС – искусственная иммунная сеть (система);

ИНР – интервал неопределенности решения

ИНС – искусственная нейронная сеть;

ЛЧМ – линейная частотная модуляция

НКП – нейро-корреляционный преобразователь (нейросетевой преобразователь образов в код на основе корреляционных нейронов)

НПБК – нейросетевой преобразователь биометрия-код

ОАЭ – отоакустическая эмиссия

ПБК – преобразователь биометрия-код

ПИ – приобретённый иммунитет

ПФС – психофизиологическое состояние

ТК – технический комитет

ФЗ – федеральный закон

ФС – функциональное состояние

ЧОТ – частота основного тона

AIC – artificial intelligence constructor, AIConstructor

AINE artificial immune network

AIRS – artificial immune recognition system

API – application programming interface

AUC – area under curve

CLI – command line interface

CNN – convolutional neural network

CSS – Cascading Style Sheets

EER – Equal Error Rate (коэффициент равной вероятности ошибок)

ER – entity relationship

FAR – false access rate (ошибка «ложного допуска»)

FRR – false reject rate (ошибка «ложного отказа»)

GDPR – General Data Protection Regulation

HTML – hypertext markup language

HTTP – hypertext transfer protocol

HTTPS – hypertext transfer protocol secure

IEC – International Electrotechnical Commission

ISO – International Organization for Standardization

JSON – JavaScript object notation

JTC – joint technical committee

k-NN – k-nearest neighbor

LDA – latent Dirichlet allocation

MAC – mean accuracy

MFCCs – mel-frequency cepstral coefficients

MVC – model-view-controller

RMSE – root mean square error

ROC – receiver operating characteristic

SC – standards committee

SDK – software development kit

SQL – structured query language

STFT – short-time Fourier transform (быстрое оконное преобразование Фурье)

SVM – support vector machine

Список литературы

1. Алгоритмы тестирования биометрико-нейросетевых механизмов защиты информации : моногр. / Б. С. Ахметов, В. И. Волчихин, А. И. Иванов, А. Ю. Малыгин. – Алматы : Изд-во КазНТУ им. К. И. Сатпаева, 2013. – 152 с.

2. Анализ методов распознавания образов человека по особенностям электроэнцефалограмм (обзор) / А. Е. Сулавко, А. И. Куприк, М. А. Старков, Д. Г. Стадников // Вопросы защиты информации. – 2018. – № 4 (123). – С. 36–46.

3. Аутентификация пользователей компьютера на основе клавиатурного почерка и особенностей лица / П. С. Ложников, А. Е. Сулавко, Е. В. Бурая, В. Ю. Писаренко // Вопросы кибербезопасности. – 2017. – № 3. – С. 24–34. – DOI: 10.21681/2311-3456-2017-3-24-34.

4. Ахметвалеев А.М., Катасёв А.С. Инструментальный комплекс программ для автоматизации определения функционального состояния человека // Автоматизация процессов управления. 2018. № 2 (52). С. 112-121.

5. Ахметвалеев А.М., Катасёв А.С., Подольская М.А. Модель коллектива нейронных сетей и программный комплекс для определения функционального состояния человека // Прикаспийский журнал: управление и высокие технологии. 2018. № 1 (41). С. 69-85.

6. Бабенко Л.К., Русаловский И.Д. Масштабирование цифровых изображений с применением гомоморфного шифрования // Вопросы кибербезопасности. 2021. № 3 (43). С. 2-10.

7. Бабенко Л.К., Шумилин А.С., Алексеев Д.М. Алгоритм обеспечения безопасности конфиденциальных данных медицинской информационной системы хранения и обработки результатов обследований // Известия ЮФУ. Технические науки. 2020. № 5 (215). С. 6-16.

8. Бабенко Л.К., Шумилин А.С., Алексеев Д.М. Алгоритм обеспечения защиты конфиденциальных данных облачной медицинской информационной системы // Известия ЮФУ. Технические науки. 2021. № 5 (222). С. 120-134.

9. Баринов А.И., Баринова А.О., Катасёв А.С. Нейросетевая сверточная модель обнаружения нарушений масочного режима в общественных местах // Вестник НЦБЖД. 2021. № 4 (50). С. 39-45.

10. Безяев, А. В. Биометрико-нейросетевая аутентификация: обнаружение и исправление ошибок в длинных кодах без накладных расходов на избыточность / А. В. Безяев ; Пензенский гос. ун-т. – Пенза : Изд-во ПГУ, 2020. – 40 с. – ISBN 978-5-907262-59-1.

11. Безяев, А. В. Нейросетевая молекула: механизм направленной квантовой коррекции большого числа ошибок длинного кода высокоразмерных биометрических образов / А. В. Безяев // Безопасность информационных технологий : сб. науч. ст. по материалам I Всерос. науч.-техн. конф. (Пенза, 24 апр. 2019 г.) / Пензенский гос. ун-т. – Пенза : Изд-во ПГУ, 2019. – С. 102–112.

12. Безяев, А. В. Оптимизация структуры самокорректирующегося био-кода, хранящего синдромы ошибок в виде фрагментов хеш-функций / А. В. Безяев, А. И. Иванов, Ю. В. Фунтикова // Вестник УрФО. Безопасность в информационной сфере. – 2014. – № 3 (13). – С. 4–13.

13. Биометрическая аутентификация по клавиатурному почерку с учетом силы нажатия на клавиши, параметров вибрации и движения рук оператора / А. Е. Сулавко, А. Р. Хамзин, А. А. Лыжин [и др.] // Вопросы защиты информации. – 2018. – № 2. – С. 41–50.

14. Биометрическая идентификация рукописных образов с использованием корреляционного аналога правила Байеса / А. И. Иванов, П. С. Ложников, Е. И. Качайкин, А. Е. Сулавко // Вопросы защиты информации. – 2015. – № 3. – С. 48–54.

15. Браницкий А.А., Дойникова Е.В., Котенко И.В. Использование нейросетей для прогнозирования подверженности пользователей социальных сетей деструктивным воздействиям // Информационно-управляющие системы. 2020. № 1 (104). С. 24-33.

16. Браницкий А.А., Шарма Яш.Д., Котенко И.В., Федорченко Е.В., Красов А.В., Ушаков И.А. Определение психического состояния пользователей

социальной сети reddit на основе методов машинного обучения // Информационно-управляющие системы. 2022. № 1 (116). С. 8-18.

17. Брюхомицкий Ю.А. Верификация динамических биометрических параметров личности на основе вероятностной нейронной сети // Известия ЮФУ. Технические науки. 2020. № 5 (215). С. 52-60.

18. Брюхомицкий Ю.А. Клавиатурный мониторинг на основе иммунологического клонирования // Безопасность информационных технологий. – 2016. - №4. – С. 5-11.

19. Брюхомицкий Ю.А. Модель искусственной иммунной системы с двойной пластичностью // Информационное противодействие угрозам терроризма. – 2013. - №20. - С. 76-83

20. Ю.А. Брюхомицкий. Регулирование распознающих свойств искусственных иммунных систем с двойной пластичностью. – 2013. - №20. - С. 83-87.

21. Брюхомицкий Ю.А., Абрамов Е.С. Верификация рукописных текстов с использованием иммунологических и нейросетевых технологий // Вопросы защиты информации. 2019. № 4 (127). С. 31-37.

22. Брюхомицкий Ю.А., Федоров В.М. Иммунологический метод текстонезависимой верификации личности по голосу // Известия ЮФУ. Технические науки. 2019. № 5 (207). С. 123-134.

23. Влияние психофизиологического состояния диктора на параметры его голоса и результаты биометрической аутентификации по речевому паролю / А. Е. Сулавко, А. В. Еременко, Р. В. Борисов, Д. П. Иниватов // Компьютерные инструменты в образовании. – 2017. – № 4. – С. 29–47.

24. Влияние психофизиологического состояния подписанта на результаты его идентификации по рукописному образу естественным и искусственным интеллектами / А. Е. Сулавко, С. С. Жумажанова, А. А. Нигрей, Л. Н. Закутнева // Безопасность информационных технологий. – 2017. – Т. 24, № 4. – С. 87–97.

25. Возможность идентификации пользователя по особенностям работы с мышью / А. Е. Сулавко, В. Ю. Писаренко, А. И. Голева [и др.] // Аппроксимация логических моделей, алгоритмов и задач - АЛМАЗ'2 : тез. докл. Междунар. конф.

(Омск, 27–30 апр. 2015 г.) / Ом. гос. техн. ун-т [и др.]. – Омск : Изд-во ОмГТУ, 2015. – С. 59–61.

26. Воронцов, К. В. Лекции по алгоритмическим композициям / К. В. Воронцов. – URL: http://www.machinelearning.ru/wiki/images/0/0d/Voron-ML-Compositions.pdf (дата обращения: 20.06.2022).

27. Второй национальный стандарт России по быстрому автоматическому обучению больших искусственных нейронных сетей на малых выборках биометрических данных / А. И. Иванов, А. В. Безяев, Е. А. Малыгина, Ю. И. Серикова // Безопасность информационных технологий : сб. науч. ст. по материалам I Всерос. науч.-техн. конф. (Пенза, 24 апр. 2019 г.) / Пензенский гос. ун-т. – Пенза : Изд-во ПГУ, 2019. – С. 174–177.

28. Вульфин А.М. Интеллектуальный анализ видеоданных в системе контроля соблюдения правил промышленной безопасности [Электронный ресурс] // Моделирование, оптимизация и информационные технологии. – 2020. – № 8(2). – С. 1–16. – Режим доступа: https://moit.vivt.ru/wpcontent /uploads /2020/05/Vulfin_2_20_1.pdf.

29. Вульфин А.М. Интеллектуальный анализ данных пользовательского окружения в задаче обнаружения удаленного управления [Электронный ресурс] // Моделирование, оптимизация и информационные технологии. – 2020. – № 8(2). – С. 1–19. – Режим доступа: https://moit.vivt.ru/wpcontent /uploads/2020/05/Vulfin_2_20_2.pdf

30. Вульфин, А. М. Обнаружение сетевых атак в гетерогенной промышленной сети на основе технологий машинного обучения / А. М. Вульфин // Программная инженерия. – 2022. – Т. 13, № 2. – С. 68-80. – DOI 10.17587/prin.13.68-80

31. Гайфулина Д.А., Котенко И.В. Применение методов глубокого обучения в задачах кибербезопасности. Часть 1 // Вопросы кибербезопасности. 2020. № 3 (37). С. 76-86.

32. Гайфулина Д.А., Котенко И.В. Применение методов глубокого обучения в задачах кибербезопасности. Часть 2 // Вопросы кибербезопасности. 2020. № 4 (38). С. 11-21.

33. Галушкин, А. И. Синтез многослойных систем распознавания образов / А. И. Галушкин. – Москва : Энергия, 1974. – 366 с.

34. Гарипов, И. М. Методы распознавания личности на основе анализа характеристик наружного уха (Обзор) / И. М. Гарипов, А. Е. Сулавко, И. А. Куприк // Вопросы защиты информации. – 2020. – № 1. – С. 33–41.

35. Генерация ключевых последовательностей и верификация субъектов на основе двумерного изображения лица / А. Е. Сулавко, А. В. Еременко, С. С. Жумажанова, Е. В. Бурая // Автоматизация процессов управления. – 2017. – № 1. – С. 58–66.

36. Генерация ключей шифрования на основе голосовых отпечатков человека / А. В. Еременко, А. Е. Сулавко, Р. В. Борисов, С. С. Мамонтов // Научно-технический прогресс: актуальные и перспективные направления будущего : сб. III Междунар. науч. конф., 10–11 авг. 2016 г. – Кемерово : Изд-во Зап.-Сиб. науч. центр, 2016. – С. 92–94.

37. Генерация криптографических ключей на основе голосовых отпечатков человека / Р. В. Борисов, А. Е. Сулавко, А. Е. Самотуга, А. В. Еременко // Труды научно-технической конференции кластера пензенских предприятий, обеспечивающих безопасность информационных технологий. – 2014. – Т. 9. – С. 79–82.

38. Гибридная интеллектуальная система обнаружения атак на основе комбинации методов машинного обучения / В. И. Васильев, А. М. Вульфин, В. Е. Гвоздев, Р. Р. Шамсутдинов // Моделирование, оптимизация и информационные технологии. – 2021. – Т. 9, № 3(34). – DOI 10.26102/2310-6018/2021.34.3.019

39. Горшков, Ю. Г. Обработка речевых и акустических биомедицинских сигналов на основе вейвлетов : моногр. / Ю. Г. Горшков. – Москва : Радиотехника, 2017. – 239 с. – SBN 978-5-93108-138-0.

40. ГОСТ Р 52633.0-2006. Защита информации. Техника защиты информации. Требования к средствам высоконадежной биометрической аутентификации : нац. стандарт : утв. и введ. в действие Приказом Федер. агентства по техн. регулированию и метрологии от 27 дек. 2006 г. № 372-ст : дата введ. 2007-04-01 /

разраб. Гос. науч.-исслед. испытательным институтом проблем техн. защиты информации Федер. службы по техн. и экспортному контролю (ГНИИИ ПТЗИ ФСТЭК России), Техническим комитетом по стандартизации ТК 362 «Защита информации». – Москва : Стандартинформ, 2007. – 25 с.

41. ГОСТ Р 52633.5-2011. Защита информации. Техника защиты информации. Автоматическое обучение нейросетевых преобразователей биометрия-код : нац. стандарт : утв. и введ. в действие Приказом Федер. агентства по техн. регулированию и метрологии от 1 дек. 2011 г. № 685-ст : дата введ. 2011-12-01 / разраб. Федер. гос. учреждением «Гос. науч.-исслед. испытательный ин-т проблем техн. защиты информации Федер. службы по техн. и экспортному контролю», Федер. гос. унитарным предприятием «Пензенский науч.-исслед. электротехн. Ин-т». – Москва : Стандартинформ, 2012. – 20 с.

42. Дагаева М.В., Катасёва Д.В., Катасёв А.С. Аугментация данных и построение нейросетевых моделей распознавания рукописных символов в системах биометрической аутентификации // Информация и безопасность. 2018. Т. 21. № 3. С. 366-371.

43. Дагаева М.В., Катасёва Д.В., Катасёв А.С. Обнаружение подмены пользователей в компьютерных системах на основе искусственной нейронной сети // Информация и безопасность. 2018. Т. 21, № 3. С. 296-301. заменить

44. Дагаева М.В., Катасёва Д.В., Катасёв А.С. Распознавание изображений человеческого лица на основе нейросетевой биометрической системы // Информация и безопасность. 2018. Т. 21. № 3. С. 394-399.

45. Дагаева М.В., Катасёва Д.В., Катасёв А.С., Кирпичников А.П. Нейросетевая модель динамической биометрии для обнаружения подмены пользователей в компьютерных системах // Вестник технологического университета. 2018. Т. 21, № 2. С. 115-119.заменить

46. Дагаева М.В., Сулейманов М.А., Катасёва Д.В., Катасёв А.С., Кирпичников А.П. Технология построения отказоустойчивых нейросетевых моделей распознавания рукописных символов в системах биометрической аутентификации // Вестник Технологического университета. 2018. Т. 21. № 2. С. 133-138.

47. Дасгупта, Д. Искусственные иммунные системы и их применение / под. ред. Д. Дасгупты ; пер с англ. А. А. Романюха. – Москва : Физматлит, 2006. – 344 с. – ISBN 5-9221-0706-2.

48. Де Прадо, Маркос Лопез. Машинное обучение: алгоритмы для бизнеса / Де Прадо Маркос Лопез. – Санкт Петербург : Питер, 2019. – 432 с. – ISBN: 978-5-4461-1154-1.

49. Диагностика состояния человека: математические подходы / А. В. Богомолов, Л. А. Гридин, Ю. А. Кукушкин, И. Б. Ушаков. – Москва : Медицина : Шико, 2003. – 461 с. – ISBN 5225041302.

50. Епифанцев, Б. Н. Алгоритм идентификации гипотез в пространстве малоинформативных признаков на основе последовательного применения формулы Байеса / Б. Н. Епифанцев, П. С. Ложников, А. Е. Сулавко // Межотраслевая информационная служба – 2013. – № 2. – С. 57–62.

51. Епифанцев, Б. Н. Альтернативные сценарии авторизации при идентификации пользователей по динамике подсознательных движений / Б. Н. Епифанцев, П. С. Ложников, А. Е. Сулавко // Вопросы защиты информации. – 2013. – № 2. – С. 28–35.

52. Епифанцев, Б. Н. Сравнение алгоритмов комплексирования признаков в задачах распознавания образов / Б. Н. Епифанцев, П. С. Ложников, А. Е. Сулавко // Вопросы защиты информации. – 2012. – № 1. – С. 60–66.

53. Еременко, А. В. Генерация ключевых последовательностей на основе параметров подсознательных движений / А. В. Еременко, П. С. Ложников, А. Е. Сулавко // Информационные системы и технологии. – 2017. – № 1. – С. 99–109.

54. Еременко, А. В. Двухфакторная аутентификация пользователей компьютерных систем на удаленном сервере по клавиатурному почерку / А. В. Еременко, А. Е. Сулавко // Прикладная информатика. – 2015. – № 6. – С. 48–59.

55. Еременко, А. В. Исследование алгоритма генерации криптографических ключей из биометрической информации пользователей компьютерных систем / А. В. Еременко, А. Е. Сулавко // Информационные технологии. – 2013. – № 11. – С. 47–51.

56. Еременко, А. В. Современное состояние и пути модернизации преобразователей биометрия-код / А. В. Еременко, А. Е. Сулавко, Д. А. Волков // Информационные технологии. – 2016. – № 3. – С. 203–210.

57. Еременко, Ю. И. Интеллектуальная система идентификации объектов с помощью алгоритмов иммунных систем / Ю. И. Еременко, И. В. Мельникова, А. А. Шаталов // Вестник Воронежского государственного технического университета. – 2015. – Т. 11, № 6. – С. 38–47.

58. Жиляков, Е. Г. Алгоритмы обнаружения основного тона речевых сигналов / Е. Г. Жиляков, А. А. Фирсова, Н. А. Чеканов // Научные ведомости БелГУ. Сер. История. Политология. Экономика. Информатика. – 2012. – № 1 (120), вып. 21. – С. 135–143.

59. Жиров, А. О. Безопасные облачные вычисления с помощью гомоморфной криптографии / А. О. Жиров, О. В. Жирова, С. Ф. Кренделев // Безопасность информационных технологий. – 2013. – Т. 20, № 1. – С. 6–12.

60. Иванов, А. И. Доверенный искусственный интеллект в защищенном исполнении для биометрии и иных важных приложений. Проблемы шифрования / А. И. Иванов // Системы безопасности. – 2020. – № 4. – URL: https://www.secuteck.ru/articles/doverennyj-iskusstvennyj-intellekt-v-zashchishchennom-ispolnenii-dlya-biometrii-i-inyh-vazhnyh-prilozhenij-problemy-shifrovaniya (дата обращения: 16.12.2021).

61. Иванов, А. И. Искусственный интеллект в защищенном исполнении: синтез статистико-нейросетевых автоматов многокритериальной проверки гипотезы независимости малых выборок биометрических данных : препринт / А. И. Иванов, Т. А. Золотарева ; Пензенский гос. ун-т. – Пенза : Изд-во ПГУ, 2020. – 104 с.

62. Иванов, А. И. Использование сетей корреляционных нейронов с многоуровневым квантованием: защита от извлечения знаний из параметров решающего правила : препринт / А. И. Иванов, А. Е. Сулавко ; Пензенский гос. ун-т. – Пенза : Изд-во ПГУ, 2020. – 48 с.

63. Иванов, А. И. НейроДинамика: гиперускорение направленных переборов или повышение достоверности статистических оценок на малых выборках :

препринт / А. И. Иванов ; Пензенский гос. ун-т. – Пенза : Изд-во ПГУ, 2021. – 106 с. – ISBN 978-5-907456-70-9.

64. Иванов, А. И. Нейросетевая защита конфиденциальных биометрических образов гражданина и его личных криптографических ключей : моногр. / А. И. Иванов ; Пензенский науч.-исслед. электротехн. ин-т. – Пенза : Изд-во ПНИЭИ, 2014. – 57 с.

65. Иванов, А. И. Нейросетевое обобщение семейства статистических критериев среднего геометрического и среднего гармонического для прецизионного анализа малых выборок биометрических данных / А. И. Иванов, К. А. Перфилов, В. С. Лукин // Информационно-управляющие телекоммуникационные системы, средства поражения и их техническое обеспечение : сб. науч. ст. Всерос. науч.-техн. конф. / под общ. ред. В. С. Безяева. – Пенза : Изд-во АО НПП «Рубин», 2019. – С. 50–63.

66. Иванов, А. И. Оценка надежности верификации автографа на основе искусственных нейронных сетей, сетей многомерных функционалов Байеса и сетей квадратичных форм / А. И. Иванов, П. С. Ложников, А. Е. Сулавко // Компьютерная оптика. – 2017. – Т. 41, № 5. – С. 765–774. – DOI: 10.18287/2412-6179-2017-41-5-765-774.

67. Иванов, А. И. Перспектива многократного увеличения ресурсов доверенных вычислений за счет привлечения гибрида нейросетевой обработки биометрии и гомоморфного шифрования / А. И. Иванов, В. С. Князьков // Состояние и перспективы развития современной науки по направлению «Техническое зрение, распознавание образов : материалы III Всерос. науч.-техн. конф. (Анапа, 18 марта 2021). – Анапа : Эра, 2021. – С. 173–176.

68. Иванов, А. И. Подсознание искусственного интеллекта: программирование автоматов нейросетевой биометрии языком их обучения / А. И. Иванов ; Пензенский науч.-исслед. электротехн. ин-т. – Пенза : Изд-во ПНИЭИ, 2012. – 125 с.

69. Иванов, А. И. Проект третьего национального стандарта России по быстрому автоматическому обучению больших сетей корреляционных нейронов

на малых обучающих выборках биометрических данных / А. И. Иванов, А. Е. Сулавко // Вопросы кибербезопасности. – 2021. – № 3. – С. 84–93. – DOI:10.21681/2311-3456-2021-3-84-93.

70. Иванов, А. И. Среда моделирования «БиоНейроАвтограф» : учеб.-метод. пособие / А. И. Иванов ; Пензенский гос. ун-т. – Пенза : Изд-во ПГУ, 2020. – 60 с.

71. Иванов, А. И. Таблица вероятности появления разных стартовых условий для атак Маршалко на нейроны с общими входными связями / А. И. Иванов, И. А. Крохин // Состояние и перспективы развития современной науки по направлению «Техническое зрение, распознавание образов : материалы III Всерос. науч.-техн. конф. (Анапа, 18 марта 2021 г.). – Анапа : ЭРА, 2021. – С. 171–172.

72. Идентификационный потенциал клавиатурного почерка с учетом параметров вибрации и силы нажатия на клавиши / А. В. Еременко, А. Е. Сулавко, Д. В. Мишин, А. А. Федотов // Прикладная информатика. – 2017. – Т. 12, № 1 (67). – С. 79–94.

73. Идентификационный потенциал пользователей компьютерных систем в процессе их профессиональной деятельности : моногр. / Б. Н. Епифанцев, А. Е. Сулавко, А. С. Ковальчук [и др.]. – Омск : Изд-во СибАДИ, 2017. – 1 CD-ROM. – ISBN 978-5-00113-046-8.

74. Идентификационный потенциал рукописных паролей в процессе их воспроизведения / Б. Н. Епифанцев, П. С. Ложников, А. Е. Сулавко, С. С. Жумажанова // Автометрия. – 2016. – № 3. – С. 28–36.

75. Идентификация личности по особенностям лица с использованием искусственной иммунной системы и формулы гипотез Байеса / А. Е. Сулавко, Е. В. Шалина // Интеллектуальный анализ сигналов, данных и знаний: методы и средства : сб. ст. II Всерос. науч.-практ. конф. с междунар. участием им. В. В. Губарева (Новосибирск, 11–13 дек. 2018 г.) / Новосиб. гос. техн. ун-т. – Новосибирск : Изд-во НГТУ, 2018. – С. 303–307.

76. Идентификация психофизиологических состояний подписантов по особенностям воспроизведения автографа / А. Е. Сулавко, А. В. Еременко, Е. А.

Левитская, А. Е. Самотуга // Информационно-измерительные и управляющие системы. – 2017. – № 1. – С. 40–48.

77. Идентификация функционального состояния водителей транспортных средств с учетом отклонений наблюдаемой вариабельности сердечного ритма / А. Е. Сулавко, А. С. Ковальчук, З. В. Семенова, С. С. Осипов // Труды научно-технической конференции кластера Пензенских предприятий, обеспечивающих безопасность информационных технологий. – Пенза : АО «ПНИЭИ», 2016. – Т. 10. – С. 60–62.

78. Идентификация человека с высокой точностью по особенностям работы головного мозга на основе визуальной стимуляции / А. Е. Сулавко, С. С. Жумажанова, Д. Г. Стадников [и др.] // Биомедицинская радиоэлектроника. – 2018. – № 12. – С. 12–25. – DOI: 10.18127/j15604136-201812-03.

79. Ильин, Е. П. Психофизиология состояний человека / Е. П. Ильин. – Москва : Питер, 2005. – 411 с. – ISBN 5-469-00446-5.

80. Иммунные алгоритмы распознавания образов и их применение в биометрических системах (Обзор) / А. Е. Сулавко, Е. В. Шалина, Д. Г. Стадников, А. Г. Чобан // Вопросы защиты информации. – 2019. – № 1. – С. 38–46.

81. Искусственный интеллект в защищенном исполнении на базе иммунных сетевых моделей распознавания образов на примере преобразователей биометрия-код / Е. В. Шалина, Н. В. Малинин, А. Е. Сулавко, Д. Г. Стадников // Вопросы защиты информации. – 2020. – № 2. – С. 31–40.

82. Исследование утечек информации ограниченного доступа в 2020 году // InfoWatch : сайт. – URL: https://www.infowatch.ru/analytics/analitika/issledovanie-utechek-informatsii-ogranichennogo-dostupa-v-2020-godu (дата обращения: 17.12.2021).

83. Катасёв А.С., Курбанов Б. Сверточная нейросетевая модель определения усталости человека по выражению лица // Вестник Технологического университета. 2023. Т. 26. № 3. С. 67-71.

84. Комплексирование независимых биометрических признаков при распознавании субъектов на основе сетей квадратичных форм, персептронов и

меры ХИ-модуль / А. Е. Сулавко, А. В. Еременко, Е. В. Толкачева, Р. В. Борисов // Информационно-управляющие системы. – 2017. – № 1 (86). – С. 50–62. – DOI: 10.15217/issn1684-8853.2017.1.50.

85. Комплексированная система идентификации личности по динамике подсознательных движений / Б. Н. Епифанцев, П. С. Ложников, А. Е. Сулавко, Р. В. Борисов // Безопасность информационных технологий. – 2011. – № 4. – С. 97–102.

86. Комплексная система распознавания водителей транспортных средств и их психофизиологического состояния по динамическим биометрическим признакам / А. Е. Сулавко, С. С. Жумажанова, З. В. Семенова [и др.] // Автоматизация. Современные технологии. – 2017. – № 8. – С. 373–380.

87. Компрометация тела, или как утекают биометрические данные // InfoWatch : сайт. – URL: https://www.infowatch.ru/analytics/daydzhesty-i-obzory/komprometatsiya-tela-ili-kak-utekayut-biometricheskie-dannye (дата обращения: 16.12.2021).

88. Костин, Д. В. Сравнительный анализ алгоритмов машинного обучения для проведения классификации сетевого зашифрованного трафика / Д. В. Костин, О. И. Шелухин // T-Comm: Телекоммуникации и транспорт. – 2016. – Т. 10, № 9. – С. 43–52.

89. Котенко, И. В. Модель компрометации объектов критической информационной инфраструктуры / И. В. Котенко, С. С. Хмыров // Актуальные проблемы инфотелекоммуникаций в науке и образовании (АПИНО 2022) : XI Международная научно-техническая и научно-методическая конференция, Санкт-Петербург, 15–16 февраля 2022 года. Том 1. – Санкт-Петербург: Санкт-Петербургский государственный университет телекоммуникаций им. проф. М.А. Бонч-Бруевича, 2022. – С. 614-619.

90. Котов, В. Д. Система обнаружения сетевых вторжений на основе механизмов иммунной модели / В. Д. Котов, В. И. Васильев // Известия ЮФУ. Технические науки. – 2011. – № 12. – URL: https://cyberleninka.ru/article/n/sistema-

obnaruzheniya-setevyh-vtorzheniy-na-osnove-mehanizmov-immunnoy-modeli (дата обращения: 26.07.2018).

91. Ложников, П. С. Биометрическая защита гибридного документооборота : моногр. / П. С. Ложников. – Новосибирск : Изд-во СО РАН, 2017. – 129 с. – ISBN 978-5-7692-1561-2.

92. Ложников, П. С. Методы распознавания человека по особенностям лица (Обзор) / П. С. Ложников, А. Е. Сулавко, С. С. Жумажанова // Вопросы защиты информации. – 2017. – № 4. – С. 32–43.

93. Ложников, П. С. Модель защиты гибридных документов на основе рукописных подписей их владельцев с учетом психофизиологического состояния подписантов / П. С. Ложников, А. Е. Сулавко, А. Е. Самотуга // Вопросы защиты информации. – 2016. – № 4. – С. 47–59.

94. Ложников, П. С. Технология идентификации пользователей компьютерных систем по динамике подсознательных движений / П. С. Ложников, А. Е. Сулавко // Автоматизация и современные технологии. Машиностроение. – 2015. – № 5. – С. 31–36.

95. Лукин, В. С. Сравнение мощности обычной и логарифмической форм статистических критериев среднего гармонического при использовании для проверки гипотезы нормального распределения данных малой выборки / В. С. Лукин // Известия высших учебных заведений. Поволжский регион. Технические науки. – 2020. – № 4. – С. 19–26.

96. Лукин, В. С. Формирование кода аутентификации из биометрических данных на основе автоматического обучения нового класса искусственных нейронов среднего гармонического / В. С. Лукин, А. И. Иванов // Теория и практика обеспечения информационной безопасности : сб. науч. тр. по материалам Всерос. науч.-теоретич. конф. (Москва, 3 дек. 2021 г). – Москва : Моск. техн. ун-т связи и информатики, 2021. – С. 270–276.

97. Малыгина, Е. А. Биометрико-нейросетевая аутентификация: перспективы применения сетей квадратичных нейронов с многоуровневым квантованием

биометрических данных : препринт / Е. А. Малыгина ; Пензенский гос. ун-т. – Пенза : Изд-во ПГУ, 2020. – 110 с.

98. Матвеев, Ю. Н. Технологии биометрической идентификации личности по голосу и другим модальностям // Вестник МГТУ им. Н.Э. Баумана. Сер. Приборострое-ние. – 2012. – № 3 (3). – С. 46–61.

99. Машин, В. А. Классификация функциональных состояний и диагностика психоэмоциональной устойчивости на основе факторной структуры показателей вариабельности сердечного ритма / В. А. Машин, М. Н. Машина // Российский физиологический журнал им. И. М. Сеченова. – 2004. – Т. 90, № 12. – С. 1508–1521.

100. Метод защиты текстовых документов на электронных и бумажных носителях на основе скрытого биометрического идентификатора субъекта, получаемого из подписи / А. В. Еременко, А. Е. Сулавко, Е. В. Толкачева, Е. А. Левитская // Информационные технологии. – 2016. – Т. 22, № 8. – С. 628–634.

101. Михерский, Р. М. Применение искусственной иммунной системы для распознавания зрительных образов / Р. М. Михерский // Компьютерная оптика. – 2018. – Т. 42, № 1. – С. 113–117.

102. Непрерывная идентификация субъектов на основе скрытого мониторинга периферийного оборудования компьютерных систем / Е. А. Левитская, П. С. Ложников, А. Е. Сулавко, А. В. Еременко // Труды научно-технической конференции кластера Пензенских предприятий, обеспечивающих безопасность информационных технологий. – Пенза : Изд-во Пензенского науч.-исслед. электротехн. ин-та, 2014. – Т. 9. – С. 76–78.

103. Нечеткий экстрактор для генерации ключей шифрования на основе параметров клавиатурного почерка / А. Е. Сулавко, А. В. Еременко, Е. В. Толкачева, С. С. Жумажанова // Информационные технологии и вычислительные системы. – 2016. – № 4. – С. 69–79.

104. Нигрей, А. А. Методы автоматической оценки психофизиологического состояния человека по параметрам электроэнцефалограмм (обзор) / А. А. Нигрей,

С. С. Жумажанова, А. Е. Сулавко // Биомедицинская радиоэлектроника. – 2020. – № 1. – С. 21–33. – DOI 10.18127/j15604136-202001-0.

105. Николенко, С. Глубокое обучение. Погружение в мир нейронных сетей / С. Николенко, А. Кадурин, Е. Архангельская. – Санкт-Петербург [и др.] : Питер, 2021. – 476 с. – ISBN 978-5-4461-1537-2.

106. О многообразии метрик, позволяющих наблюдать реальные статистики распределения биометрических данных «нечетких экстракторов» при их защите наложением гаммы / А. И. Иванов, С. А. Сомкин, Д. Ю. Андреев, Е. А. Малыгина // Вестник УрФО. Безопасность в информационной сфере. – 2014. – № 2 (12). – С. 16–23.

107. Об оценке возможностей человека по распознаванию рукописных образов в процессе их воспроизведения на экране монитора / В. И. Васильев, А. Е. Сулавко, С. С. Жумажанова, А. А. Нигрей // Омский научный вестник. – 2017. – № 5. – С. 175–180.

108. Обеспечение информационной безопасности АСУ ТП с использованием метода предиктивной защиты / С. В. Гарбук, Д. И. Правиков, А. В. Полянский, И. В. Самарин // Вопросы кибербезопасности. – 2019. – № 3 (31). – С. 63–71. – DOI: 10.21681/2311-3456-2019-3-63-71.

109. Обзор международного рынка биометрических технологий и их применение в финансовом секторе. Январь 2018 / Банк России. – URL: https://www.cbr.ru/content/document/file/36012/rev_bio.pdf (дата обращения: 16.12.2021).

110. Оценка идентификационного потенциала электроэнцефалограмм с использованием статистического подхода и сверточных нейронных сетей / А. Е. Сулавко, П. С. Ложников, А. Г. Чобан [и др.] // Информационно-управляющие системы. – 2020. – № 6. – С. 37–49. – DOI: 10.31799/1684-8853-2020-6-37-49

111. Оценка идентификационных возможностей биометрических признаков от стандартного периферийного оборудования / В. И. Васильев, П. С. Ложников, А. Е. Сулавко, С. С. Жумажанова // Вопросы защиты информации. – 2016. – № 1. – С. 12–20.

112. Оценка идентификационных возможностей особенностей работы пользователя с компьютерной мышью / Р. В. Борисов, Д. Н. Зверев, А. Е Сулавко, В. Ю. Писаренко // Вестник Сибирской государственной автомобильно-дорожной академии. – 2015. – № 5 (45). – С. 106–113.

113. Оценка информативности характеристик рукописных образов для идентификации психофизиологического состояния человека / А. Е. Сулавко, А. В. Еременко, Е. А. Левитская, А. Е. Самотуга, Е. В. Толкачева // Информационно-измерительные и управляющие системы. – 2017. – № 11. – С. 35–46.

114. Оценка ускорения вычислений от перехода к воспроизведению эффектов нейродинамики при анализе числа возможных состояний больших сетей искусственных нейронов / А. И. Иванов, А. И. Газин, А. Е. Сулавко, Д. Г. Стадников // Вопросы защиты информации. – 2020. – № 4. – С. 32–38.

115. Патент № 2543927 Российская Федерация, МПК G 06 K 9/00. Способ идентификации личности по особенностям динамики написания пароля : № 2014116281/08 : заявл. 22.04.2014 : опубл. 10.03.2015 / Б. Н. Епифанцев, П. С. Ложников, А. Е. Самотуга, А. Е. Сулавко.

116. Перспективы использования искусственных нейронных сетей с многоуровневыми квантователями в технологии биометрико-нейросетевой аутентификации / В. И. Волчихин, А. И. Иванов, В. А. Фунтиков, Е. А. Малыгина // Известия высших учебных заведений. Поволжский регион. – 2013. – № 4 (28). – С. 86–96.

117. Преобразователь образов голосовых паролей дикторов в криптографический ключ на основе комитета предварительно обученных сверточных нейронных сетей / А. Е. Сулавко, Д. П. Иниватов, Д. Г. Стадников [и др.] // Вопросы защиты информации. – 2021. – № 4. – С. 23–33.

118. Проверка гипотезы независимости малых выборок: воспроизведение эффектов нейродинамики через случайное прореживанием исходных данных / А. И. Иванов, Т. А. Золотарева, А. Е. Сулавко, А. Г. Чобан // Вопросы защиты информации. – 2020. – № 4. – С. 42–47.

119. Прозоровский, В. И. Вопросы организации экспертизы алкогольного опьянения / В. И. Прозоровский, И. С. Карандаев, А. Ф. Рубцов // Судебно-медицинская экспертиза. – 1967. – № 1. – С. 3–8.

120. Разграничение доступа к информации на основе скрытого мониторинга действий пользователей в информационных системах: непрерывная идентификация / А. В. Еременко, Е. А. Левитская, А. Е. Сулавко, А. Е. Самотуга // Вестник Сибирской государственной автомобильно-дорожной академии. – 2014. – № 6 (40). – С. 92–102.

121. Раскин, Д. Интерфейс: новые направления в проектировании компьютерных систем / Д. Раскин. – Санкт-Петербург : Символ-плюс, 2010. – 272 с.

122. Распознавание водителей и их функциональных состояний по обычному и тепловому изображениям лица / П. С. Ложников, А. Е. Сулавко, Е. В. Толкачева, С. С. Жумажанова // Труды научно-технической конференции кластера Пензенских предприятий, обеспечивающих безопасность информационных технологий. – Пенза : АО «ПНИЭИ», 2016. – Т. 10. – С. 63–65.

123. Распознавание личности и оценка ресурсного состояния человека на основе анализа электрической активности мозга / А. Е. Сулавко, Д. Б. Пономарев, А. А. Нигрей, Б. И. Хайдин // Нанотехнологии: разработка, применение - XXI век. – 2018. – № 4. – С. 31–43. – DOI: 10.18127/j22250980-201804-05.

124. Распознавание пользователей компьютерных систем по клавиатурному почерку с учетом регистрации дополнительных признаков при помощи специальных датчиков / А. В. Еременко, А. Е. Сулавко, Д. В. Мишин, А. А. Федотов // Датчики и системы. – 2017. – № 3. – С. 9–16.

125. Распознавание психофизиологических состояний пользователей на основе скрытого мониторинга действий в компьютерных системах / В. И. Васильев, А. Е. Сулавко, Р. В. Борисов, С. С. Жумажанова // Искусственный интеллект и принятие решений. – 2017. – № 3. – С. 21–37.

126. Самотуга, А. Е. Обнаружение подделок рукописных паролей в процессе их воспроизведения / А. Е. Самотуга, А. Е. Сулавко // Научно-технический прогресс: актуальные и перспективные направления будущего : сб. III Междунар. науч.

конф. (Кемерово, 10–11 авг. 2016 г.). – Кемерово : ООО «Западно-Сибирский научный центр», 2016. – Т. 1. – С. 53–56.

127. Саттон, Р. С. Обучение с подкреплением / Р. С. Саттон, Э. Дж. Барто ; пер. с англ. А. А. Слинкина. – Москва : ДМК Пресс, 2020. – 552. – ISBN 978-5-97060-097-9.

128. Свидетельство о государственной регистрации программы для ЭВМ № 2021660512 Российская Федерация. AIC desktop : № 2021617236 : заявл. 17.05.2021 : опубл. (зарег.) 28.06.2021 / А. Е. Сулавко, Д. Г. Стадников, А. Г. Чобан, Д. П. Иниватов; заявитель Ом. гос. техн. ун-т. – 1 с.

129. Свидетельство о государственной регистрации программы для ЭВМ № 2017616888 Российская Федерация. Среда для имитационного моделирования экспериментов и проверки гипотез по распознаванию образов «SHV-kernel» : № 2017614035 : заявл. 24.04.2017 : опубл. (зарег.) 19.06.2017 / А. Е. Сулавко ; заявитель Ом. гос. техн. ун-т. – 1 с.

130. Свидетельство о государственной регистрации программы для ЭВМ № 2019663412 Российская Федерация. Программный модуль для цифрового подписания PDF-документов «PdfDigiSign» : № 2019662174 : заявл. 07.10.2019 : опубл. (зарег.) 16.10.2019 / П. С. Ложников, М. А. Семиколенов, А. Е. Сулавко ; заявитель Ом. гос. техн. ун-т. – 1 с.

131. Свидетельство о государственной регистрации программы для ЭВМ № 2012612981 Российская Федерация. Распределенная система управления доступом к ресурсам компьютера на основе регистрируемых событий : опубл. (зарег.) 15.06.2012 / А. Е. Сулавко, А. Л. Богдан ; заявитель Ом. гос. техн. ун-т. – 1 с.

132. Свидетельство о государственной регистрации программы для ЭВМ № 2011619263 Российская Федерация. Мультифакторная система аутентификации «TEOFRAST-M» : № 2011619263 : заявл. 05.12.2011 : опубл. (зарег.) 01.02.2012 / П. С. Ложников, В. А. Перевальский, А. Е. Сулавко ; заявитель Ом. гос. техн. ун-т. – 1 с.

133. Свидетельство о государственной регистрации программы для ЭВМ № 2011611363 Российская Федерация. Система безопасности компьютера на основе

регистрируемых событий в компьютерных сетях : опубл. (зарег.) 28.04.2011/ А. Е. Сулавко, С. А. Голованов ; заявитель Ом. гос. техн. ун-т. – 1 с.

134. Свидетельство о государственной регистрации программы для ЭВМ № 2010610473 Российская Федерация. Программный модуль для обеспечения безопасности бухгалтерских информационных систем «TEOFRAST-B» : № 2009616252 : заявл. 10.11.2010 : опубл. (зарег.) 11.01.2010 / П. С. Ложников, А. В. Еременко, В. А. Перевальский, А. Е.Сулавко ; заявитель Ом. гос.техн.ун-т. – 1 с.

135. Свидетельство о регистрации электронного ресурса № 23578 от 26.04.2018. Разрез файлов формата WAV / Д. П. Иниватов, А. Е. Сулавко ; Ом. гос. техн. ун-т. – Москва : ОФЭРНиО. – 1 с.

136. Снижение требований к размеру тестовой выборки биометрических данных при переходе к использованию многомерных корреляционных функционалов Байеса / А. И. Иванов, П. С. Ложников, А. Е. Сулавко, Ю. И. Серикова // Инфокоммуникационные технологии. – 2017. – № 2. – С. 186–193.

137. Средняя стоимость ущерба от утечек данных. 14 декабря 2021 // InfoWatch : сайт. – URL: https://www.infowatch.ru/analytics/daydzhesty-i-obzory/srednyaya-stoimost-uscherba-ot-utechek-dannykh (дата обращения: 16.12.2021).

138. Структурный синтез многослойных нейронных сетей на основе энтропийного подхода / В. И. Васильев, А. М. Вульфин, И. Б. Герасимова, Л. Р. Черняховская // Vestnik UGATU. – 2019. – Т. 23, № 2(84)

139. Сулавко, А. Е. Абстрактная модель искусственной иммунной сети на основе комитета классификаторов и ее использование для распознавания образов клавиатурного почерка / А. Е. Сулавко // Компьютерная оптика. – 2020. – Т. 44, № 5. – С. 830–842. – DOI: 10.18287/2412-6179-CO-717.

140. Сулавко, А. Е. Архитектура перспективных нейронов для обработки биометрических данных с высокой взаимной корреляционной зависимостью / А. Е. Сулавко // Вопросы защиты информации. – 2018. – № 1. – С. 35–48.

141. Сулавко, А. Е. Биометрическая аутентификация на основе сети гиперболических нейронов Байеса с трехуровневыми квантователями / А. Е. Сулавко // Информационные технологии и автоматизация управления : материалы

XI Всерос. науч.-практ. конф. студентов, аспирантов, работников образования и промышленности (Омск, 29–30 мая 2020 г.) / Ом. гос. техн. ун-т. – Омск : Изд-во ОмГТУ, 2020. – С. 199–206. – 1 CD-ROM.

142. Сулавко, А. Е. Биометрическая аутентификация по клавиатурному почерку на основе иммунного алгоритма распознавания образов / А. Е. Сулавко, Е. В. Шалина, Д. Г. Стадников // Интеллектуальный анализ сигналов, данных и знаний: методы и средства : сб. ст. II Всерос. науч.-практ. конф. с междунар. участием им. В. В. Губарева (Новосибирск, 11–13 дек. 2018 г.) / Новосиб. гос. техн. ун-т. – Новосибирск : Изд-во НГТУ, 2018. – С. 307–315.

143. Сулавко, А. Е. Биометрическая аутентификация пользователей информационных систем по клавиатурному почерку на основе иммунных сетевых алгоритмов / А. Е. Сулавко, Е. В. Шалина // Прикладная информатика. – 2019. – № 3 (81). – С. 39–53. – DOI: 10.24411/1993-8314-2019-10014.

144. Сулавко, А. Е. Влияние психофизиологического состояния подписантов на биометрические параметры рукописных образов и результаты их верификации / А. Е. Сулавко, А. Е. Самотуга // Информационно-управляющие системы. – 2017. – № 6. – С. 29–42. – DOI: 10.15217/issn1684-8853.2017.6.29.

145. Сулавко, А. Е. Влияние функционального состояния оператора на параметры его клавиатурного почерка в системах биометрической аутентификации / А. Е. Сулавко // Датчики и системы. – 2017. – № 11. – С. 19–30.

146. Сулавко, А. Е. Высоконадежная аутентификация по рукописным паролям на основе гибридных нейронных сетей с обеспечением защиты биометрических эталонов от компрометации / А. Е. Сулавко // Информационно-управляющие системы. – 2020. – № 4. – С. 61–77. – DOI: 10.31799/1684-8853-2020-4-61-77.

147. Сулавко, А. Е. Высоконадежная двухфакторная биометрическая аутентификация по рукописным и голосовым паролям на основе гибких нейронных сетей / А. Е. Сулавко // Компьютерная оптика. – 2020. – Т. 44, № 1. – С. 82–91. – DOI: 10.18287/2412-6179-CO-567.

148. Сулавко, А. Е. Генерация криптографических ключей на основе голосовых сообщений / А. Е. Сулавко, А. В. Еременко, Р. В. Борисов // Прикладная информатика. – 2016. – № 5. – С. 76–89.

149. Сулавко, А. Е. Идентификация образов электроэнцефалограмм пользователей компьютерных систем при наборе парольных фраз на клавиатуре / А. Е Сулавко, С. С. Жумажанова, Д. Г. Стадников // Искусственный интеллект и принятие решений. – 2019. – № 2. – С. 15–27.

150. Сулавко, А. Е. Исключение искаженных биометрических данных из эталона субъекта в системах идентификации / А. Е. Сулавко, А. В. Еременко, А. Е. Самотуга // Информационные технологии и вычислительные системы. – 2013. – № 3. – С. 96–101.

151. Сулавко, А. Е. Искусственный интеллект в защищенном исполнении / А. Е. Сулавко // Информационная безопасность: современная теория и практика : сб. науч. тр. студентов, аспирантов и преподавателей по материалам III Межвуз. науч.-практ. конф. (Омск, 24 нояб. 2020 г.) / Сиб. гос. автомобил.-дорож. ун-т (СибАДИ). – Омск : Изд-во СибАДИ, 2020. – С. 112–114.

152. Сулавко, А. Е. Метод биометрической аутентификации на основе кепстральных характеристик эхограмм наружного уха и нейросетевого преобразователя биометрия-код / А. Е. Сулавко, А. А. Храмов // Прикладная информатика. – 2022. –Т. 17, № 1. – С. 69–82. – DOI: 10.37791/2687-0649-2022-17-1-69-82.

153. Сулавко, А. Е. Метод сжатия собственных областей классов образов в пространстве малоинформативных признаков / А. Е. Сулавко, А. В. Еременко // Искусственный интеллект и принятие решений. – 2014. – № 2. – С. 102–109.

154. Сулавко, А. Е. Модель защищенного нейро-иммунного контейнера для задач биометрической аутентификации / А. Е. Сулавко, А. А. Лыжин // Фундаментальные и прикладные исследования молодых ученых : сб. материалов IV Междунар. науч.-практ. конф. студентов, аспирантов и молодых ученых (Омск, 6–7 февр. 2020 г.). – Омск : Изд-во СибАДИ, 2020. – С. 378–382.

155. Сулавко, А. Е. Настройка и балансировка двухмерных гиперболических квантователей Байеса в бинарном исполнении, обеспечивающих равновероятные состояния разрядов выходного кода для образов «Чужой» / А. Е. Сулавко, А. И. Иванов // Безопасность информационных технологий : тр. II Всерос. науч.-техн. конф. / Пензенский гос. ун-т. – Пенза : Изд-во ПГУ, 2020. – С. 11–15.

156. Сулавко, А. Е. Непрерывная скрытая идентификация субъектов на основе стандартного периферийного оборудования / А. Е. Сулавко, А. В. Еременко // Аппроксимация логических моделей, алгоритмов и задач : материалы второй Междунар. конф. (Омск, 27–30 апр. 2015 г.) / Ом. гос. техн. ун-т. – Омск : Изд-во ОмГТУ, 2015. – С. 53–58.

157. Сулавко, А. Е. Разграничение доступа к информации на основе скрытого мониторинга действий пользователей в информационных системах: портрет нелояльного сотрудника / А. Е. Сулавко, А. В. Еременко, Е. А. Левитская // Известия Транссиба. – 2015. – № 1 (21). – С. 80–89.

158. Сулавко, А. Е. Разностные нейроны Байеса с множеством квантователей для высоконадежной аутентификации и защищенного исполнения искусственного интеллекта / А. Е. Сулавко // Безопасность информационных технологий : сб. науч. ст. по материалам II Всерос. науч.-техн. конф. / Пензенский гос. ун-т. – Пенза: Изд-во ПГУ, 2020. – С. 103–111.

159. Сулавко, А. Е. Тестирование нейронов для распознавания биометрических образов при различной информативности признаков / А. Е. Сулавко // Прикладная информатика. – 2018. – № 1. – С. 128–143.

160. Технологии скрытой биометрической идентификации пользователей компьютерных систем (Обзор) / В. И. Васильев, П. С. Ложников, А. Е. Сулавко А. В. Еременко // Вопросы защиты информации. – 2015. – № 3. – С. 37–47.

161. Технология использования больших нейронных сетей для преобразования нечетких биометрических данных в код ключа доступа : моногр. / Б. С. Ахметов, А. И. Иванов, В. А. Фунтиков, А. В. Безяев, Е. А. Малыгина. – Алматы : LEM, 2014. – 144 с. – ISBN 978-601-239-327-9.

162. ТС 26.2.002-2020. Системы обработки информации. Криптографическая обработка информации. Защита нейросетевых контейнеров с использованием криптографических алгоритмов. – Москва, 2020. – URL: https://tc26.ru/standarts/tekhnicheskie-spetsifikatsii/ts-26-2-002-2020-zashchita-neyrosetevykh-konteynerov-s-ispolzovaniem-kriptograficheskikh-algoritmov.html?ysclid=lg4xr79th4290187075 (дата обращения: 17.12.2021).

163. Утечки биометрических данных и агрессивное целевое вымогательство: «Лаборатория Касперского» спрогнозировала развитие сложных угроз в 2020 году // Kaspersky : сайт. – URL: https://www.kaspersky.ru/about/press-releases/2019_utechki-biometricheskih-dannyh-i-agressivnoe-celevoe-vymogatelstvo-laboratoriya-kasperskogo-sprognozirovala-razvitie-slozhnyh-ugroz-v-2020-godu (дата обращения: 16.12.2021).

164. Утечки данных в госсекторе стран мира // TAdviser. – URL: https://www.tadviser.ru/index.php/Статья:Утечки_данных_в_госсекторе_стран_мира (дата обращения: 17.06.2022).

165. Уэйнберг, Р. С. Основы психологии спорта и физической культуры / Р. С. Уэйнберг, Д. Гоулд. – Киев : Олимпийская литература, 1998. – 336 с.

166. Флах, П. Машинное обучение. Наука и искусство построения алгоритмов, которые извлекают знания из данных : пер. с англ. / П. Флах. – Москва : ДМК Пресс, 2015. – 399 с. – ISBN 978-5-97060-273-7.

167. Храмов, А. А. Спектральный анализ эхограмм ушного канала для биометрической идентификации / А. А. Храмов, А. Е. Сулавко // Образование. транспорт. инновации. Строительство : сб. материалов IV Нац. науч.-практ. конференции. – Омск : Изд-во СибАДИ, 2021. – С. 825–827.

168. Чернышев, Ю. О. Искусственные иммунные системы: обзор и современное состояние / Ю. О. Чернышев, Г. В. Григорьев, Н. Н. Венцов // Программные продукты и системы. – 2014. – № 4. – С. 136–142.

169. Чуйков, А. В. Нейросетевая система преобразования биометрических признаков пользователя в криптографический ключ / А. В. Чуйков, А. М. Вульфин, В. И. Васильев // Доклады Томского государственного университета

систем управления и радиоэлектроники. – 2018. – Т. 21, № 3. – С. 35-41. – DOI 10.21293/1818-0442-2018-21-3-35-41

170. Шелухин, О. И. Влияние структуры обучающей выборки на эффективность классификации приложений трафика методами машинного обучения / О. И Шелухин, А. Г. Симонян, А. В. Ванюшина // T-Comm: Телекоммуникации и транспорт. – 2017. – Т. 11, № 2. – С. 25–31.

171. Экспериментальная оценка надежности верификации подписи сетями квадратичных форм, нечеткими экстракторами и персептронами / П. С. Ложников, А. Е. Сулавко, А. В. Еременко, Д. А. Волков // Информационно-управляющие системы. – 2016. – № 5. – С. 73–85. – DOI: 10.15217/issn1684-8853.2016.5.73.

172. 3D ear normalization and recognition based on local surface variation / Y. Zhang, Z. Mu, L. Yuan [et al.] // Applied Sciences. – 2017. – Vol. 7 (1). – P. 104.

173. A Handwritten Character Recognition Algorithm based on Artificial Immune / Chen Yuefeng, Liang Chunli, Yang Donghong [et al.] // International Conference on Computer Application and System Modeling (ICCASM 2010) (Taiyuan, 22–24 October 2010). – IEEE, 2010. – DOI: 10.1109/ICCASM.2010.5622270.

174. A Perspective Analysis of Handwritten Signature Technology / Moises Diaz, Miguel A. Ferrer, Donato Impedovo [et al.] // ACM Computing Surveys. – 2019. – Vol. 51 (6). – Article 117.

175. A secure face-verification scheme based on homomorphic encryption and deep neural networks / Y. Ma, L. Wu, X. Gu [et al.] // IEEE Access. – 2017. – Vol. 5. – P. 16532–16538.

176. A Survey of Federated Learning for Edge Computing: Research Problems and Solutions / Qi Xia, Winson Ye, Zeyi Tao [et al.] // High-Confidence Computing. – 2021. – Vol. 1, no, 1. – DOI: 10.1016/j.hcc.2021.

177. A survey on ensemble learning for data stream classification / H. M. Gomes, J. P. Barddal, F. Enembreck, A. Bifet // ACM Computing Surveys. – 2017. – Vol. 50, no 2. – P. 1–36. – DOI: 10.1145/3054925.

178. Advanced Materials homepage. – URL http://aiconstructor.ru/page14247028.html (date accessed: 01.11.2020).

179. Advances and Open Problems in Federated Learning / P. Kairouz, H. B. McMahan, B. Avent [et al.] // Foundations and Trends in Machine Learning. – 2021. – Vol. 14. – DOI: 10.1561/2200000083.

180. AIConstructor - облачная среда разработки искусственного интеллекта для цифровой трансформации предприятий без написания кода / А. Е. Сулавко, Д. П. Иниватов, А. В. Еременко, Е. В. Шалина // Информационная безопасность: современная теория и практика : сб. науч. тр. студентов, аспирантов и преподавателей по материалам III Межвуз. науч.-практ. конф. (Омск, 24 нояб. 2020 г.) / Сиб. гос. автомобил.-дорож. ун-т (СибАДИ). – Омск : Изд-во СибАДИ, 2020. – С. 115–120.

181. Akhmetov, B. S. Training of neural network biometry-code converters / B. S. Akhmetov, A. L. Ivanov, Z. K. Alimseitova // News of the National Academy of Sciences of the Republic of Kazakhstan. Series of Geology and Technical Sciences. – 2018. – Vol. 1. – P. 61–68.

182. Akkermans, T. H. Acoustic ear recognition / T. H. Akkermans, T. A. Kevenaar, D. W. Schobben // International Conference on Biometrics. – Berlin, Heidelberg : Springer. – P. 697–705.

183. Antal, M. Keystroke Dynamics on Android Platform / Margit Antal, László Zsolt Szabó, Izabella Laszlo // Procedia Technology. – 2015. – Vol. 19. – P. 820–826. – DOI: 10.1016/j.protcy.2015.02.118.

184. Applicability of classical and hybrid neural network algorithms in problems of recognition of biometric patterns / V. I. Vasilyev, A. E Sulavko, G. A. Fofanov, D. P. Inivatov // XIV International Scientific-Technical Conference on Actual Problems of Electronics Instrument Engineering (APEIE) (Novosibirsk, 2–6 October 2018). – IEEE, 2018. – P. 563–568. – DOI: 10.1109/APEIE.2018.8545122.

185. Application of artificial neural networks forhandwritten biometric images recognition / A. Malygin, N. Seilova, K. Boskebeev, Zh. Alimseitova // Computer Modelling and New Technologies. – 2017. – Vol. 21 (1). – P. 31–38.

186. Application of the Clonal Selection Algorithm in artificial immune systems for shape recognition / N. Isa, N. M. Sabri, K. S. Jazahanim, N. K. Taylor // International Conference on Information Retrieval & Knowledge Management. – Malysia, 2010. – P. 223–228.

187. Arbab-Zavar, B. On model-based analysis of ear biometrics / B. Arbab-Zavar, M. S. Nixon, D. J. Hurley // First IEEE International Conference on Biometrics: Theory, Applications, and Systems. – IEEE, 2007. – P. 1–5.

188. Artificial immune system based neural networks for solving multi-objective programming problems / W. F. Abd El-Wahed, E. M. Zaki, A. M. El-Refaey // Egyptian Informatics Journal. – 2010. – Vol. 11 (2). – P. 59–65.

189. Asad, Muhammad. FedOpt: Towards Communication Efficiency and Privacy Preservation in Federated Learning / Muhammad Asad, Ahmed Moustafa, Takayuki Ito // Applied Sciences. – 2020. – Vol. 10 (8). – P. 2864. – URL: https://doi.org/10.3390/app10082864 (date accessed: 07.04.2021).

190. Assessing the Level of Uncertainty of small samples of Multidimensional Biological and Biometric Data / B. Akhmetov, A. Ivanov, E. Malygina [et al.] // International journal of engineering sciences & research technology. – 2014. – Vol. 3 (7). – P. 284–288.

191. Baker, B. Designing neural network architectures using reinforcement learning / B. Baker, O. Gupta // 5th International Conference on Learning Representations. – Toulon, Franc. – 2017. – P. 1–18. – URL: https://arxiv.org/pdf/1611.02167.pdf (date accessed: 07.04.2021).

192. Bersini, H. The Immune Learning Mechanisms: Recruitment, Reinforcement and their Applications / H. Bersini, F. Varela // Computing with Biological Metaphors. – 1994. – URL: https://www.bibsonomy.org/publication/fb4adc4eeadcf020c6f0e7c749ec8600/n.nanas (дата обращения: 20.06.2022).

193. Biometric authentication on the basis of lectroencephalograms parameters / A. E. Sulavko, A. E. Samotuga, D. G. Stadnikov, V. A. Pasenchuk, S. S. Zhumazhanova //

Journal of Physics: Conference Series. – 2019. – Vol. 1260 (2). – P. 022011. – DOI:10.1088/1742-6596/1260/2/022011.

194. Biometric Technology in Securing the Internet Using Large Neural Network Technology / B. Akhmetov, A. Doszhanova, A. Ivanov [et al.] // International Journal of Computer and Information Engineering. – 2013. – Vol. 7 (7). – P. 129–139.

195. Boddeti, V. Naresh. Secure Face Matching Using Fully Homomorphic Encryption / V. Naresh Boddeti // 9th International Conference on Biometrics Theory, Applications and Systems (BTAS). – IEEE, 2018. – P. 1–10. – DOI: 10.1109/BTAS.2018.8698601.

196. Bogdanov, D. S. Data recovery for a neural network-based biometric authentication scheme / D. S. Bogdanov, V. O. Mironkin // Математические вопросы криптографии. – 2019. – Vol. 10, № 2. – С. 61–74.

197. Boneh, Dan. Evaluating 2-DNF formulas on ciphertexts / Dan Boneh, Eu-Jin Goh, Kobbi Nissim // TCC'05. Theory of Cryptography Conference. – 2005. – P. 325–341. – URL: https://doi.org/10.1007/978-3-540-30576-7_18 (date accessed: 07.04.2021).

198. Boosting the Margin: A New Explanation for the Effectiveness of Voting Methods / Robert E. Schapire, Yoav Freund, Peter Bartlett, Wee Sun Lee // The Annals of Statistics. – 1998. – Vol. 26 (5). – P. 1651–1686.

199. Brzezinski, D. Reacting to Different Types of Concept Drift:The Accuracy Updated Ensemble Algorithm / D. Brzezinski, J. Stefanowski // IEEE Transactions On Neural Networks And Learning Systems. – 2014. – Vol. 25. – P. 81–94. – DOI: 10.1109/TNNLS.2013.2251352.

200. Catak, F. O. A Privacy-Preserving Fully Homomorphic Encryption and Parallel Computation Based Biometric Data Matching : preprints / F. O. Catak, S. Yildirim Yayilgan, M. Abomhara, 2020. – № 2020070658. – DOI: 10.20944/preprints202007.0658.v1.

201. Chen, H. Human ear recognition in 3D / H. Chen, B. Bhanu // IEEE Transactions on Pattern Analysis and Machine Intelligence. – 2007. – Vol. 29 (4). – P. 718–737.

202. Chmielewski, A. An Immune Approach to Recognition of Handwritten Words / Andrzej Chmielewski, Slawomir T. Wierzchon // International Conference on Biometrics and Kansei Engineering (Cieszyn, 25–28 June 2009). – IEEE, 2009. – P. 49–54.

203. Choraś, M. Perspective methods of human identification: ear biometrics / M. Choraś // Opto-electronics review. – 2008. – Vol. 16 (1). – P. 85–96.

204. Chowdhury, D. Modeling immune network through cellular automata: a unified mechanism of immunological memory / D. Chowdhury, V. Deshpande, D. Stauffer // International Journal of Modern Physics. – 1994. – Vol. 5 (6). – P. 1049–1072.

205. Common genetic encoding for both direct and indirect encodings of networks / Yohannes Kassahun, Gerald Sommer, Mark Edgington [et al.] // Genetic and Evolutionary Computation Conference. – ACM Press, 2007. – P. 1029–1036.

206. Convolutional encoder–decoder networks for pixel-wise ear detection and segmentation / Ž. Emeršič, L. L. Gabriel, V. Štruc, P. Peer // IET Biometrics. – 2018. – Vol. 7 (3). – P. 175–184.

207. Corus, D. Fast Artificial Immune Systems / D. Corus, P. S. Oliveto, D. Yazdani // International Conference on Parallel Problem Solving from Nature. – 2018. – № 2. – P. 67–78. – DOI: 10.1007/978-3-319-99259-4_6.

208. Craig, Gentry. A fully homomorphic encryption scheme / Gentry Craig. – Stanford University, 2009. – 199 p.

209. Cryptographic key generation from voice / F. Monrose, M. K. Reiter, Q. Li, S. Wetzel // IEEE Symposium on Security and Privacy (Oakland, CA, USA, 14–16 May 2000. – IEEE, 2001. – DOI: 10.1109/SECPRI.2001.924299.

210. Dasgupta, D. MILA-multilevel immune learning algorithm / D. Dasgupta, S. Yu, N. S. Majumdar // Genetic and Evolutionary Computation Conference. –Berlin, Heidelberg : Springer, 2003. – P. 183–194.

211. De Castro L. N. aiNET: An artificial immune network for data analysis / L. N. de Castro, F. J. von Zuben // Data mining: a heuristic approach. – IGI Global, 2002. – P. 231–260.

212. De Castro, L. N. The clonal selection algorithm with engineering applications / L. N. de Castro, F. J. Von Zuben // Workshop on Artificial Immune Systems and Their Applications. – Las Vegas, USA, 2000. – P. 36–37.

213. De Waard, D. The measurement of drivers' mental workload / Dick de Waard ; Traffic Safety Research Centre VSC. – Haren : University of Groningen, 1996. – 135 p. – ISBN 90-6807-308-7.

214. Dechter, R. Learning While Searching in Constraint-Satisfaction-Problems / R. Dechter // Conference: Proceedings of the 5th National Conference on Artificial Intelligence. – Philadelphia, 1986. – Vol. 1: Science. – P. 178–183.

215. Deep hashing for compact binary codes learning / V. Erin Liong, J. Lu, G. Wang, P. Moulin, J. Zhou // IEEE Conference on Computer Vision and Pattern Recognition. – Boston, MA, 2015. – P. 2475–2483.

216. Deep secure encoding for face template protection / R. K. Pandey, Y. Zhou, B. U. Kota, V. Govindaraju // IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW). – 2016. – P. 9–15. – DOI: 10.1109/CVPRW.2016.17.

217. Deniz, E. Clonal selection algorithm application to simple microwave matching network / E. Deniz, S. Ülker // Microwave and Optical Technology Letters. – 2011. – Vol. 53 (5). – P. 991–993.

218. Differential privacy-enabled federated learning for sensitive health data / O. Choudhury, A. Gkoulalas-Divanis, T. Salonidis [et al.]. – URL: https://arxiv.org/abs/1910.02578 (date accessed: 07.04.2021).

219. Djeddi, C. Artificial Immune Recognition System for Arabic writer identification / C. Djeddi, L. Souici-Meslati // International symposium on innovations in information and communications technology (Amman, Jordan, 29 November 2011). – IEEE, 2011. – P. 159–165.

220. Dodis, Y. Fuzzy Extractors: How to Generate Strong Keys from Biometrics and Other Noisy Data / Y. Dodis, L. Reyzin, A. Smith // International Conference on the Theory and Applications of Cryptographic Techniques. – Springer, Berlin, Heidelberg, 2004. – P. 523–540.

221. D-vector based speaker verification system using Raw Waveform CNN / Jung Jee-weon, Hee-Soo Heo, Il-Ho Yang [et al.]. – URL: https://www.researchgate.net/publication/322912387_D-vector_based_speaker_verification_system_using_Raw_Waveform_CNN (date accessed: 01.02.2022).

222. EarEcho: Using Ear Canal Echo for Wearable Authentication / Y. Gao, W. Wang, V. V. Phoha [et al.] // Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies. – 2019. – Vol. 3 (3). – P. 81:1–81:24.

223. Efficient detection and recognition of 3D ears / S. M. Islam, R. Davies, M. Bennamoun, A. S. Mian // International Journal of Computer Vision. – 2011. – Vol. 95 (1). – P. 52–73.

224. Elrefaei, L. A. Machine vision gait-based biometric cryptosystem using a fuzzy commitment scheme / L. A. Elrefaei, Al-A. M. Mohammadi. – https://doi.org/10.1016/j.jksuci.2019.10.011 // Journal of King Saud University-Computer and Information Sciences. – 2019. – URL: https://www.sciencedirect.com/science/article/pii/S1319157819300916 (date accessed: 07.04.2021).

225. Emeršič, Ž. Ear recognition: More than a survey / Ž. Emeršič, V. Štruc, P. Peer // Neurocomputing. – 2017. – Vol. 255. – P. 26–39.

226. Experimental Studies of Network Traffic of Mobile Devices with Android OS / O. I. Sheluhin, S. D. Erokhin, A. V Osin, V. V. Barkov // Systems of Signals Generating and Processing in the Field of on Board Communications. – IEEE, 2019. – P. 1–4. – DOI: 10.1109/SOSG.2019.8706824.

227. Faraoun, K. M. Artificial Immune Systems for text-dependent speaker recognition / K. M. Faraoun, A. Boukelif // Journal of Computer Science. – 2006. – Vol. 5 (4). – P. 19–26.

228. Faster R-CNN: Towards real-time object detection with region proposal networks / S. Ren, K. He, R. Girshick, J. Sun // Advances in neural information processing systems. – 2015. – P. 91–99.

229. Flexible fast learning neural networks and their application for building highly reliable biometric cryptosystems based on dynamic features / V. I. Vasilyev, P. S. Lozhnikov, A. E. Sulavko [et al] // IFAC-PapersOnLine. – 2018. – Vol. 51 (30). – P. 527–532. – DOI: 10.1016/j.ifacol.2018.11.272.

230. Fung, Clement. The Limitations of Federated Learning in Sybil Settings / Clement Fung, Chris J. M. Yoon, Ivan Beschastnikh // 23rd International Symposium on Research in Attacks, Intrusions and Defenses (RAID-2020). – 2020. – P. 301–316. – URL: https://www.usenix.org/system/files/raid20-fung.pdf (date accessed: 07.04.2021).

231. Fuzzy commitment scheme for generation of cryptographic keys based on iris biometrics / S. Adamovic, M. Milosavljevic, M. Veinovic [et al.] // IET Biometrics. – 2017. – Vol. 6, no. 2. – P. 89–96. – DOI: 10.1049/iet-bmt.2016.0061.

232. Fuzzy vault scheme based on xed-length templates applied to dynamic signature verification / W. Ponce-Hernandez, R. Blanco-Gonzalo, J. Liu-Jimenez, R. Sanchez-Reillo // IEEE Access. – 2020. – Vol. 8. – P. 11152–11164.

233. Garain, U. Recognition of handwritten Indic script digits using clonal selection algorithm / U. Garain, M. P. Chakraborty, D. Dasgupta // International Conference on Artificial Immune Systems. – 2006. – Vol. 4163. – P. 256–266.

234. Greensmith, J. Detecting danger: The dendritic cell algorithm / J. Greensmith, U. Aickelin, S. Cayzer // Robust Intelligent Systems. – London : Springer, 2008. – P. 89–112.

235. Greensmith, J. The Deterministic Dendritic Cell Algorithm / J. Greensmith, U. Aickelin // International conference on artificial immune systems. – Berlin, Heidelberg : Springer, 2008. – С. 291–302.

236. Hafemann, L. G. Characterizing and evaluating adversarial examples for offline handwritten signature verification / L. G. Hafemann, R. Sabourin, L. S. Oliveira // IEEE Transactions on Information Forensics and Security. – 2019. – Vol. 14 (8). – P. 2153–2166.

237. Hafemann, Luiz G. Writer-independent Feature Learning for Offline Signature Verification using Deep Convolutional Neural Networks / Luiz G. Hafemann, Robert

Sabourin, Luiz S. Oliveira // International Joint Conference on Neural Networks (IJCNN). – IEEE, 2016. – DOI: 10.1109/IJCNN.2016.7727521.

238. Hao, F. Crypto with Biometrics Effectively / F. Hao, R. Anderson, J. Daugman // IEEE Transactions on Computers. – 2006. – Vol. 55 (9). – P. 1081–1088.

239. Hellström, E. Feature learning with deep neural networks for keystroke biometrics: A study of supervised pre-training and autoencoders. Computer Science and Engineering, master's level / E. Hellström. – Lulea: Lulea University of Technology, 2018. – 75 p.

240. Hine, G. E. A zero-leakage fuzzy embedder from the theoretical formulation to real data / G. E. Hine, E. Maiorana, P. Campisi // IEEE Transactions on Information Forensics and Security. – 2017. – Vol. 12 (7). – P. 1724–1734.

241. Hinton, G. E. Training products of experts by minimizing contrastive divergence / G. E. Hinton // Neural computation. – 2002. – Vol. 14, №. 8. – P. 1771–1800.

242. Homomorphic Encryption for Speaker Recognition / A. Nautsch, S. Isadskiy, J. Kolberg [et al.] // Protection of Biometric Templates and Vendor Model Parameters. Proc. Odyssey 2018. The Speaker and Language Recognition Workshop. – URL: https://arxiv.org/pdf/1803.03559.pdf (date accessed: 07.04.2021).

243. Hua, Quan Z. Online signature verification based on the hybrid HMM/ANN Model / Quan Z. Hua, Liu K. Hong // International Journal of Computer Science and Network Security. – 2007. – Vol. 7 (3). – P. 313–320.

244. Hurley, D. J. Force field feature extraction for ear biometrics / D. J. Hurley, M. S. Nixon, J. N.Carter // Computer Vision and Image Understanding. – 2005. – Vol. 98 (3). – P. 491–512.

245. Iannarelli, A. V. Forensic identification series: ear identification / A. V. Iannarelli. – California : Paramont Publishing Company, 1989. – 213 p.

246. Identification of the Psychophysiological State of the User Based on Hidden Monitoring in Computer Systems / V. I. Vasilyev, A. E. Sulavko, S. S. Zhumazhanova, R. V. Borisov // Scientific and Technical Information Processing. – 2018. – Vol. 45 (6). – P. 398–410. – DOI: 10.3103/S0147688218060096.

247. Identification potential capacity of typical hardware for the purpose of hidden recognition of computer network users / V. I. Vasilyev, A. E. Sulavko, A. V. Eremenko, S. S. Zhumazhanova // Dynamics of Systems, Mechanisms and Machines: conference proceeding (Omsk, 15–17 November 2016) / Omsk State Technical University. – IEEE, 2016. – P. 1–5/ – DOI: 10.1109/Dynamics.2016.7819106.

248. Identification Potential of Online Handwritten Signature Verification / B. N. Epifantsev, P. S. Lozhnikov, A. E. Sulavko, S. S. Zhumazhanova // Optoelectronics, Instrumentation and Data Processing. – 2016. – Vol. 3 (52). – P. 238–244. – DOI: 10.3103/S8756699016030043.

249. Ignatenko, T. Willems. Information Leakage in Fuzzy Commitment Schemes / T. Ignatenko, M. J. Frans // IEEE Transactions on Information Forensics and Security. – 2010. – Vol. 5 (2). – P. 337–348. – DOI: 10.1109/TIFS.2010.2046984.

250. Iranmanesh, V. Online Signature Template Protection by Shuffling and One time Pad Schemes with Neural Network Verification / V. Iranmanesh // Proceedings of the International Conference on Computer Science and Computational Mathematics (ICCSCM '13). – IEEE, 2013. – P. 53–59.

251. Islam, S. M. Fast and fully automatic ear detection using cascaded adaboost / S. M. Islam, M. Bennamoun, R. Davies // Workshop on Applications of Computer Vision. – IEEE, 2008. – P. 1–6.

252. ISO/IEC 18033-6:2019. IT Security techniques - Encryption algorithms. Part 6. Homomorphic encryption. – URL: https://www.iso.org/obp/ui/#iso:std:iso-iec:18033:-6:ed-1:v1:en (date accessed: 07.04.2021).

253. Ivanov, A. I. A Complete Statistical Model of a Handwritten Signature as an Object of Biometric Identification / A. I. Ivanov, E. I. Kachajkin, P. S. Lozhnikov. // International Siberian Conference on Control and Communications (SIBCON) (Moscow, 12–14 May 2016). – IEEE, 2016 – DOI: 10.1109/SIBCON.2016.7491678.

254. Ivanov, A. I. Reducing the Size of a Sample Sufficient for Learning Due to the Symmetrization of Correlation Relationships Between Biometric Data / A. I. Ivanov, P. S. Lozhnikov, Y. I. Serikova // Cybernetics and Systems Analysi. – 2016. – Vol. 52 (3). – P. 379–385. – DOI: 10.1007/s10559-016-9838-x.

255. I-Vector/HMM Based Text-Dependent Speaker Verification System for RedDots Challenge / H. Zeinali, H. Sameti, L. Burget [et al.]. – URL: https://www.researchgate.net/publication/303895014_i-VectorHMM_Based_Text-Dependent_Speaker_Verification_System_for_RedDots_Challenge (date accessed: 01.02.2022).

256. Jain, A. K. Biometric Template Security / A. K. Jain, K. Nandakumar, A. Nagar // EURASIP Jornal on Advances in Signal Processing. – 2008. – Vol. 1. – P. 113-1–113-17. – DOI: 10.1155/2008/579416.

257. Jeges, E. Model-based human ear identification / E. Jeges, L. Máté // World Automation Congress. – IEEE, 2006. – P. 1–6.

258. Jeong, J. Moir. Kepstrum approach to real-time speech-enhancement methods using two microphones / J. Jeong, T. J. Moir // Res. Lett. Inf. Math. Sci. – 2005. – Vol. 7. – P. 135–145.

259. Juniper Research: The Future of Cybercrime & Security. – DOI: 10.1016/S1361-3723(18)30082-4 // Computer Fraud & Security. – 2018. – Vol. 2018 (9). – P. 4.

260. Kadwe, Y. A Review on Concept Drift / Y. Kadwe, V. Suryawanshi // IOSR Journal of Computer Engineering (IOSRJCE). – 2015. – Vol. 17, no. 1. – P. 20–26. – DOI: 10.9790/0661-17122026.

261. Kenny, P. A small footprint i-vector extractor / P. Kenny. – URL: https://studylib.net/doc/18199504/a-small-footprint-i-vector-extractor?ysclid=lg6k90dgpu470258079 (date accessed: 01.02.2022).

262. Kevin, S. Killourhy. Comparing Anomaly Detectors algorithms for Keystroke Dynamics / Kevin S. Killourhy, Roy A. Maxion // Proceedings of the 39th Annual International Conference on Dependable Systems and Networks (DSN-2009) (Lisbon, Portugal, 29 June – 2 July 2009). – IEEE, 2009. – P. 125–134. – DOI: 10.1109/DSN.2009.5270346.

263. Kholmatov, A. Identity authentication using improved online signature verification method // Pattern Recognition Letters. – 2005. – Vol. 26 (15). – P. 2400–2408.

264. Knight, T. AINE: an immunological approach to data mining / T. Knight, J. Timmis // Proceedings International Conference on Data Mining. – IEEE Computer Society, 2001. – P. 297–304.

265. Kobojek, P. Application of Recurrent Neural Networks for User Verification based on Keystroke Dynamics / P. Kobojek, K. Saeed // Journal of Telecommunications and Information Technology. – 2016. – Vol. 3. – P. 80–90.

266. Kotenko, I., Avramenko, V., Malikov, A., Saenko, I. An Approach to the Synthesis of a Neural Network System for Diagnosing Computer Incidents // Studies in Computational Intelligence, 2022, 1026, стр. 407–416

267. Kotenko, I., Saenko, I., Lauta, O., Vasiliev, N., Kribel, K. Attacks Against Artificial Intelligence Systems: Classification, The Threat Model and the Approach to Protection // Lecture Notes in Networks and Systems, 2023, 566 LNNS, pp. 293–302

268. Koutn´ık, J. Evolving deep unsupervised convolutional networks for vision-based reinforcement learning / J. Koutn´ık, J. Schmidhuber, F. Gomez // Proceedings of the 2014 Annual Conference on Genetic and Evolutionary Computation (GECCO 2014). – New York, 2014. – P. 541–548. – DOI: 10.1145/2576768.2598358.

269. Kumar Jindal, A. Face template protection using deep convolutional neural network / A. Kumar Jindal, S. Chalamala, S. Kumar Jami // IEEE Conference on Computer Vision and Pattern Recognition Workshops. – 2018. – P. 462–470. – URL: https://openaccess.thecvf.com/content_cvpr_2018_workshops/papers/w11/Jindal_Face_Template_Protection_CVPR_2018_paper.pdf (date accessed: 07.04.2021).

270. Kumar, A. Ear authentication using Log-Gabor wavelets / A. Kumar, D. Zhang // Biometric Technology for Human Identification IV. – 2007. – Vol. 6539. – P. 65390A. – DOI: 10.1117/12.720244.

271. Kumar, Akshat. An Artificial Immune System based Approach for English Grammar Checking / Akshat Kumar, Shivashankar B. Nair // International Conference on Artificial Immune Systems. – Berlin, Heidelberg : Springer, 2007. – Vol. 4628. – P. 348–357.

272. Kurkova, V. Model complexities of shallow networks representing highly varying functions / V. Kurkova, M. Sanguineti // Neurocomputing. – 2016. – Vol. 171. – P. 598–604.

273. Kurkova, V. Probabilistic lower bounds for approximation by shallow perceptron networks / V. Kurkova, M. Sanguineti // Neural Networks. – 2017. – Vol. 91. – P. 34–41.

274. Learning pairwise SVM on hierarchical deep features for ear recognition / I. Omara, X. Wu, H. Zhang [et al.] // IET Biometrics. – 2018. – Vol. 7 (6). – P. 557–566.

275. LeCun, Y. Deep Learning / Y. LeCun, Y. Bengio, G. Hinton // Nature. – 2015. – Vol. 521. – P. 436–444. – DOI: 10.1038/nature14539.

276. López, G. Q. Chapter 23. Immunological computation / G. Q. López, L. A. Morales, L. F. Niño // Autoimmunity: From Bench to Bedside [Internet]. – URL: https://www.ncbi.nlm.nih.gov/books/NBK459484/ (date accessed: 20.06.2022).

277. Lozhnikov, P. Cloud biometrical system identification through handwriting dynamics «SignToLogin» : Certificate of registration № ТХ 7-640-429 / P. Lozhnikov, A. Sulavko. – 18.12.2012.

278. Lozhnikov, P. S. Application of noise tolerant code to biometric data to verify the authenticity of transmitting information / P. S. Lozhnikov, A. E. Sulavko, D. A. Volkov // International Siberian Conference on Control and Communications (SIBCON) (Omsk, 21–23 May 2015). – IEEE, 2015. – P. 1–3. – DOI: 10.1109/SIBCON.2015.7147126.

279. Lozhnikov, P. S. Generation of a biometrically activated digital signature based on hybrid neural network algorithms / P. S. Lozhnikov, A. E. Sulavko // Journal of Physics: Conference Series. – 2018. – Vol. 1050. – P. 012047. – DOI: 10.1088/1742-6596/1050/1/012047.

280. Lozhnikov, P. S. Personal Identification and the Assessment of the Psychophysiological State While Writing a Signature / P. S. Lozhnikov, A. E. Sulavko, A. E. Samotuga // Information. – 2015. – Vol. 6. – P. 454–466. – DOI: 10.3390/info6030454.

281. Lozhnikov, P. S. Usage of fuzzy extractors in a handwritten-signature based technology of protecting a hybrid document management system / P. S. Lozhnikov, A. E. Sulavko, D. A. Volkov // 10th International Conference on Application of Information and Communication Technologies (AICT), 12–14 October 2016. – Baku, 2016. – P. 395–400. – DOI: 10.1109/ICAICT.2016.7991728.

282. Lozhnikov, P. S. Usage of quadratic form networks for users' recognition by dynamic biometrie images / P. S. Lozhnikov, A. E. Sulavko // 2017 Dynamics of Systems, Mechanisms and Machines (Dynamics) : conference proceedings (Omsk, 14–16 November 2017) / Omsk State Technical University. – IEEE, 2017. – P. 1–6. – DOI: 10.1109/Dynamics.2017.8239480.

283. Luh, G. Face recognition based on artificial immune networks and principal component analysis with single training image per person / G. Luh // Immune Computation. – 2014. – Vol. 2 (1). – P. 21–34.

284. Mahto, Shivangi. Ear Acoustic Biometrics Using Inaudible Signals and Its Application to Continuous User Authentication / Shivangi Mahto, T. Arakawa, Takafumi Koshinaka // 26th European Signal Processing Conference (Rome, Italy, 3–7 September 2018). – IEEE, 2018. – P. 1407–1411.

285. Maiorana, E. Fuzzy commitment for function based signature template protection / E. Maiorana, P. Campisi // IEEE Signal Processing Letters. – 2010. – Vol. 17. – P. 249–252.

286. Mansor, M. A. Artificial immune system algorithm with neural network approach for social media performance metrics / M. A. Mansor, S. Sathasivam, M. S. M. Kasihmuddin // AIP Conference Proceedings. – 2018. – Vol. 1974, no. 1. – P. 020072. – DOI: 10.1063/1.5041603.

287. Marshalko, G. B. On the security of a neural network-based biometric authentication scheme / G. B. Marshalko. // Matematicheskie. Voprosy. Kriptografii. – 2014. – Vol. 5, № 2. – P. 87–98. – DOI: https://doi.org/10.4213/mvk120.

288. Mascord, D. J. Behavioral and physiological indices of fatigue in a visual tracking task / D. J. Mascord, R. A. Heath // Journal of Safety Research. – 1992. – Vol. 23. – P. 19–25.

289. Mathematical and information maintenance of biometric systems / Z. Boriev S. Sokolov, A. Nyrkov, A. Nekrasova // OP Conf. Series: Materials Science and Engineering. – 2016. – Vol. 124. – P. 012046. – DOI: 10.1088/1757-899X/124/1/012046.

290. Methods of generating key sequences based on keystroke dynamics / P. S. Lozhnikov, A. E. Sulavko, A. V. Eremenko, E. V. Buraya // Dynamics of Systems, Mechanisms and Machines : X International IEEE Scientific and Technical Conference (Omsk, 15–17 November 2016) /) / Omsk State Technical University. – IEEE, 2016. – P. 1–5. – DOI: 10.1109/Dynamics.2016.7819038.

291. Methods of Generating Key Sequences based on Parameters of Handwritten Passwords and Signatures / P. S. Lozhnikov, A. E. Sulavko, A. V. Eremenko, D. A. Volkov // Information. MDPI. – 2016. – Vol. 7 (4). – P. 59. – DOI: 10.3390/info704005.

292. Mishra, P. K. Artificial Immune System: State of the Art Approach / P. K. Mishra, M. Bhusry // International Journal of Computer Applications. – 2015. – Vol. 20, № 120. – P. 25–32. – DOI: 10.5120/21344-4357.

293. Muda, A. K. A Framework of Artificial Immune System in Writer Identification / A. K. Muda, S. M. Shamsuddin // International Symposium of Bio-inspired Computing. – Johor Bahru, Malaysia, 2005. – URL: http://eprints.utm.my/id/eprint/577/1/04.pdf (date accessed: 04.04.2022).

294. Muda, N. A. D. Bio-inspired audio content-based retrieval framework (B-ACRF) / N. A. D. Muda, C. C. Wilson, S. Ling // Proceedings of the World Academy of Science, Engineering and Technology. – 2009. – Vol. 53. – P. 791–796.

295. Mulionoa, Y. Keystroke Dynamic Classification using Machine Learning for Password Authorization / Yohan Mulionoa, Hanry Hamb, Dion Darmawan // Procedia Computer Science. – 2018. – Vol. 135. – P. 564–569.

296. Multi-biometric template protection based on Homomorphic Encryption / M. Gomez-Barrero, E. Maiorana, J. Galbally [et al.] // Pattern Recognition. – 2017. – Vol. 67. – P. 149–163.

297. Neural network biometric cryptography system / A.M. Vulfin, V.I. Vasilyev, A.V. Nikonov, A.D. Kirillova // Proceedings of the Information Technologies and Intelligent Decision Making Systems (ITIDMS2021) (January 20, 2021). CEUR. – 2021. – Vol-2843.

298. Neuroscience / Dale Purves, George J. Augustine, David Fitzpatrick, William C. Hall [et al.] ; Ed. by D. Purves. – 5th edition. – New York : Sinauer Associates, Inc., 2018. – 759 p.

299. On dynamic feature weighting for feature drifting data streams / J. P. Barddal, H. M. Gomes, F. Enembreck [et al.] // Joint european conference on machine learning and knowledge discovery in databases. – 2016. – Vol. LNAI 9852. – P. 129–144. – DOI: 10.1007/978-3-319-46227-1_9.

300. On the Performance of Indirect Encoding Across the Continuum of Regularity / J. Clune, K. O. Stanley, R. T. Pennock, C. Ofria // IEEE Transactions on Evolutionary Computation. – 2011. – Vol. 15 (3). – P. 346–367. – DOI: 10.1109/TEVC.2010.2104157.

301. On the Reconstruction of Face Images from Deep Face Templates / Guangcan Mai, Kai Cao, Pong C. Yuen, Anil K. Jain // IEEE Transactions on Pat-tern Analysis and Machine Intelligence. – 2019. – Vol. 41 (5). – P. 1188–1202.

302. On-line Handwritten Signature Verification Based on Two Levels Back Propagation Neural Network / Zhan Enqi, Guo Jinxu, Zheng Jianbin [et al.] // International Symposium on Intelligent Ubiquitous Computing and Education (Chengdu, China, 15–16 May 2009). – IEEE, 2009. – P. 202–2005.

303. Online Handwritten Signature Verification Using Neural Network Classifier Based on Principal Component Analysis / V. Iranmanesh [et al.] // Scientific World Journal. – 2014. – Vol. 2014. – P. 1–8.

304. Online learning: A comprehensive survey / C. H. Hoi Steven, Sahoo Doyen, Lu Jing, Zhao Peilin // Neurocomputing. – 2021. – Vol. 459. – P. 249–289.

305. Optimizing deep learning hyper-parameters through an evolutionary algorithm / S. R. Young, D. C. Rose, T. P. Karnowsky [et al.] // Proceedings of the workshop on

machine learning in high-performance computing environments. – New York, 2015. – DOI: 10.1145/2834892.2834896.

306. Packed Homomorphic Encryption Based on Ideal Lattices and Its Application to Biometrics / M. Yasuda, T. Shimoyama, J. Kogure, K. Yokoyama, T. Koshiba // CD-ARES 2013: Security Engineering and Intelligence Informatics : International Conference on Availability, Reliability, and Security. – P. 55–74. – DOI: 10.1007/978-3-642-40588-4_5.

307. Patent № 578787 U.S., IPC G 06 V 40/10. Systems and methods for biometric identification using the acoustic properties of the ear canal : application 01.04.1996 : publ. 28.07.1998 / A. M. Bouchard, G. C. Osbourn.

308. Perspectives of subjects' psychophysiological state identification using dynamic biometric features / P. S. Lozhnikov, A. E. Sulavko, R. V. Borisov, S. S. Zhumazhanova // Journal of Physics: Conference Series. – 2018. – Vol. 1050 (1). – P. 012046.

309. Pflug, A. Ear biometrics: a survey of detection, feature extraction and recognition methods / A. Pflug, C. Busch // ET biometrics. – 2012. – Vol. 1 (2). – P. 114–129.

310. Possibility of Decrease in a Level of Data Correlation During Processing Small Samples Using Neural Networks by Generating New Statistic Tests / A. I. Ivanov, A. G. Bannykh, P. S. Lozhnikov, A. E. Sulavko, D. P. Inivatov // Journal of Physics: Conference Series. – 2020. – Vol. 1546. – P. 012080. – DOI: 10.1088/1742-6596/1546/1/012080.

311. Prakash, S. An efficient ear recognition technique invariant to illumination and pose / S. Prakash, P. Gupta // Telecommunication Systems. – 2013. – Vol. 52 (3). – P. 1435–1448.

312. Privacy-Preserving Deep Learning via Additively Homomorphic Encryption / L. T. Phong, Y. Aono, T. Hayashi [et al.]. // IEEE Transactions on Information Forensics and Security. – 2018. – Vol. 13, no. 5. – P. 1333–1345. – DOI: 10.1109/TIFS.2017.2787987.

313. Privacy-protected biometric templates: Acoustic ear identification / P. T. Tuyls, E. Verbitskiy, T. Ignatenko [et al.] // Biometric Technology for Human Identification International Society for Optics and Photonics. – 2004. – Vol. 5404. – P. 176–182.

314. Probst, Rudolf. Basic Otorhinolaryngology: A Step-by-Step Learning Guide Paperback / Rudolf Probst, Gerhard Grevers, Heinrich Iro. – 2nd edition. – Thieme, 2017. – 430 p.

315. Protasov, V. A Method for Evolutionary Decision Reconciliation, and Expert Theorems / V. Protasov, Z. Potapova, E. Melnikov // The Third International Conference on Intelligent Systems and Applications (INTELLI 2014). – Seville, Spain, 2014. – P. 43–47.

316. Protasov, V. A Method for Evolutionary Decision Reconciliation, and Expert Theorems / V. Protasov, Z. Potapova, E. A. Melnikov // The Third International Conference on Intelligent Systems and Applications INTELLI (Seville, Spain, June 22–26, 2014). – 2014. – P. 43–47.

317. Ratha, N. K. Enhancing security and privacy in biometrics–based authentication systems / N. K. Ratha, J. H. Connell, R. M. Bolle // IBM Systems Journal. – 2001. – Vol. 40 (3). – P. 614–634.

318. Rogers, L. L. Optimization of Groundwater Remediation Using Artificial Neural Networks With Parallel Solute Transport Modeling / L. L. Rogers, F. U. Dowla // Water Resources Research. – 1994. – Vol. 30. – P. 457–481.

319. Roy, N. D. Fast and robust retinal biometric key generation using deep neural nets / N. D. Roy, A. Biswas // Multimedia Tools and Applications. – 2020. – Vol. 79 (9). – P. 6823–6843.

320. Salehghaffari, Hossein. Speaker Verification using Convolutional Neural Networks / Hossein Salehghaffari. – URL: https://www.semanticscholar.org/reader/7debfbd229a19c755e2282dfee1d501d81907d36 (date accessed: 01.02.2022).

321. Security and Accuracy of Fingerprint-Based / W. Yang, S. Wang, J. Hu [et al.] // Biometrics: A Review. Symmetry. – 2019. – Vol. 11 (2). – P. 141.

322. Sheluhin, O. I. The online classification of the mobile applications traffic using data mining techniques / O. I. Sheluhin, V. V Barkov, S. A. Sekretarev // T-Comm. – 2019. – Vol. 13, no.10. – P. 60–67. – DOI 10.24411/2072-8735-2018-10317.

323. Smith, By Zhanna Malekos. The Hidden Costs of Cybercrime / Zhanna L. Malekos Smith, E. Lostri, J. A. Lewis. – McAfee, 2020. – URL: https://www.mcafee.com/enterprise/en-us/assets/reports/rp-hidden-costs-of-cybercrime.pdf (date accessed: 16.12.2021).

324. Souza, Victor L. F. A writer-independent approach for offline signature verification using deep convolutional neural networks features / Victor L. F. Souza, Adriano L. I. Adriano, Robert Sabourin // 7th Brazilian Conference on Intelligent Systems (BRACIS) (Sao Paulo, 22–25 October 2018). – IEEE, 2018. – DOI: 10.1109/BRACIS.2018.00044.

325. Speaker identification and clustering using convolutional neural networks / Yanick Lukic, Carlo Vogt, Oliver D¨urr, Thilo Stadelmann // 26th International Workshop on Machine Learning for Signal Processing (MLSP) (Salerno, Italy, 13–16 September 2016). – IEEE, 2016. – DOI: 10.1109/MLSP.2016.7738816.

326. Stanley, K. O. Efficient Evolution of Neural Networks Through Complexification : PhD Thesis / K. O. Stanley ; Department of Computer Sciences, The University of Texas at Austin. – Austin, Texas USA, 2004. – 165 p.

327. Statistical approach for subject's state identification by face and neck thermograms with small training sample / S. S. Zliumazhanova, A. E. Sulavko, D. B. Ponomarev, V. A. Pasenchuk // IFAC-PapersOnLine. – 2019. – Vol. 52 (25). – P. 46–51. – DOI: 10.1016/j.ifacol.2019.12.444.

328. Strike (With) a Pose: Neural Networks Are Easily Fooled by Strange Poses of Familiar Objects / Michael A. Alcorn, Qi Li, Zhitao Gong [et al.] // Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (Long Beach, 15–20 June 2019). – IEEE, 2019. – P. 4845–4854. – DOI: 10.1109/CVPR.2019.00498.

329. Subjects Authentication Based on Secret Biometric Patterns Using Wavelet Analysis and Flexible Neural Networks / A. E. Sulavko, D. A. Volkov, S. S. Zhumazhanova, R. V. Borisov // XIV International Scientific-Technical Conference on Actual Problems of Electronics Instrument Engineering (APEIE) (Novosibirsk, 2–6 October 2018). – IEEE, 2018. – P. 218–227. – DOI: 10.1109/APEIE.2018.8545676.

330. Sulavko, A. E. Bayes-Minkowski measure and building on its basis immune machine learning algorithms for biometric facial identification / A. E. Sulavko // Journal of Physics: Conf. Series. – 2020. – Vol. 1546. – P. 012103-1–012103-7. – DOI: 10.1088/1742-6596/1546/1/012103.

331. Sulavko, A. E. Biometric pattern recognition using wide networks of gravity proximity measures / A. E. Sulavko, S. S. Zhumazhanova // Journal of Physics: Conference Series. – 2018. – Vol. 1050. – P. 012082-1–012082-13. – DOI: 10.1088/1742-6596/1050/1/012082.

332. Sulavko, A. E. Biometric-Based Key Generation and User Authentication Using Acoustic Characteristics of the Outer Ear and a Network of Correlation Neurons / A. E. Sulavko // Sensors. – 2022. – Vol. 22. – P. 9551. – DOI: 10.3390/s22239551.

333. Sulavko, A. E. Personal Identification Based on Acoustic Characteristics of the Outer Ear Using Cepstral Analysis, Bayesian Classifier and Artificial Neural Networks / A. E. Sulavko, A. E. Samotuga, I. A. Kuprik // IET Biometrics. – 2021. – Vol. 10 (6). – P. 692–705. – DOI: 10.1049/bme2.12037.

334. Sulavko, A. E. Perspective Neural Network Algorithms for Dynamic Biometric Pattern Recognition in the Space of Interdependent Features / A. E. Sulavko, S. S. Zhumazhanova, G. A. Fofanov // Dynamics of Systems, Mechanisms and Machines : conference proceeding (Omsk, 13–15 November 2018) / Omsk State Technical University. – IEEE, 2018. – P. 1–12. – DOI: 10.1109/Dynamics.2018.8601440.

335. Sulavko, A. E. Users' identification through keystroke dynamics based on vibration parameters and keyboard pressure / A. E. Sulavko, A. A. Fedotov, A. V. Eremenko // Dynamics of Systems, Mechanisms and Machines: conference proceeding (Omsk , 14–16 November 2017) / Omsk State Technical University. – IEEE, 2017. – P. 1–7. – DOI: 10.1109/Dynamics.2017.8239514.

336. Sulavko, A. E. Comparison of functionals based on statistic tests for generating fast learning wide neural networks / A. E. Sulavko // Инфографика и информационный дизайн: визуализация данных в науке : материалы Междунар. науч.-практ. конф. (Омск, 17–18 нояб. 2017 г.) / Ом. гос. техн. ун-т. – Омск : Изд-во ОмГТУ, 2017. – С. 210–223.

337. Sulavko, A. E. Human psychophysiological state recognition based on analysis of thermograms of face and neck regions / A. E. Sulavko, S. S. Zhumazhanova // Dynamics of Systems, Mechanisms and Machines: conference proceedings (Omsk, 14–16 November 2017) / Omsk State Technical University. – IEEE, 2017. – DOI: 10.1109/Dynamics.2017.8239515.

338. Sun, Y. An artificial neural network framework for gait-based biometrics / Y. Sun, B. Lo // IEEE journal of biomedical and health informatics. – 2018. – Vol. 23 (3). – P. 987–998.

339. Tachibana, Hideyuki. Efficiently Trainable Text-to-Speech System Based on Deep Convolutional Networks with Guided Attention / Hideyuki Tachibana, Katsuya Uenoyama, Shunsuke Aihara // International Conference on Acoustics, Speech and Signal Processing (ICASSP) (Calgary, AB, 15–20 April 2018). – IEEE, 2018. – DOI: 10.1109/ICASSP.2018.8461829.

340. Tan, Y. Y. Wavelet Theory and its Application to Pattern Recognition / Y. Y. Tan, J. Liu, L. H. Yang, H. Ma. – World Scientific, 2000. – 344 p.

341. Text-dependent speaker verification: Classifiers, databases and RSR2015 / Anthony Larcher, Kong Aik Lee, Bin Ma, Haizhou Li // Speech Communication. – 2014. – Vol. 60. – P. 56–77.

342. The Application of a Dendritic Cell Algorithm to a Robotic Classifier / R. Oates, L. N. de Castro, F. J. Von Zuben [et al.] // International Conference on Artificial Immune Systems. – Springer, Berlin, Heidelberg, 2007. – C. 204–215.

343. The RedDots Data Collection for Speaker Recognition / Kong Aik Lee, Anthony Larcher, Guangsen Wang [et al.]. – URL: https://isca-speech.org/archive/pdfs/interspeech_2015/lee15_interspeech.pdf (date accessed: 01.02.2022).

344. THRIVE: threshold homomorphic encryption based secure and privacy preserving biometric verification system / C. Karabat, M. S. Kiraz, H. Erdogan [et al.] // EURASIP Jornal on Advances in Signal Processing. – 2015. – Vol. 71. – URL: https://doi.org/10.1186/s13634-015-0255-5. – (date accessed: 07.04.2021).

345. Timmis, J. Challenges for Artificial Immune Systems / J. Timmis. – DOI: 10.1007/11731177_42 // Neural Nets. – Berlin/Heidelberg : Springer-Verlag, 2006. – P. 355–367.

346. Torfi, Amirsina. Text-independent speaker verification using 3d convolutional neural networks / Amirsina Torfi, Jeremy Dawson, Nasser M. Nasrabadi // IEEE International Conference on Multimedia and Expo (ICME), 23–27 July 2018. IEEE, 2018. – DOI: 10.1109/ICME.2018.8486441.

347. Torres, Wilson Abel Alberto. Effectiveness of Fully Homomorphic Encryption to Preserve the Privacy of Biometric Data / Wilson Abel Alberto Torres, Nandita Bhattacharjee, Bala Srinivasan // 16th International Conference on Information Integration and Web-based Applications & Services (iiWAS '14). Association for Computing Machinery. – New York, USA, 2014. – P. 152–158. – DOI: https://doi.org/10.1145/2684200.2684296.

348. Unlinkable improved multibiometric iris fuzzy vault / C. Rathgeb, B. Tams, J. Wagner, C. Busch // EURASIP Journal on Information Security. – 2016. – Vol. 1. – P. 1–16.

349. Utterance Verification for Text-Dependent Speaker Recognition: A Comparative Assessment Using the RedDots Corpus / T. Kinnunen, M. Sahidullah, I. Kukanov [et al.]. – URL: https://www.researchgate.net/publication/303922314_Utterance_Verification_for_Text-Dependent_Speaker_Recognition_A_Comparative_Assessment_Using_the_RedDots_Corpus (date accessed: 01.02.2022).

350. Verbancsics, P. Image classification using generative neuroevolution for deep learning / P. Verbancsics, J. Harguess // IEEE Winter Conference on Applications of Computer Vision. – Waikoloa, Hawaii, USA, 2015. – P. 488–493.

351. Voxceleb: Large-scale speaker verification in the wild / Arsha Nagrani, Joon Son Chung, Weidi Xie, Andrew Zisserman // Computer Speech & Language. – 2020. – Vol. 60. – P. 101027.

352. Wang, Lei. A Novel Neural Network Based on Immunity / Lei Wang, Michele Courant // Proceedings of the International Conference on Artificial Intelligence (IC-AI'02). – Las Vegas, Nevada, 2002. – P. 147–153.

353. Xiao, R. B. A Framework of AIS Based Pattern Classification and Matching for Engineering Creative Design / R. B. Xiao, L. Wang, Y. Liu // Proceedings of the First 7 International Conference on Machine Learning and Cybernetics. – Beijing : IEEE, 2002. – P. 1554–1558.

354. Xiao, Yu. Pipeline image diagnosis algorithm based on neural immune ensemble learning / Yu Xiao, Lu YuHua, Gao Qiang // International Journal of Pressure Vessels and Piping. – 2021. – Vol. 189. – P. 104249.

355. Yan, P. Biometric recognition using 3D ear shape / P. Yan, K. W. Bowyer // IEEE Transactions on pattern analysis and machine intelligence. – 2007. – Vol. 29 (8). – P. 1297–1308.

356. Yasuoka, Yuto. Evaluation of Optimization Methods for Neural Network / Yuto Yasuoka, Yuki Shinomiya, Yukinobu Hoshino // Joint 8th International Conference on Soft Computing and Intelligent Systems (SCIS) and 17th International Symposium on Advanced Intelligent Systems (Sapporo, 25–28 August 2016. – IEEE, 2016. – DOI: 10.1109/SCIS-ISIS.2016.0032.

357. Yu, S. Conserved self pattern recognition algorithm / S. Yu, D. Dasgupta // International Conference on Artificial Immune Systems. – Berlin, Heidelberg : Springer, 2008. – P. 279–290.

358. Yuan, L. Ear detection based on improved adaboost algorithm / L. Yuan, F. Zhang // International Conference on Machine Learning and Cybernetics. – IEEE, 2009. – Vol. 4. – P. 2414–2417.

359. Yuan, L. Ear recognition based on Gabor features and KFDA / L. Yuan, Z. Mu // The Scientific World Journal. – 2014. – Vol. 2014 (2). – P. 702076. – DOI: 10.1155/2014/702076.

360. Zhang, G. Ensemble Machine Learning. Methods and Applications / C. Zhang, Y. Ma. – Boston : Springer, 2012. – 329 p. – ISBN 978-1-4419-9325-0. – DOI: 10.1007/978-1-4419-9326-7.

361. Zhou, J. An efficient 3-D ear recognition system employing local and holistic features / J. Zhou, S. Cadavid, M. Abdel-Mottaleb // IEEE transactions on Information Forensics and Security. – 2012. – Vol. 7 (3). – P. 978–991.

362. Zhu, Q. Local and Holistic Feature Fusion for Occlusion-Robust 3D Ear Recognition / Q. Zhu, Z. Mu // Symmetry. – 2018. – Vol. 10 (11). – P. 565.

Приложение 1 Результаты дополнительных экспериментов по анализу и классификации биометрических образов

Исследование свойств функционалов для построения на их основе мер близости и нейронов.

Таблица 1 – Функционалы для сравнения эмпирической функции вероятности $P(\hat{a})$ с ее эталонным описанием $\dot{P}(\hat{a})$ (функции плотности вероятности $p(\hat{a})$ с ее эталонным описанием $\dot{p}(\hat{a})$) с заданной точностью κ

Краткое обозначение	Название	Формула
$HiI(\kappa)$	Хи-квадрат Пирсона (интегральный)	$HiI = \int_{-\infty}^{+\infty} \frac{(P(\hat{a}) - \dot{P}(\hat{a}))^2}{\dot{P}(\hat{a})} d\hat{a},$
$HiD(\kappa)$	Хи-квадрат Пирсона (дифференциальный)	$HiD = \int_{-\infty}^{+\infty} \frac{(p(\hat{a}) - \dot{p}(\hat{a}))^2}{\dot{p}(\hat{a})} d\hat{a},$
$KfMI(\kappa)$	Крамера-фон Мизеса (интегральный)	$KfMI = \int_{-\infty}^{+\infty} (P(\hat{a}) - \dot{P}(\hat{a}))^2 d\hat{a},$
$KfMD(\kappa)$	Крамера-фон Мизеса (дифференциальный)	$KfMD = \int_{-\infty}^{+\infty} (p(\hat{a}) - \dot{p}(\hat{a}))^2 d\hat{a},$
$SKfMI(\kappa)$	Смирнова-Крамера-фон Мизеса (интегральный)	$SKfMI = \int_{-\infty}^{+\infty} (P(\hat{a}) - \dot{P}(\hat{a}))^2 d\dot{P}(\hat{a}),$
$SKfMD(\kappa)$	Смирнова-Крамера-фон Мизеса (дифференциальный)	$SKfMD = \int_{-\infty}^{+\infty} (p(\hat{a}) - \dot{p}(\hat{a}))^2 \dot{p}(\hat{a}) d\hat{a},$
$DgI(\kappa)$	Джини (интегральный)	$DgI = \int_{-\infty}^{+\infty} \lvert P(\hat{a}) - \dot{P}(\hat{a}) \rvert d\hat{a},$
$DgD(\kappa)$	Джини (дифференциальный)	$DgD = \int_{-\infty}^{+\infty} \lvert p(\hat{a}) - \dot{p}(\hat{a}) \rvert d\hat{a},$
$ADI(\kappa)$	Андерсона-Дарлинга (интегральный)	$ADI = \int_{-\infty}^{+\infty} \frac{(P(\hat{a}) - \dot{P}(\hat{a}))^2}{\dot{P}(\hat{a}) \cdot (1 - \dot{P}(\hat{a}))} d\dot{P}(\hat{a}),$
$ADD(\kappa)$	Андерсона-Дарлинга (дифференциальный)	$ADD = \int_{-\infty}^{+\infty} \frac{(p(\hat{a}) - \dot{p}(\hat{a}))^2}{\dot{p}(\hat{a}) \cdot (1 - \dot{p}(\hat{a}))} \dot{p}(\hat{a}) d\hat{a},$
$ADID(\kappa)$	Андерсона-Дарлинга (интегро-дифференциальный)	$ADID = \int_{-\infty}^{+\infty} \frac{(p(\hat{a}) - \dot{p}(\hat{a}))^2}{\dot{P}(\hat{a}) \cdot (1 - \dot{P}(\hat{a}))} \dot{p}(\hat{a}) d\hat{a},$
$VI(\kappa)$	Ватсона (интегральный)	$VI = \int_{-\infty}^{+\infty} (\dot{P}(\hat{a}) - P(\hat{a}) - \int_{-\infty}^{+\infty} [\dot{P}(\hat{a}) - P(\hat{a})] d\dot{P}(\hat{a}))^2 d\dot{P}(\hat{a}),$

Продолжение таблицы 1

VD(κ)	Ватсона (дифференциальный)	$VD = \int_{-\infty}^{+\infty} (\dot{p}(\hat{a}) - p(\hat{a}) - \int_{-\infty}^{+\infty} [\dot{p}(\hat{a}) - p(\hat{a})]\dot{p}(\hat{a})d\hat{a})^2 \dot{p}(\hat{a})d\hat{a},$
FrI(κ)	Фроцини (интегральный)	$FrI = \int_{-\infty}^{+\infty} \lvert P(\hat{a}) - \dot{P}(\hat{a}) \rvert \cdot d\dot{P}(\hat{a}),$
FrD(κ)	Фроцини (дифференциальный)	$FrD = \int_{-\infty}^{+\infty} \lvert p(\hat{a}) - \dot{p}(\hat{a}) \rvert \cdot \dot{p}(\hat{a})d\hat{a},$
sgI(κ)	среднего геометрического вероятности	$sgI = \int_{-\infty}^{+\infty} \sqrt{P(\hat{a}) \cdot (1 - \dot{P}(\hat{a}))}d\hat{a},$
sgD(κ)	среднего геометрического плотности вероятности	$sgD = \int_{-\infty}^{+\infty} \sqrt{p(\hat{a})\dot{p}(\hat{a})}d\hat{a},$
sgI^2(κ)	квадрата среднего геометрического вероятности	$sgI\hat{}2 = \int_{-\infty}^{+\infty} P(\hat{a}) \cdot (1 - \dot{P}(\hat{a}))d\hat{a},$
sgD^2(κ)	квадрата среднего геометрического плотности вероятности	$sgD\hat{}2 = \int_{-\infty}^{+\infty} p(\hat{a})\dot{p}(\hat{a})d\hat{a},$
KSD(κ) или *KS(κ)*	Колмогорова-Смирнова (дифференциальный)	$KS = \sup_{-\infty<\hat{a}<+\infty} (p(\hat{a}) - \dot{p}(\hat{a})),$
KuD(κ) или *Ku(κ)*	Кёйпера (Купера) (дифференциальный)	$Ku = \sup_{-\infty<\hat{a}<+\infty} (p(\hat{a}) - \dot{p}(\hat{a})) - \inf_{-\infty<\hat{a}<+\infty} (p(\hat{a}) - \dot{p}(\hat{a})),$
MaxSq(κ)	максимума площади пересечения сравниваемых функций плотностей вероятности	$MaxSq = 1 - \int_{-\infty}^{+\infty} (\min(\hat{a}) = \begin{cases} p(\hat{a}), если\ p(\hat{a}) < \dot{p}(\hat{a}) \\ \dot{p}(\hat{a}), если\ p(\hat{a}) > \dot{p}(\hat{a}) \end{cases})d\hat{a},$

Ряд функционалов строятся на формуле гипотез Байеса – метод последовательного применения формулы Байеса (МППФБ) и многомерный функционал наибольшего правдоподобия (МФНПБ), который для случая верификации (2-х гипотез) можно представить в виде:

$$P_h(\bar{a}) = \frac{0{,}5\prod_{j=1}^{N} p_h(a_j)}{\sum_{i=1}^{\Gamma}(0{,}5\prod_{j=1}^{N} p_i(a_j))} = \frac{\prod_{j=1}^{N} p_h(a_j)}{\sum_{i=1}^{\Gamma}\prod_{j=1}^{N} p_i(a_j)},$$

Решение в пользу *h*-ой гипотезы принимается аналогично – по максимальной апостериорной вероятности $P_h(\bar{a})$. Как можно видеть, оба функционала (МППФБ и МФНПБ) являются интегро-дифференциальными (использующими вероятности и плотности вероятностей значений признаков).

Проведен вычислительный эксперимент по сравнению надежности решений, принимаемых отдельными нейронами (не ИНС), формируемыми на основе приведенных выше функционалов. Для простоты был рассмотрен наиболее распространённый случай (для биометрических систем), когда признаки имеют нормальный закон распределения.

Для выбора граничных условий эксперимента следует ориентироваться на обобщенные показатели $AUC(\bar{a})$ биометрических признаков. Ориентироваться на высокоинформативные признаки отпечатка пальца, радужки нет смысла. Для них показатель $AUC(\bar{a})$ намного ниже 0,1 (в разы). С такими признаками легко справляются персептроны, обученные по стандартному алгоритму ГОСТ Р 52633.5-2011. Интерес представляют малоинформативные признаки. По имеющимся оценкам информативность признаков динамики подписи (получаемых алгоритмами спектрального, корреляционного и вейвлет анализа) составляет порядка $AUC(\bar{a})\approx 0{,}5$. Для спектральных признаков речевых паролей данная оценка составляет: $AUC(\bar{a})\approx 0{,}55$ (разные пароли), $AUC(\bar{a})\approx 0{,}65$ (фиксированная контрольная фраза). Для клавиатурного почерка оценки информативности почти аналогичны. Более информативными являются признаки лица с показателем $AUC(\bar{a})\approx 0{,}35$. Таким образом, средние обобщенные оценки информативности составляют примерно $0{,}3<AUC(\bar{a})<0{,}7$, для различных субъектов и признаков разброс показателя лежит в интервале $0{,}1<AUC(a_j)<0{,}9$.

Реальные биометрические данные коррелированы, однако, характер этой зависимости весьма сложен, сила связи между разными признаками одного биометрического образа всегда отличается. Имеющиеся обобщенные оценки этой зависимости для динамических биометрических образов указывают на то, что существуют как слабо зависимые пары признаков, так и признаки, зависимость между которыми значительна или весьма высока. Коррелированность признаков можно оценить по данным обучающей выборки через матрицу парных коэффициентов корреляции между векторами значений соответствующих признаков. К сожалению, на малых обучающих выборках (30 примеров и менее) имеются значительные погрешности в вычислении коэффициентов корреляции

биометрических данных, которые достигают ±0,65, что в некоторых случаях практически сводит на нет возможности качественной оценки корреляции между признаками. Чтобы компенсировать случайную погрешность предложено комбинировать несколько различных корреляционных функционалов либо использовать Байесовские корреляционные функционалы. Благодаря этому удается получить более приемлемые результаты оценки. Однако на выборках менее 20 примеров существенные погрешности вычислений все равно остаются. На практике на этапе обучения биометрической системы вполне осуществимым видится подход почти безошибочного разделения признаков на зависимые и условно независимые, используя в качестве маркера границу вычисляемого коэффициента корреляции от 0,5 до 0,6.

В рамках эксперимента было сгенерировано 4 набора из 100 признаков для 100 образов (по 100 значений каждого признака для каждого образа): независимые признаки с $AUC(\bar{a})\approx 0{,}3$ и $AUC(\bar{a})\approx 0{,}7$, коррелированные признаки с $AUC(\bar{a})\approx 0{,}3$ и $AUC(\bar{a})\approx 0{,}7$. Среднеквадратичные отклонения и математические ожидания каждого признака генерировались по нормальному закону методом Монте-Карло, среднеквадратичные отклонения на интервале [0,5; 1,5], разброс математических ожиданий в каждом случае ($AUC(\bar{a})\approx 0{,}3$ и $AUC(\bar{a})\approx 0{,}7$) выставлялся таким образом, чтобы обеспечить указанные уровни информативности. Разброс показателей информативности при $AUC(\bar{a})\approx 0{,}3$ составил примерно $0{,}1<AUC(a_j)<0{,}5$, при $AUC(\bar{a})\approx 0{,}7$ – $0{,}5<AUC(a_j)<0{,}9$. Чем ниже информативность, тем кучнее расположены распределения значений признака, характеризующие различные образы

Значения признаков во всех случаях также генерировались методом Монте-Карло (как независимые величины), потом для коррелированных признаков создавалась зависимость, для этого значения каждого признака для каждого образа ранжировались по возрастанию (менялся порядок значений для каждого признака – от наименьшего к наивысшему).

В ходе эксперимента менялся объем обучающей выборки, количество подаваемых на вход функционала признаков (размерность функционала).

Вероятность ошибок 1-го и 2-го рода определялась как количество ошибок определенного рода к числу соответствующих опытов. Представленные на рисунках результаты получены при значении точности, вычисляемом по правилу Стёрджеса для случая, когда эмпирическое и эталонные распределения всегда описывались плотностью вероятности нормального закона распределения:

$$\kappa = 1 + Log_2(N \cdot K),$$

где N – количество признаков, K – количество примеров обучающей выборки «Свой».

На малых обучающих выборках гистограммы относительных частот дают высокий процент шумов квантования. Поэтому описание всех распределений функциями плотностей вероятности нормального закона в большинстве случаев дает более высокие результаты. На вероятность ошибочных решений также влияет точность вычисления интегралов, причем на каждый функционал точность влияет по-разному. Оптимальное значение точности далеко не всегда может быть вычислено по правилу Стёрджеса. К примеру, критерий Джини дает лучшие результаты на низкой точности. В процессе эксперимента были найдены оптимальные параметры точности сравниваемых критериев для рассматриваемых случаев.

Каждый функционал может показывать хорошие результаты в определенных условиях использования (в зависимости от объема обучающей выборки, информативности и коррелированности признаков и др.). В частности, низкоразмерные квадратичные формы удовлетворительно работают, если признаки информативны и не коррелируют, высокоразмерные – наоборот, если признаки малоинформативны и коррелированы (корреляция «замедляет» ход формирования решения и этап насыщения наступает при поступлении на вход большего количества признаков). На малых объемах обучающей выборки меры Пирсона и хи-модуль работают плохо.

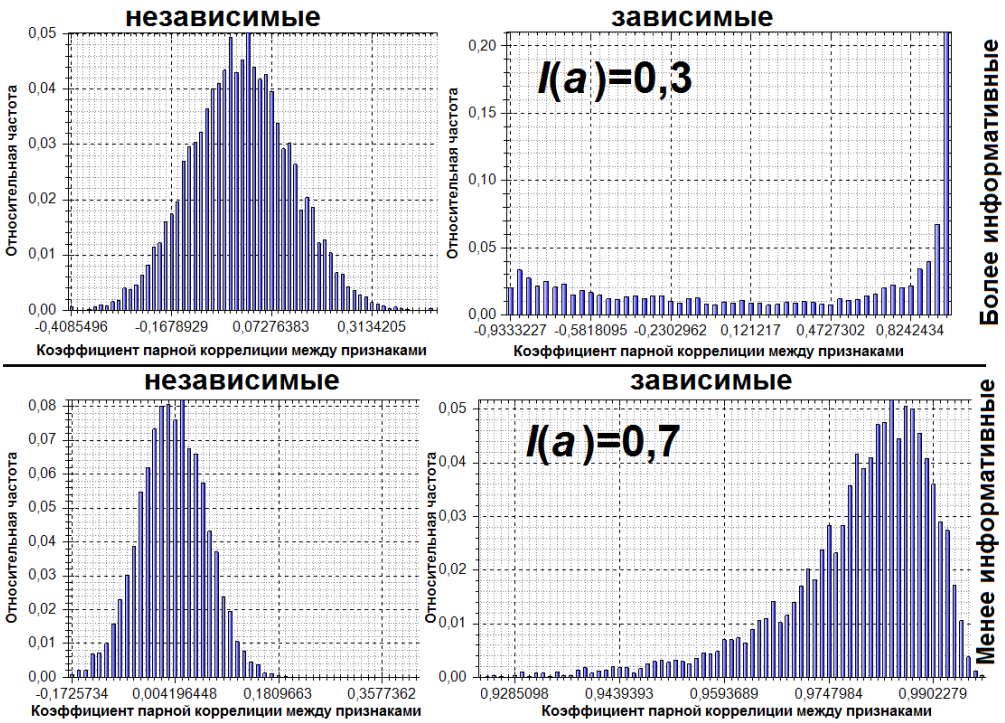


Рисунок 1 – Гистограммы относительных частот появления парных коэффициентов корреляции между сечениями зависимых и независимых признаков

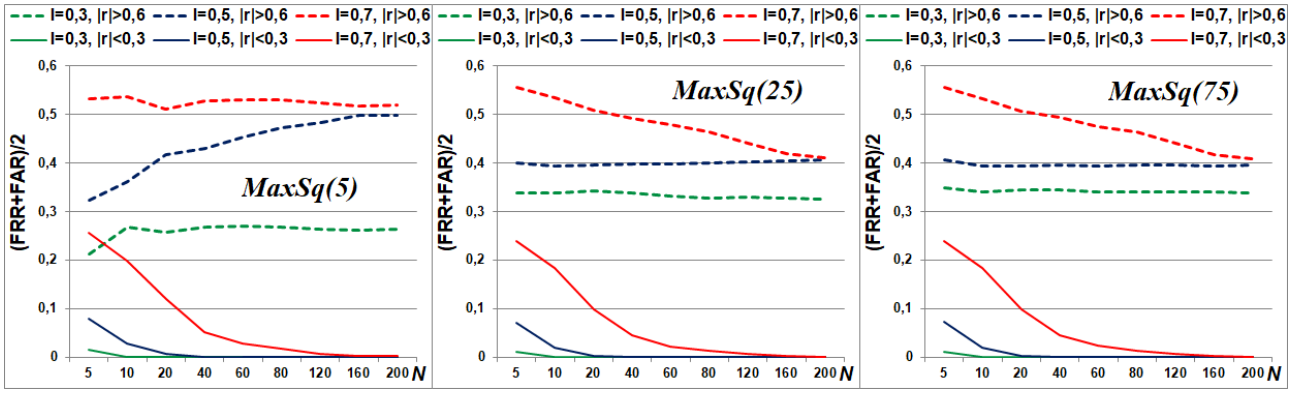


Рисунок 2 – Вероятности ошибок верификации образов при 0,3≤*AUC*≤0,7

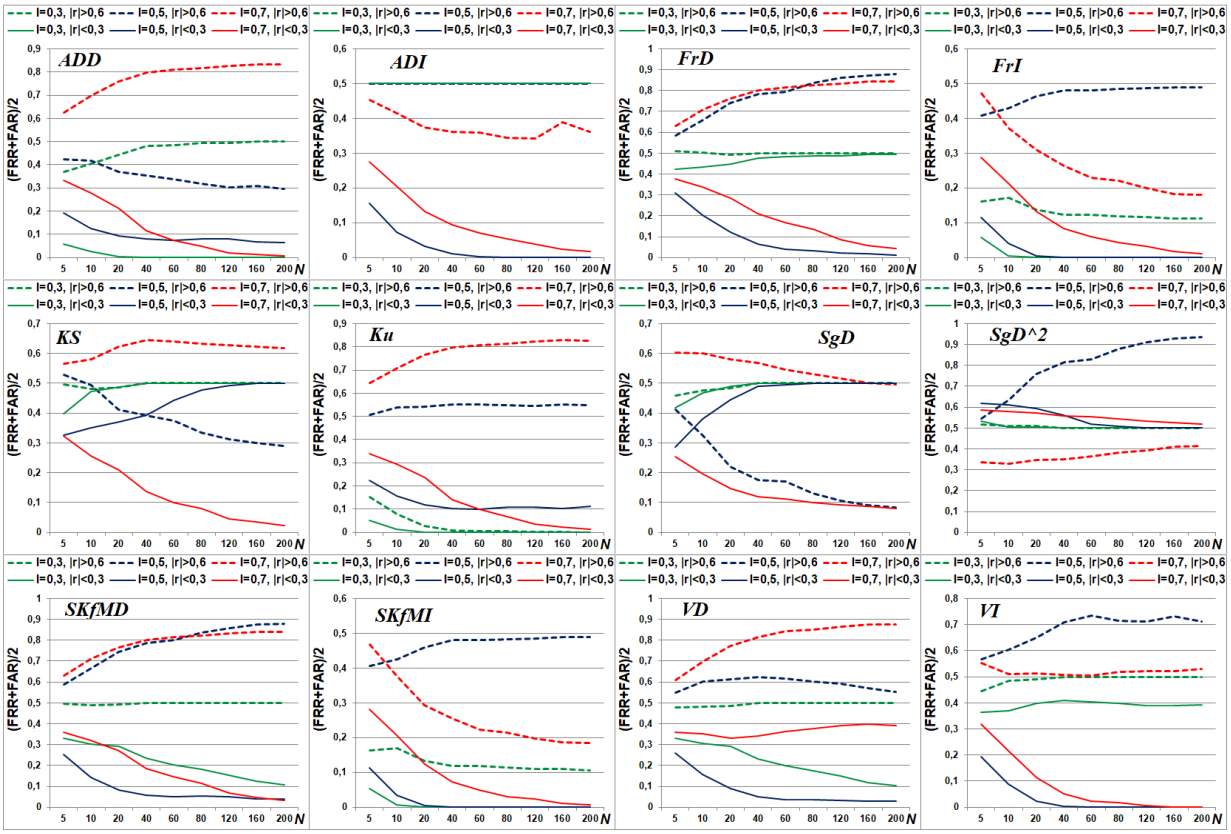


Рисунок 3 – Вероятности ошибок верификации образов при $0{,}3 \leq AUC(N) \leq 0{,}7$, $\kappa=5$

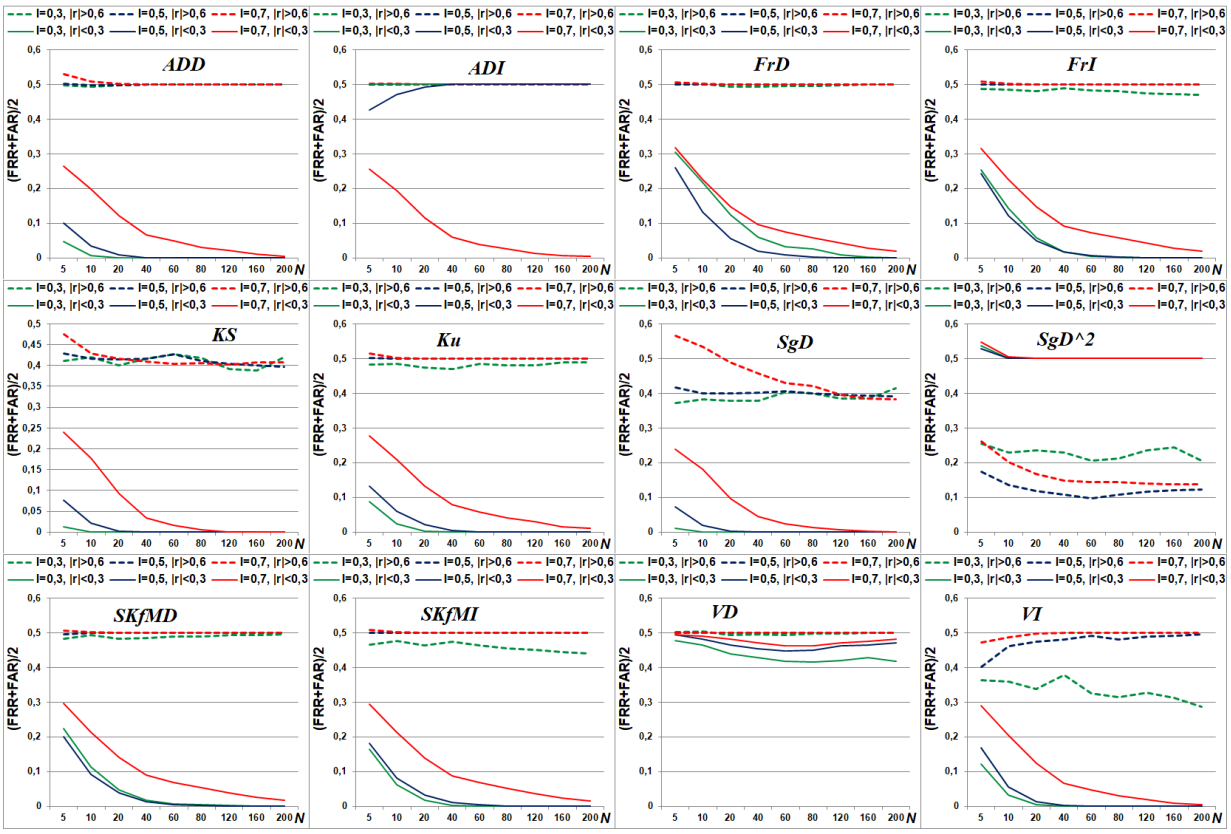


Рисунок 4 – Вероятности ошибок верификации образов при $0{,}3 \leq AUC(N) \leq 0{,}7$, $\kappa=25$

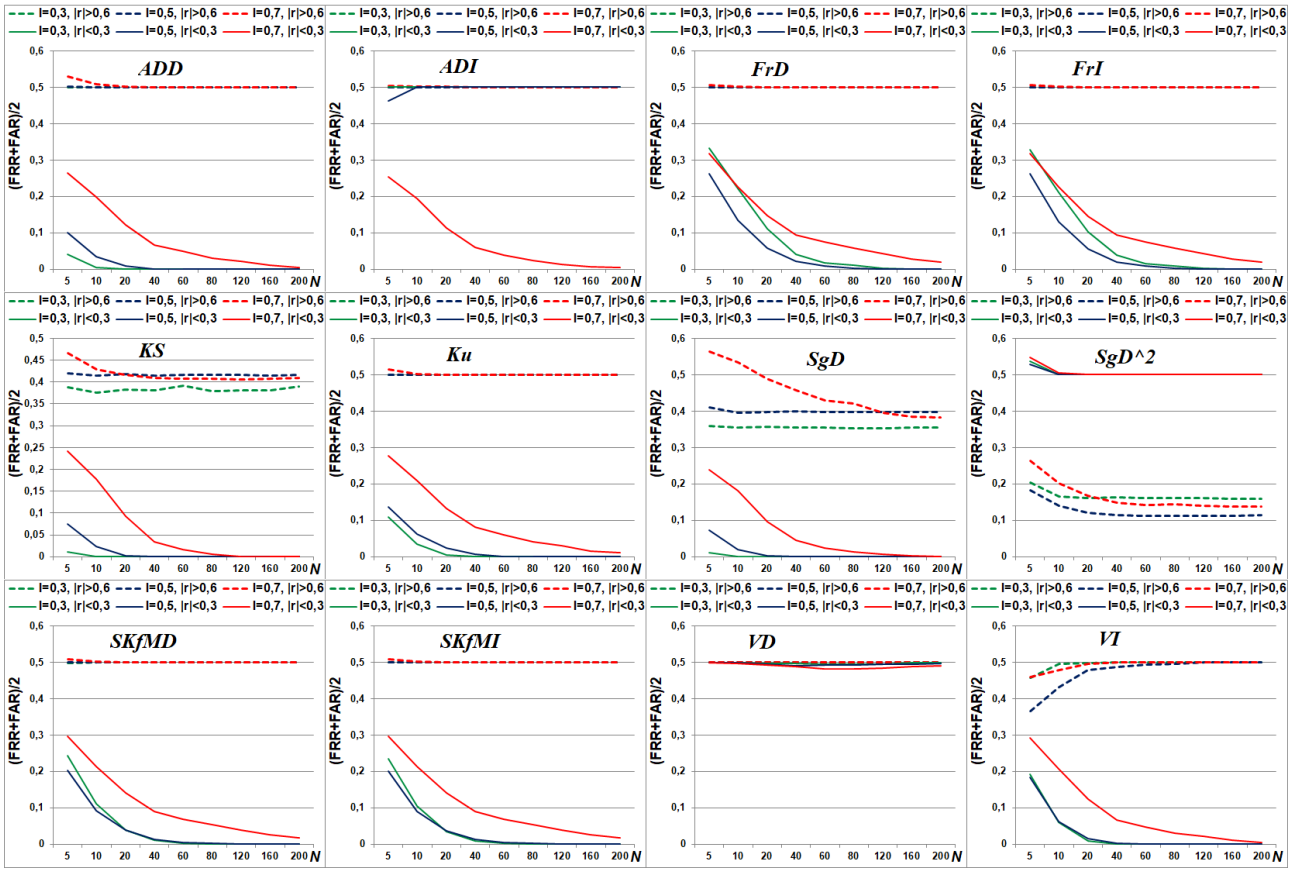


Рисунок 5 – Вероятности ошибок верификации образов при 0,3≤ $AUC(N)$≤0,7, κ=75

В случае независимых признаков функционал на базе среднего геометрического плотности вероятности (*sgD*) дает среднюю вероятность ошибок, близкую к единице. Следовательно, если инвертировать его решения можно получить эффективный обратный функционал (¬*sgD*).

Интересными являются также следующие результаты: в пространстве зависимых признаков функционалы sgI, sgD, sgD^2, МППФБ и МФНПБ эффективней работают, если признаки малоинформативны ($AUC(\bar{a})$≈0,7). Причем объем обучающей выборки слабо влияет на результаты работы sgI, sgD, sgD^2 (чего нельзя сказать о функционалах МППФБ и МФНПБ). Самым интересным является то, что для sgD^2 наилучшим вариантом оказывается малый объем обучающей выборки при малоинформативных и зависимых признаках, в этом случае данный функционал показывает результат, превосходящий по надежности другие меры близости.

По результатам эксперимента отобраны функционалы, показывающие наиболее высокую надежность при распознавании образов для рассматриваемых пространств признаков. Данные функционалы наряду с разностными, корреляционными и гиперболическими многомерными функционалами Байеса целесообразно использовать при формировании решений в широких нейронных сетях быстрого обучения.

Таблица 2 – Наилучшие функционалы для принятия решений по верификации образов в зависимости от особенностей пространства признаков

Объем обучающей выборки «Свой»	Зависимые $I(\bar{a})\approx 0{,}7$	Независимые $I(\bar{a})\approx 0{,}7$	Зависимые $I(\bar{a})\approx 0{,}3$	Независимые $I(\bar{a})\approx 0{,}3$
20 примеров	МППФБ ($N>50$)	KfMI, DgI, DgD, МППФБ, VI, ADI, FrI, SKfMI, KfMD, ¬*sgD* ($N>50$)	KfMI, SKfMI, DgI, FrI ($N>50$)	МППФБ ($N>10$), FrI, DgI, DgD, KfMI, KfMD, SKfMI, ADD, ¬*sgD* ($N>50$)
10 примеров	sgD^2 ($100>N>10$)	DgI, VI ($N>50$)	DgI ($N>50$)	МППФБ, KfMI, KfMD, DgI, DgD ($N>50$), SKfMI, FrI ($100>N\geq 50$)
5 примеров	sgD^2 ($100>N>10$)	DgI ($N>50$)	DgI ($N>50$)	KfMI, SKfMI, DgI, DgD ($N\geq 50$), FrI ($100>N\geq 50$)

Таблица 3 – Коррелированность решений некоторых функционалов

	KfMI	KfMD	SKfMI	DgI	DgD	VI	FrI	МППФБ
KfMI	1	0,240982	0,766498	0,943926	0,079987	0,023742	0,739911	-0,29598
KfMD	0,240982	1	0,094804	0,260159	0,610128	-0,01157	0,092743	-0,16349
SKfMI	0,766498	0,094804	1	0,729376	-0,00959	0,077408	0,964628	-0,37275
DgI	0,943926	0,260159	0,729376	1	0,095664	-0,06448	0,706891	-0,31354
DgD	0,079987	0,610128	-0,00959	0,095664	1	0,09147	-0,0064	-0,12156
FrI	0,739911	0,092743	0,964628	0,706891	-0,0064	0,076145	1	-0,38568

Не рекомендуется использовать совместно следующие пары функционалов, разделяющих одно и то же пространство признаков, т.к. их выходы могут быть существенно коррелированы: KfMI и DgI, SKfMI и FrI, KfMI и SKfMI, KfMI и FrI, SKfMI и DgI, DgI и FrI. При разделении различных пространств признаков (т.е. при подаче на вход этих функционалов непересекающихся сочетаний признаков) их совместное использование возможно.

Для решения проблемы проявления шумов квантования на малых обучающих выборках (если описывать эмпирическое и/или эталонное распределение гистограммой относительных частот, нормированной по длине

интервала) можно использовать способы сглаживания гистограмм. Исследования показывает, что прибегая к данным методикам, в ряде случаев удается получить существенный выигрыш по надежности решений, принимаемых рассмотренными статистическими функционалами (если описывать эмпирические распределения сглаженной гистограммой относительных частот).

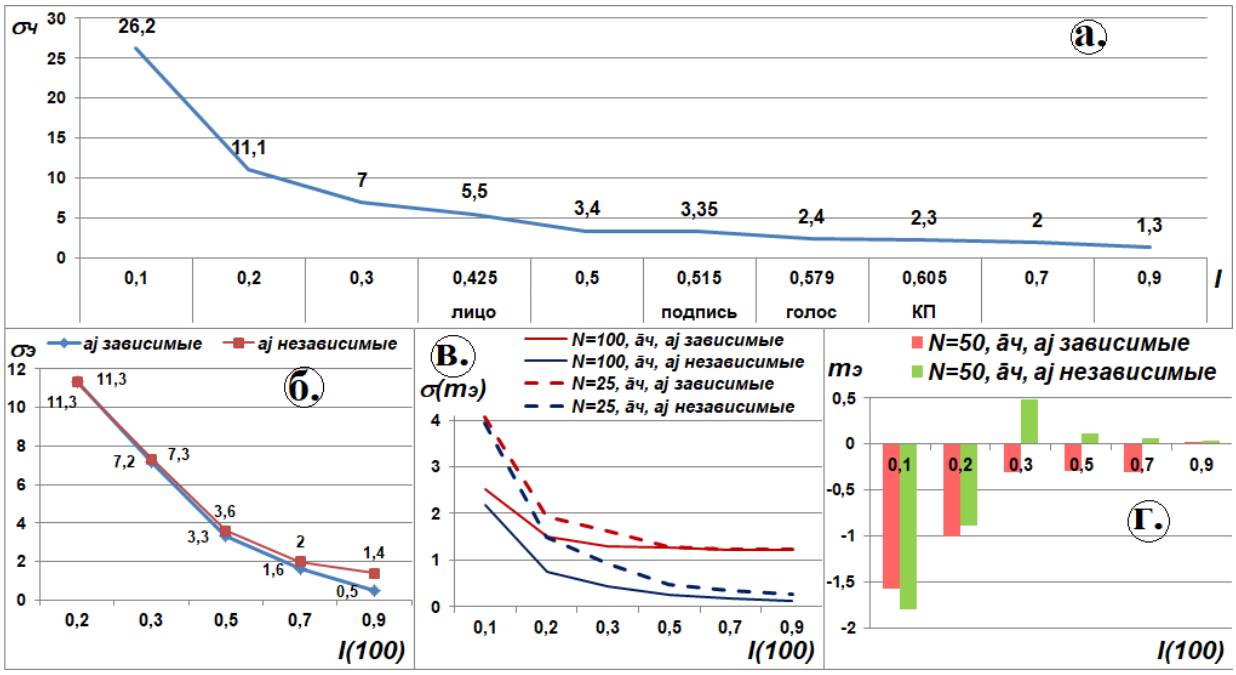


Рисунок 6 – Изменение следующих параметров эталонного (а) и эмпирического (б, в, г) распределений, порождаемых образами «Чужой», при снижении информативности признаков: а. предельного значения $\sigma_ч(N, K{=}64)$, б. среднего $\sigma_э(N, \bar{a}_ч)$, в. среднеквадратичного отклонения $m_э(N, \bar{a}_ч)$, г. среднего $m_э(N, \bar{a}_ч)$.

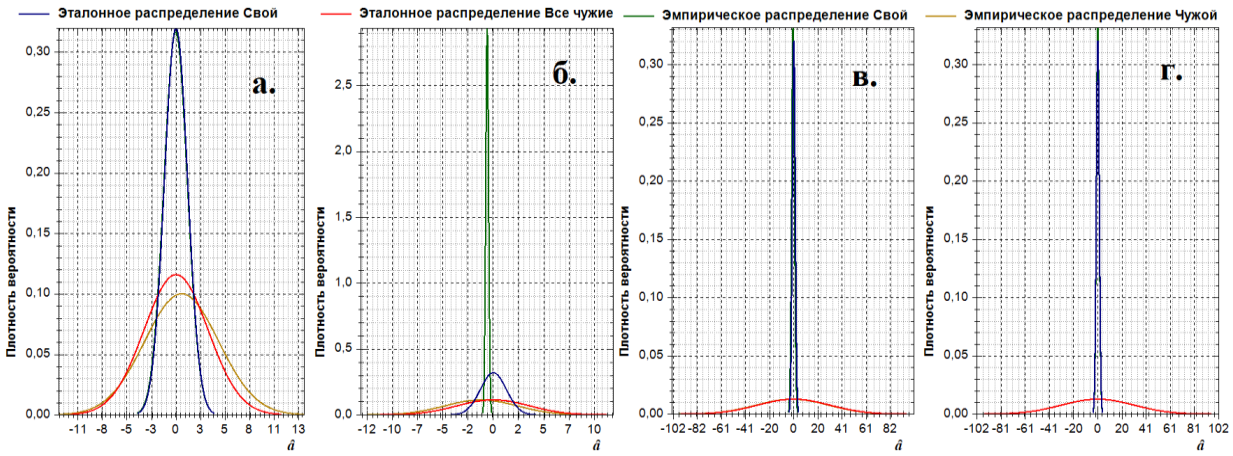


Рисунок 7 – Пример функций $p(\hat{a})$ и $\dot{p}(\hat{a})$: а. a_j независимые, б. a_j зависимы, в. a_j декоррелированы с помощью (16), г. функции $p(\hat{a})$ и $\dot{p}(\hat{a})$ центрированы ($m_c{=}m_ч{=}m_э{=}0$).

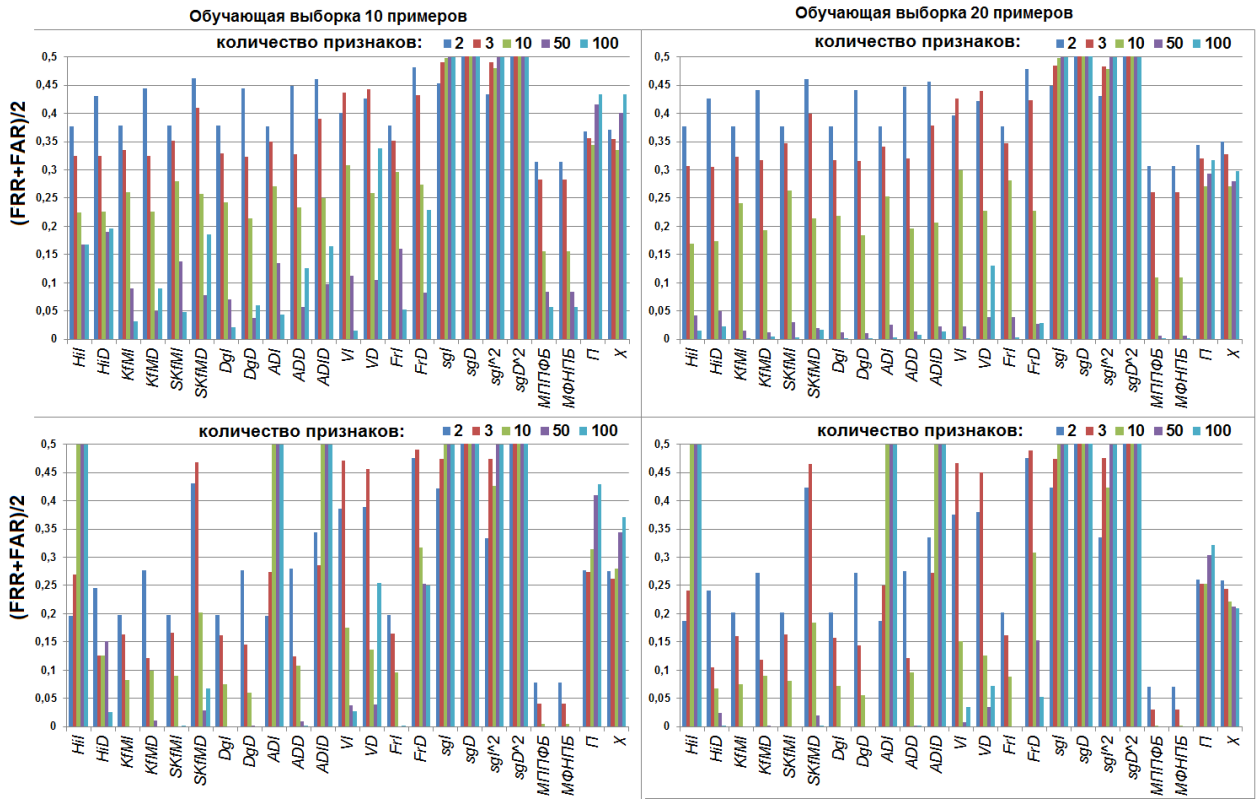


Рисунок 8 – Вероятности ошибок распознавания образов в пространстве независимых признаков

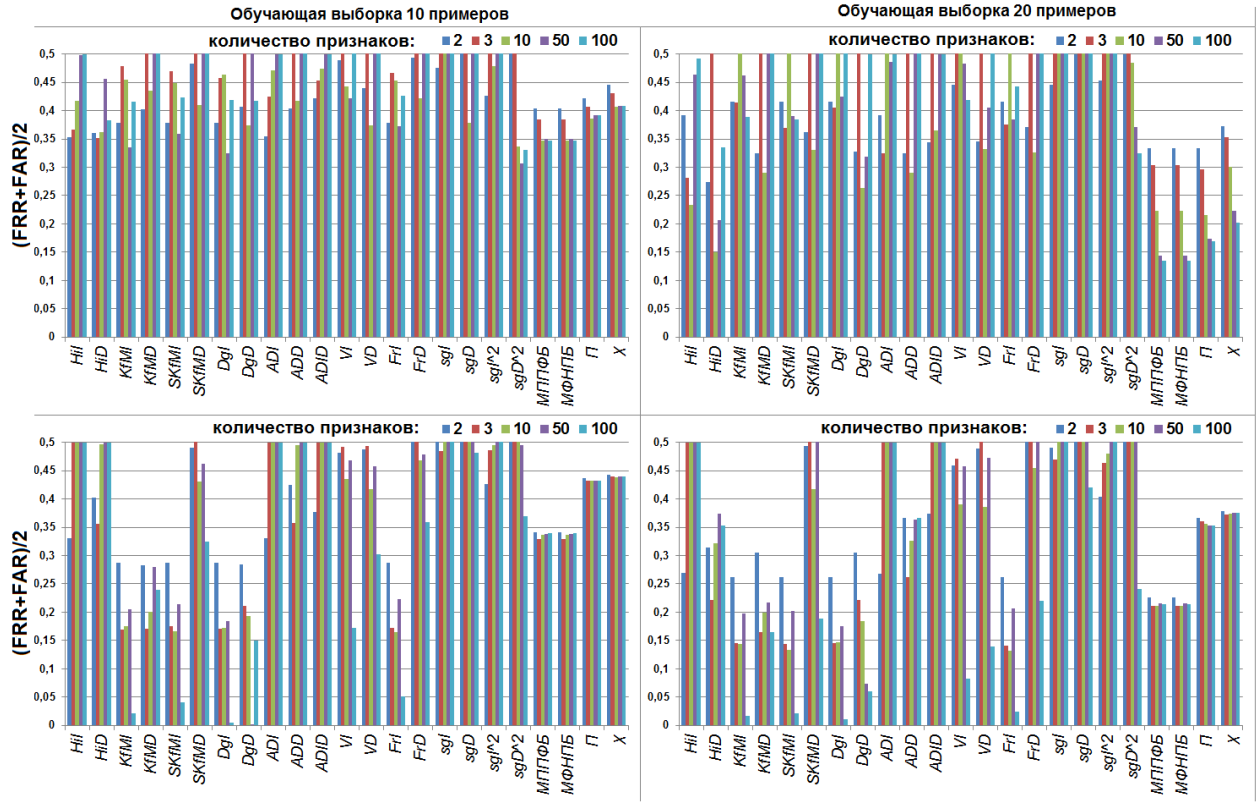


Рисунок 9 – Вероятности ошибок распознавания образов в пространстве зависимых признаков

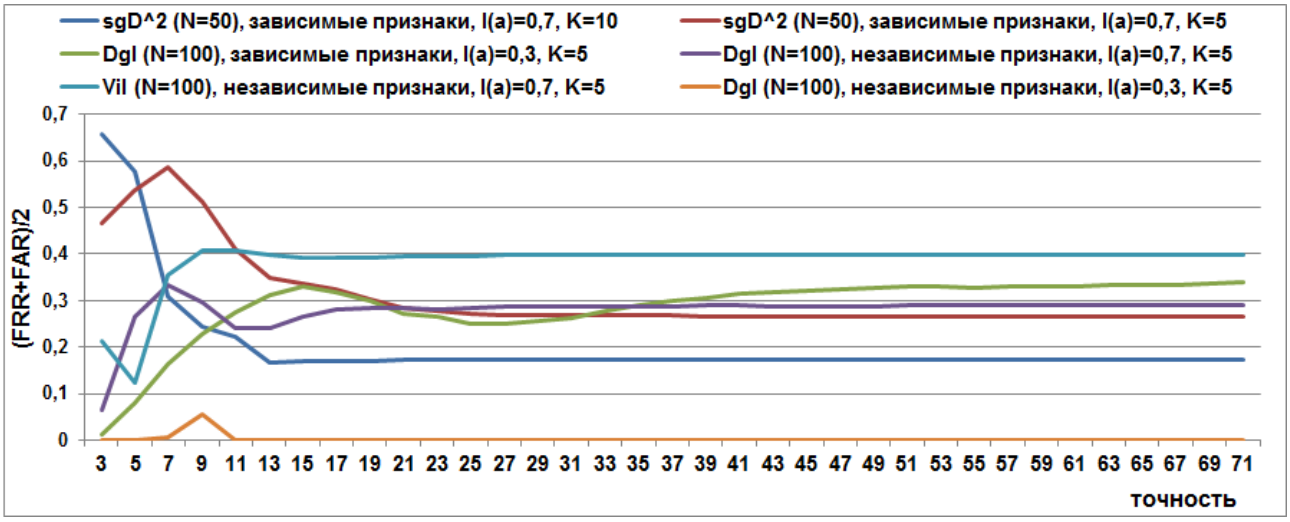


Рисунок 10 – Вероятности ошибок распознавания образов некоторыми критериями в зависимости от точности

Многослойные нейронные сети в ряде задач являются избыточными и долго обучаются (на большом числе примеров). В задачах биометрической аутентификации необходимо использовать «широкие» сети с малым числом слоев для быстрого обучения. Проведен поиск перспективных функционалов для построения таких нейронов. Каждый из них способен эффективно формировать решения при верификации образов в определенных условиях: в пространстве информативных и малоинформативных, зависимых и условно независимых (с коэффициентом парной корреляции менее 0,5) признаков, при малом (20 примеров) и сверхмалом (5-10 примеров) числе обучающих примеров образа «Свой».

В процессе работы были получены интересные свойства некоторых функционалов. В пространстве малоинформативных и сильно зависимых признаков наилучшие результаты показывает функционал на основе критерия квадрата среднего геометрического плотности вероятности сравниваемых распределений. В пространстве условно независимых признаков хорошие результаты показывают многие функционалы: на базе формулы гипотез Байеса, обратного критерия среднего геометрического плотности вероятности, интегрального и дифференциального критерия Джини и другие. Некоторые из них дают в существенной степени коррелированные результаты (например,

интегральные критерии Смирнова-Крамера-фон Мизеса и Фроцини). Некоторые лучше работают в пространстве малоинформативных признаков (например, интегральный критерий Андерсона-Дарлинга), другие – в пространстве информативных признаков (дифференциальный критерий Андерсона-Дарлинга). МППФБ лучше работает, если признаки информативны и независимы. Однако, если признаки сильно коррелированы, то данный метод совершает меньше ошибок, когда их информативность низкая (а не высокая, как ожидалось).

Распознавания и комплексирование биометрических образов на основе сетей квадратичных и классических нейронов.

Рассматриваются следующие независимые группы признаков:

1. Физиологические особенности лица.
2. Характеристики голоса диктора.
3. Особенности воспроизведения и внешнего вида подписей.
4. Клавиатурный почерк.

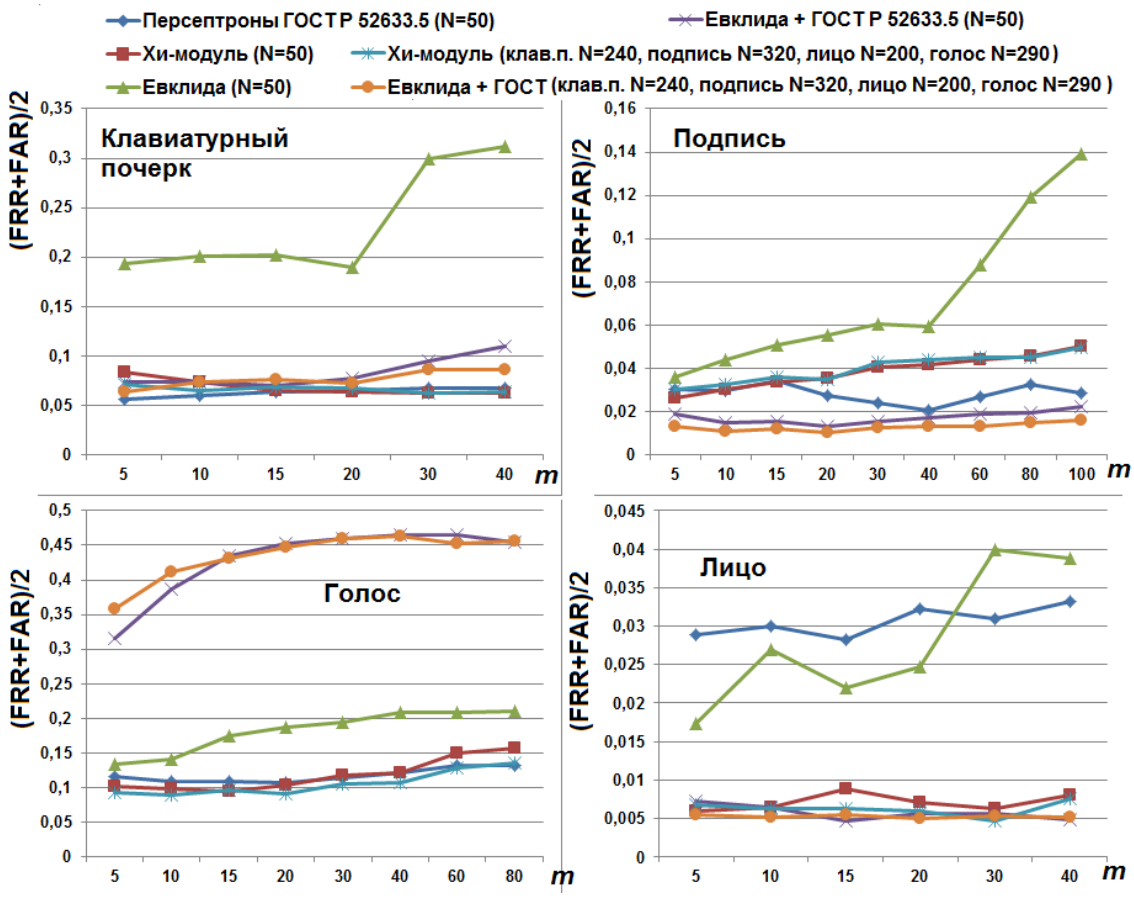


Рисунок 11 – Результаты распознавания субъектов однофакторными системами

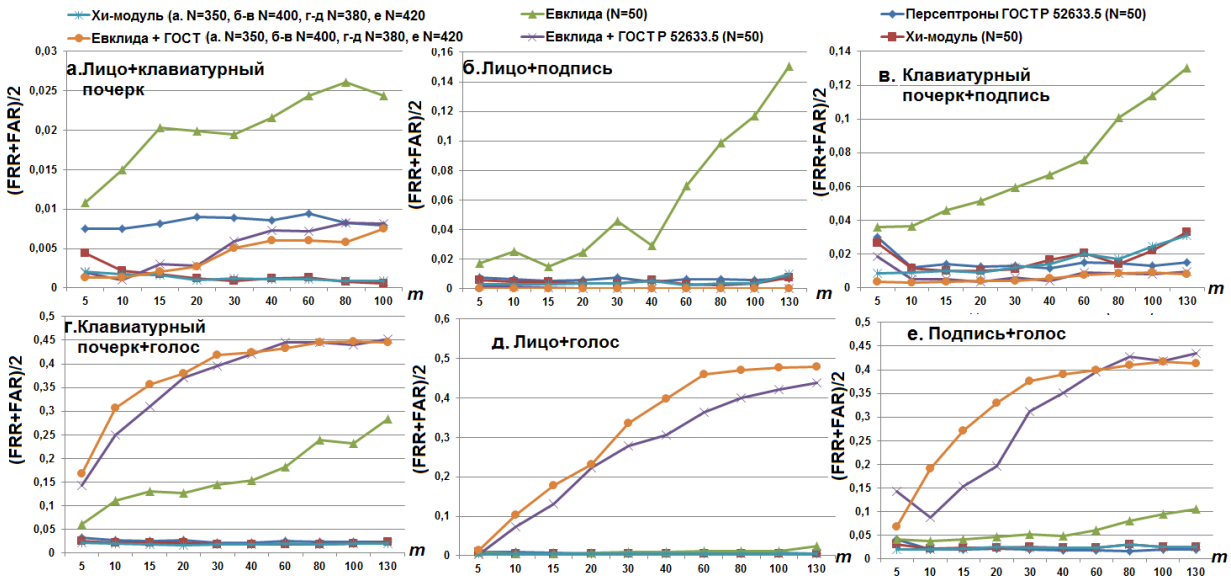


Рисунок 12 – Результаты распознавания субъектов двухфакторными системами

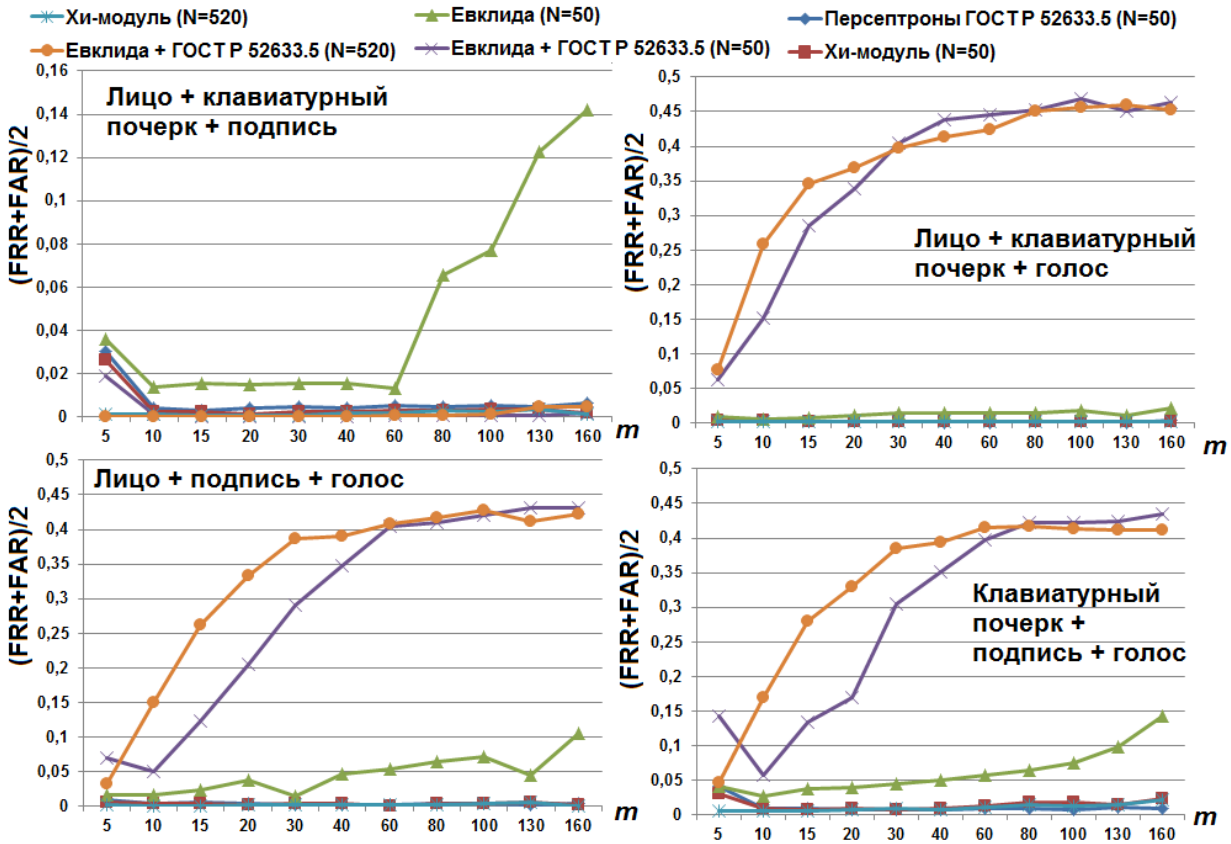


Рисунок 13 – Результаты распознавания субъектов трехфакторными системами

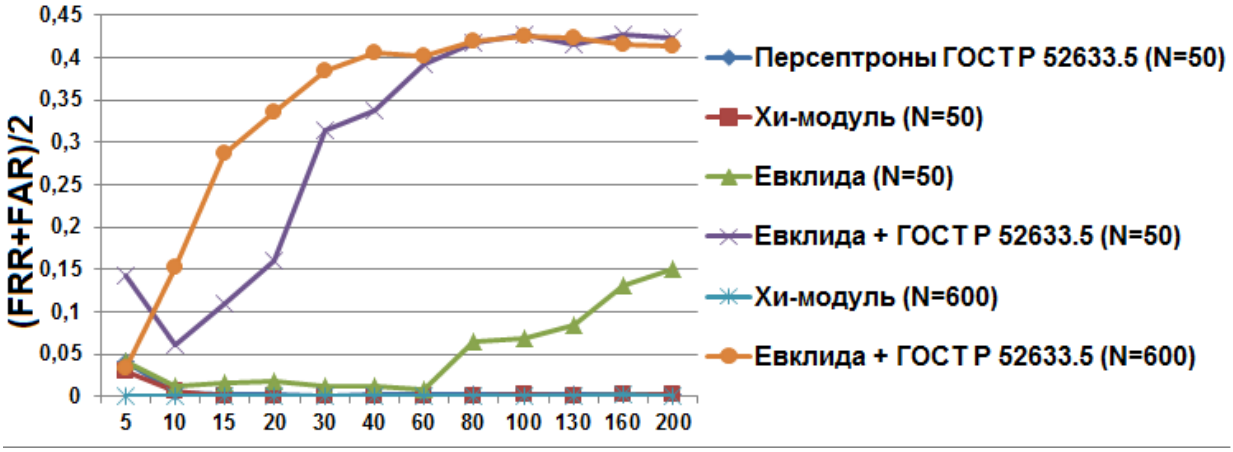


Рисунок 14 – Результаты распознавания субъектов четырехфакторной системой

Исследование свойств гравитационных мер близости и их сетей.

Предложены следующие меры близости:

$$G^2_{trig} = \frac{\sum_{j=2}^{N}\left(\sum_{t=1}^{j-1}\left(trig(j,t) = \begin{cases}\cos(\bar{a},\bar{\upsilon}_{j,t})\cdot r_{j,t}, if\ r_{j,t} > 0\\ -\sin(\bar{a},\bar{\upsilon}_{j,t})\cdot r_{j,t}, if\ r_{j,t} < 0\end{cases}\right)^2\right)}{\Pi(N) = \sqrt{\sum_{j=1}^{N}\frac{(m_j - a_j)^2}{\sigma_j^{\,2}}}},$$

$$G^2_1 = \frac{\sum_{j=2}^{N}\left(\sum_{t=1}^{j-1}\cos(\bar{a},\bar{\upsilon}_{j,t})\right)}{\Pi(N) = \sqrt{\sum_{j=1}^{N}\frac{(m_j - a_j)^2}{\sigma_j^{\,2}}}}, r_{j,t} > 0$$

$$G^2_2 = \frac{\sum_{j=2}^{N}\left(\sum_{t=1}^{j-1}\cos(\bar{a},\bar{\upsilon}_{j,t})^2\right)}{\Pi(N) = \sqrt{\sum_{j=1}^{N}\frac{(m_j - a_j)^2}{\sigma_j^{\,2}}}}, r_{j,t} > 0$$

$$G^2_3 = \frac{\sum_{j=2}^{N}\left(\sum_{t=1}^{j-1}(1+\cos(\bar{a},\bar{\upsilon}_{j,t}))\cdot(1-\sin(\bar{a},\bar{\upsilon}_{j,t}))\right)}{\Pi(N) = \sqrt{\sum_{j=1}^{N}\frac{(m_j - a_j)^2}{\sigma_j^{\,2}}}}$$

$$G^2_4 = \frac{\sum_{j=2}^{N}\left(\sum_{t=1}^{j-1}|\cos(\bar{a},\bar{\upsilon}_{j,t})|\cdot|\sin(\bar{a},\bar{\upsilon}_{j,t})|\right)}{\Pi(N) = \sqrt{\sum_{j=1}^{N}\frac{(m_j - a_j)^2}{\sigma_j^{\,2}}}}$$

$$G^2_5 = \frac{\sum_{j=2}^{N}\left(\sum_{t=1}^{j-1}\left(|\cos(\bar{a},\bar{\upsilon}_{j,t})|\cdot|\sin(\bar{a},\bar{\upsilon}_{j,t})|\right)^2\right)}{\Pi(N) = \sqrt{\sum_{j=1}^{N}\frac{(m_j - a_j)^2}{\sigma_j^{\,2}}}}$$

где $\bar{a}$ – вектор значений признаков, $\bar{\upsilon}_{j,t}$ – вектор с координатами под номерами j и t, равными единице, и остальными координатами, равными нулю, $r_{j,t}$ – коэффициент корреляции между признаками под номерами j и t, $trig(j, t)$ – функционал оценки силы притяжения эталона в 2-х пространственных

измерениях (порядок G равен 2). Физический смысл функционала $trig(j, t)$ заключается в том, чтобы тем больше усиливать притяжение, чем ближе местоположение распознаваемого образа к «линии сжатия» пространства. Фактически данный функционал учитывает дополнительную информацию о положении образа и корреляции между признаками. Суммарная сила притяжения равна отношению суммы квадратов составляющих силы во всех двумерных подпространствах исходного пространства признаков к значению меры Пирсона. Мера Пирсона играет роль расстояния в нормированной системе координат.

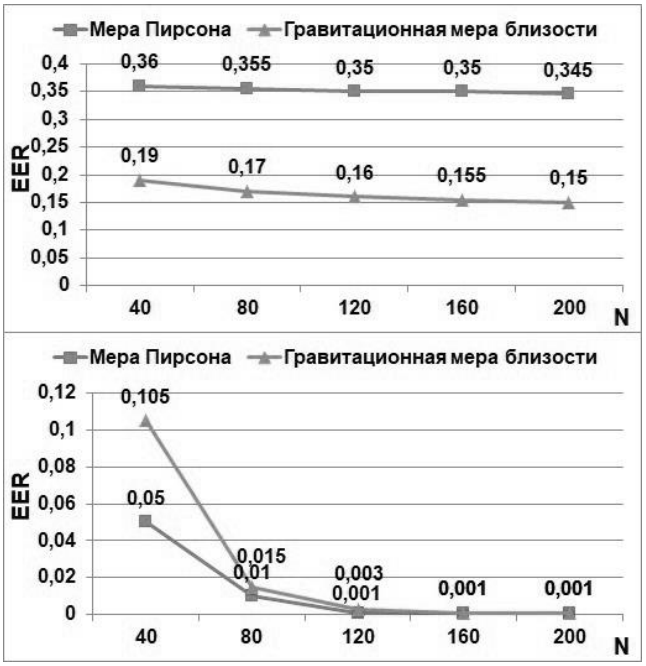


Рисунок 15 – Вероятность ошибок распознавания образов в пространстве зависимых (вверху) и независимых (внизу) малоинформативных признаков (при $I \approx 0{,}7$)

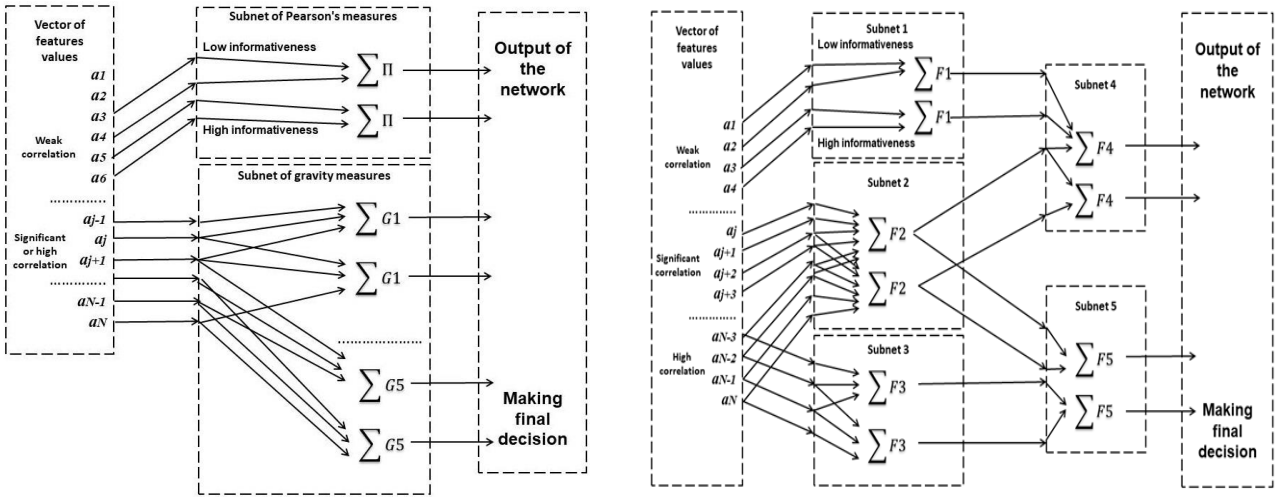


Рисунок 16 – Схема однослойной гибридной «широкой» сети на базе меры Пирсона и гравитационных метрик (слева), схема гибридной двухслойной «широкой» сети

Верификация динамических биометрических образов на основе гибридных нейросетевых моделей. Разработка клавиатуры для анализа клавиатурного почерка

Для оценки информативности признаков и надежности распознавания субъектов с использованием гибких нейронных сетей в течение нескольких месяцев формировалась база биометрических образов. Данная база включает:

- более 12000 тайных и открытых рукописных образов 161 испытуемого, включая образы индивидуального пароля, фиксированных слов («безопасность», «операция») и автографов;
- более 9000 тайных и открытых образов клавиатурного почерка 94 испытуемых;
- более 14000 тайных и открытых голосовых образов 102 испытуемых, включая образы индивидуального пароля и фиксированных фраз («идентифицируйте меня», «разграничение доступа», «разрешите доступ»);

Разработан опытный образец клавиатуры, позволяющий регистрировать базовые (временные характеристики нажатия клавиш) и дополнительные признаки клавиатурного почерка (давление на клавиши и параметры вибрации клавиатуры при наборе парольной фразы), а также производить идентификацию пользователей с использованием специально разработанного программного обеспечения. Она состоит из следующих компонент:

- стандартная клавиатура;
- 5 резистивных датчиков давления для измерения силы нажатия на клавиши;
- пьезоэлектрический датчик для преобразования механических вибраций клавиатуры в электрический сигнал;
- два времяпролетных лазерных дальномера, фиксирующих динамические характеристики кистей правой и левой рук;
- модуль для подключения USB-клавиатуры к микроконтроллеру;

– микроконтроллер Arduino.

Клавиатура регистрирует 5 функций ($\Delta T(l)$, $force(t)$, $vibro(t)$, $d_L(t)$, $d_R(t)$): паузы между нажатием клавиш, времена удержания клавиш, силу нажатия на клавиши, вибрацию, расстояния от левой и правой руки до клавиатуры. Частота дискретизации этих функций составляет 40Гц, т.е. Δt=0,025 секунд.

Рисунок 17 – Изменение расстояний рук испытуемых до клавиатуры в процессе набора парольных фраз

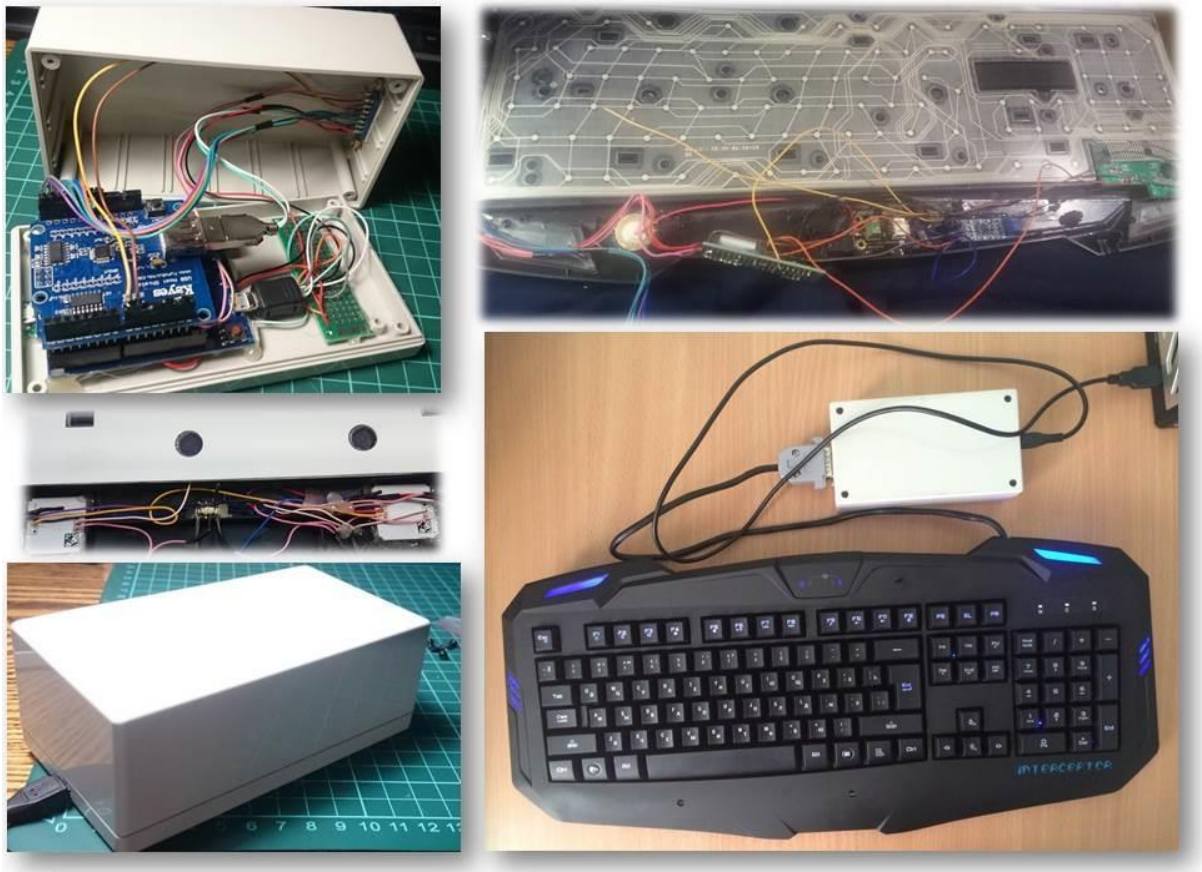


Рисунок 18 – Разработанная клавиатура и ее комплектующие

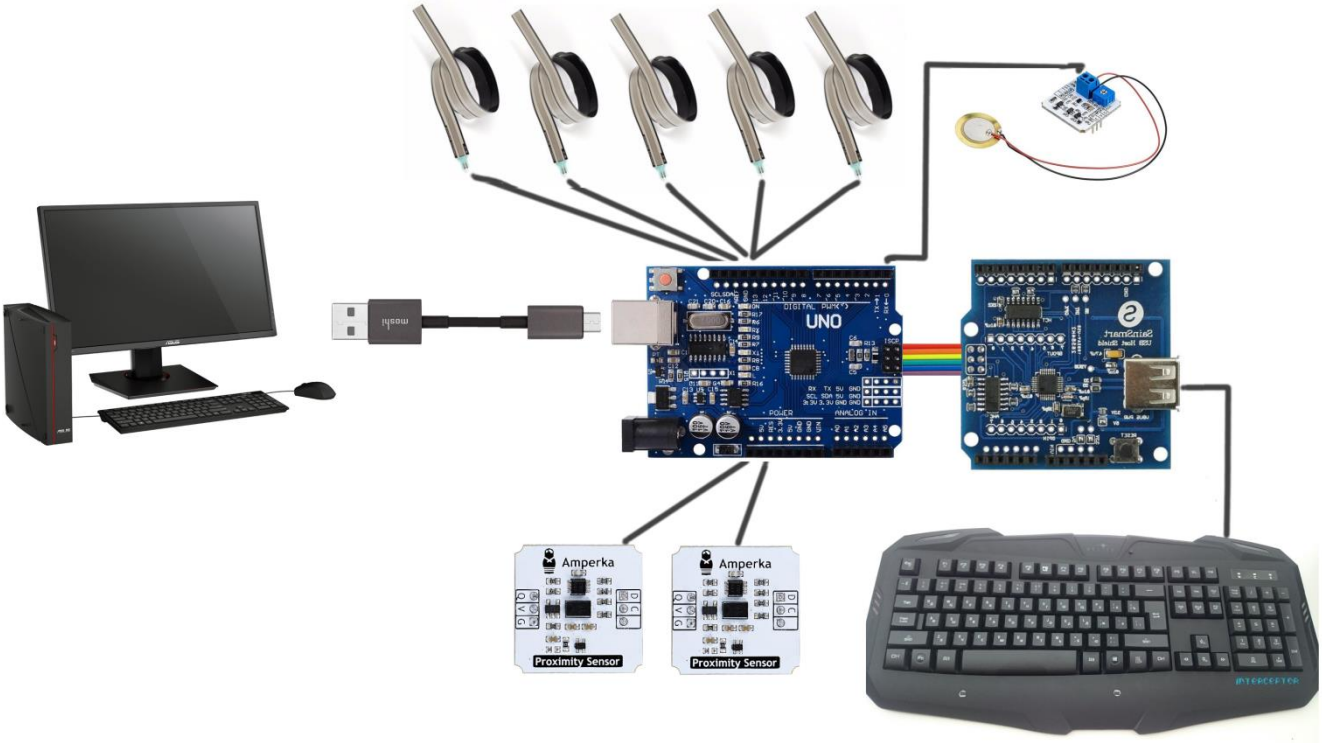


Рисунок 19 – Схема подключения модулей, из которых состоит разработанная клавиатура

Таблица 4. Информативность других признаков биометрических образов (тайных – т, открытых – о, подписей – п)

Описание признаков	Т, *AUC*	О, *AUC*	П, *AUC*	Число признаков
Расстояния в двухмерном пространстве (*x_coord* (*t*), *y_coord* (*t*))	0,468	0,411	0,451	120
Расстояния в трехмерном пространстве (*x_coord* (*t*), *y_coord* (*t*), *pressure*(*t*))	0,474	0,514	0,541	120
коэффициенты корреляции между *x_coord*(*t*), *y_coord* (*t*), *pressure*(*t*), *x_coord* '(*t*), *y_coord* '(*t*), *pressure*'(*t*), $v_{xy}(t)$	0,43	0,502	0,452	21
параметры изображения рукописного образа	0,372	0,399	0,367	5
средние значения участков *pressure*(*t*)	0,369	0,428	0,435	10
средние значения участков *x_coord* '(t)	0,58	0,564	0,405	10
средние значения участков *y_coord* '(*t*)	0,559	0,66	0,449	10
средние значения участков $v_{xy}(t)$	0,47	0,389	0,349	10
средние значения участков $v_{xyp}(t)$	0,375	0,434	0,449	10
интегралы участков *pressure*(*t*)	0,398	0,47	0,348	10
интегралы участков *x_coord* '(t)	0,583	0,559	0,449	10
интегралы участков *y_coord* '(*t*)	0,571	0,658	0,492	10
интегралы участков $v_{xy}(t)$	0,471	0,379	0,354	10
интегралы участков $v_{xyp}(t)$	0,357	0,423	0,445	10
коэффициенты корреляции между $\Delta T(l)$, *force*(*t*), *vibro*(*t*), $d_L(t)$, $d_R(t)$, $\Delta T'(l)$, *force*'(*t*), *vibro*'(*t*), $d_L'(t)$,$d_R'(t)$	0,635	0,657		45
средние значения участков $\Delta T(l)$	0,754	0,767		15
средние значения участков *force*(*t*)	0,814	0,82		15
средние значения участков *vibro*(*t*)	0,731	0,753		15
средние значения участков $d_L(t)$	0,692	0,71		15
средние значения участков $d_R(t)$	0,766	0,783		15
средние амплитуды гармоник, принадлежащих одной частоте, полученных после оконного преобразования Фурье над *u(t)*	0,38	0,45		512
частоты переходов *u(t)* через нуль	0,41	0,55		32

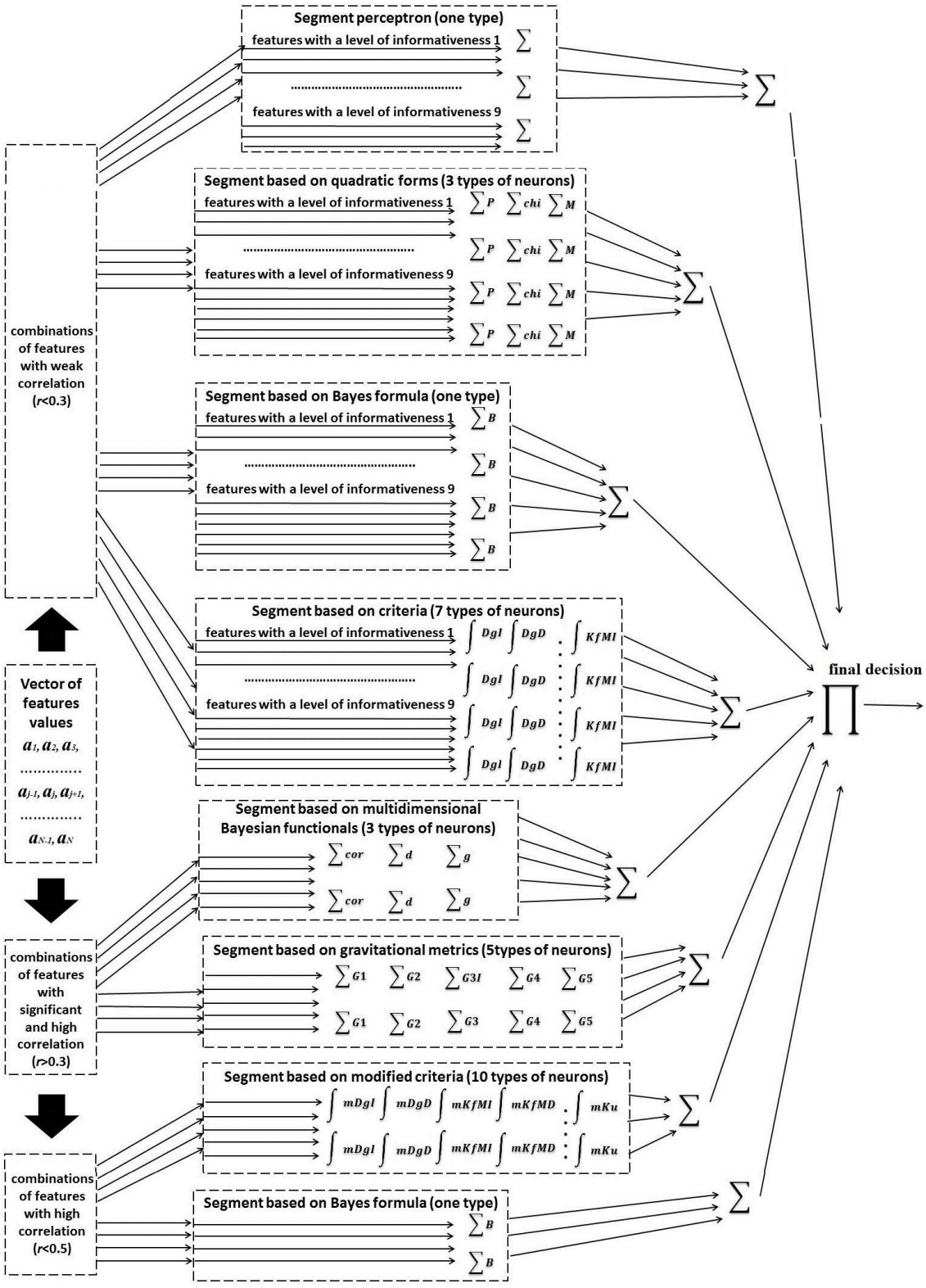


Рисунок 20 – Схема подключения модулей, из которых состоит разработанная клавиатура

Таблица 5. Результаты распознавания испытуемых по динамическим биометрическим образам (тайным – т, открытым – о, подписям – п)

Вид и тип образов		EER	FRR	FAR	Объем обучающей выборки «Свой»	Прошло дней после обучения
рукописные	Т	0,02	0,23	<0,001	35	1-3
		0,039	0,28	<0,001	20	7-42
	О	0,038	0,44	<0,001	35	1-3
		0,067	0,66	<0,001	20	7-42
	П	0,009	0,064	<0,001	35	1-3
		0,015	0,106	<0,001	20	7-42
клавиатурный почерк	Т	0,03	0,44	<0,001	35	1-3
		0,064	0,45	<0,001	20	7-42
	О	0,037	0,65	<0,001	35	1-3
		0,055	0,66	<0,001	20	7-42
голос	Т	0,019	0,12	<0,001	35	1-3
		0,054	0,21	<0,001	20	7-42
	О	0,036	0,71	<0,001	35	1-3
		0,07	0,72	<0,001	20	7-42

Приложение 2 Акты внедрения результатов работы

УТВЕРЖДАЮ
Проректор по научной и инновационной деятельности ОмГТУ, к.х.н.
_____ В.Ф. Фефелов
04 _____ 2022 г.

АКТ

об использовании результатов докторской диссертации доцента кафедры «Комплексная защита информации» Омского государственного технического университета Сулавко Алексея Евгеньевича

Комиссия в составе:
начальник управления научной деятельностью, к.т.н. Пономарев Е.В.
начальник отдела инновационной деятельности, к.т.н. Федоров А.А.

составили настоящий акт о том, что результаты докторской диссертации Сулавко А.Е. использованы при разработке национального стандарта ГОСТ Р "Искусственный интеллект. Нейросетевые алгоритмы в защищённом исполнении. Автоматическое обучение нейросетевых моделей на малых выборках в задачах классификации". Проект стандарта в первой редакции разработан ОмГТУ под руководством Сулавко А.Е. в рамках НИР «Защищенный режим исполнения искусственного интеллекта на базе автоматически обучаемых сетей автокорреляционных нейронов», регистрационный номер: 121033100052-6.

При разработке первой редакции стандарта были использованы следующие основные результаты докторской диссертации Сулавко А.Е.:

1. Концепция защищенного исполнения процедур классификации образов для реализации функции отчуждения искусственного интеллекта от принятия решений.
2. Модель корреляционных нейронов и модель нейросетевого преобразователя образов в код на их основе, а также алгоритм их автоматического синтеза и обучения на малых выборках данных.
3. Технология автоматического синтеза и обучения нейросетевых моделей доверенного искусственного интеллекта.

Стандарт прошел экспертизу технических комитетов Росстандарта и включен в программу стандартизации технического комитета «Искусственный интеллект» (ТК164). На данный момент ОмГТУ входит в подкомитет «Данные» (ПК02), относящийся к ТК164, а Сулавко А.Е. является представителем ОмГТУ в ПК02 и ТК164, а также экспертом от России без права голоса в Международном техническом комитете ISO/IEC JTC 1/SC 42 "Artificial intelligence".

Начальник управления
научной деятельностью ОмГТУ, к.т.н. _____ Пономарев Е.В.

Начальник отдела инновационной
деятельности ОмГТУ, к.т.н. _____ Федоров А.А.

УТВЕРЖДАЮ:
Директор ООО «СИБ»
_____ А.А. Помешкин

«18» апреля 2022 г.

АКТ

об использовании результатов
докторской диссертационной работы
Сулавко Алексея Евгеньевича

Комиссия в составе: председателя – начальника отдела технической защиты информации Общества с ограниченной ответственностью «Системы информационной безопасности» (далее - ООО «СИБ») Мамаева Евгения Сергеевича, членов комиссии: Реутова Владимира Владимировича – начальника коммерческого отдела ООО «СИБ» и Коротких Игоря Валерьевича – начальника отдела мониторинга информационных систем ООО «СИБ», составили настоящий Акт о том, что результаты диссертационной работы Сулавко А.Е., представленной на соискание ученой степени доктора технических наук, использованы в проектно-конструкторской деятельности ООО «СИБ» при проектировании платформы информационного обмена нового поколения с функционалом по обеспечению и поддержанию заданного уровня доверия в недоверенной среде в частности:

1. Концепции защищенного исполнения процедур высоконадежной биометрической аутентификации от исследования и компрометации знаний.
2. Модели корреляционных нейронов и модели нейросетевого преобразователя биометрия-код на их основе, а также алгоритмов их автоматического синтеза и обучения на малых выборках данных.
3. Методов и алгоритмов высоконадежной биометрической аутентификации в защищенном режиме исполнения с обеспечением защиты биометрических данных от компрометации.

Использование указанных технических предложений позволяет обеспечить возможность создания новых систем искусственного интеллекта, в частности, систем высоконадежной биометрической аутентификации для контроля доступа, обладающих повышенным уровнем устойчивости к состязательным атакам, а также высоким уровнем защищенности знаний искусственного интеллекта, в том числе, биометрических данных пользователей от компрометации.

Председатель комиссии _____ Е.С. Мамаев

Члены комиссии: _____ В.В. Реутов

_____ И.В. Коротких

18.04.2022

open{ code: }
открытый код

Общество с ограниченной ответственностью "Открытый код"
Россия, 443001, г. Самара, ул. Ульяновская, 52/55, этаж 15, ком.14
Тел./факс: (846) 331-11-11, 331-21-01(02,03,04)
E-mail: info@o-code.ru | www.o-code.ru
ОГРН 1036300222100 | ИНН 6313007301 | КПП 631501001

Исх. № 190/01 / 20.04.22

УТВЕРЖДАЮ:
Исполнительный директор
ООО «Открытый код»
_____ П.В. Ситников
«20» апреля 2022 г.

АКТ
об использовании результатов докторской диссертации
Сулавко Алексея Евгеньевича

Комиссия в составе: председателя НТС ООО «Открытый код», д.т.н., профессора Иващенко Антона Владимировича, заместителя директора по управлению проектами, к.т.н., доцента Головнина Олега Константиновича, руководителя проектов, к.т.н. В.В. Авсиевича составила настоящий акт о том, что результаты докторской диссертационной работы Сулавко А.Е. использованы в научно-исследовательской, опытно-конструкторской и проектной деятельности ООО «Открытый код» при разработке программно-аппаратных комплексов для технической защиты конфиденциальной информации. В частности, были использованы следующие технические предложения: концепция защищенного исполнения процедур высоконадежной биометрической аутентификации от исследования и компрометации знаний; модель корреляционных нейронов, модель нейросетевого преобразователя биометрия-код на основе корреляционных нейронов, алгоритмы автоматического синтеза и обучения на малых выборках данных нейросетевых преобразователей биометрия-код на основе корреляционных нейронов; методы высоконадежной биометрической аутентификации по рукописным и голосовым паролям в защищенном режиме исполнения с обеспечением защиты биометрических данных от компрометации.

Использование обозначенных технических решений позволило спроектировать и разработать функционал для многофакторной биометрической аутентификации при доступе в закрытые зоны с частотой ошибок «ложного отказа» в доступе менее 6% и частотой ошибок «ложного допуска» менее 0,01% по результатам предварительного тестирования, а также обеспечить защиту биометрических данных пользователей от компрометации.

Председатель комиссии А.В. Иващенко

Члены комиссии: О.К. Головнин

В.В. Авсиевич

KRAFT lab

Группа компаний Крафт Лаб
109052, Москва, Смирновская 25, Бизнес-центр "Смирновский"
e-mail: info@kraftlab.ru, https://www.kraftlab.ru/
телефон: +7 (495) 120-80-29
ОГРН 1167746840150, ИНН/КПП 7722374819/772201001, ОКПО 04443680

Исх. № 005/04-22 от 20.04.2022 г.

УТВЕРЖДАЮ:
Генеральный директор
ООО "КРАФТ ЛАБ"
М.А. Кондрашова
«20» апреля 2022 г.

АКТ

об использовании результатов докторской диссертации
Сулавко Алексея Евгеньевича

Комиссия в составе:

председатель: генеральный директор ООО "КРАФТ ЛАБ" М.А. Кондрашова

члены комиссии: технический директор Л.В. Плетнев
директор департамента А.А. Клиновенко

составили настоящий акт о том, что следующие результаты диссертационной работы, представленной на соискание ученой степени доктора технических наук, использованы в научно-исследовательской и опытно-конструкторской деятельности ООО "КРАФТ ЛАБ":

1. Концепции защищенного исполнения алгоритмов классификации образов, которая позволяет сформировать устойчивость модели к извлечению знаний.
2. Модели корреляционных нейронов, анализирующих корреляционные связи между признаками вместо признаков, а также робастной алгоритм автоматического синтеза и обучения сетей корреляционных нейронов на малых выборках данных.
3. Технологии автоматического синтеза моделей доверенного искусственного интеллекта, обладающих повышенной устойчивостью к состязательным атакам и исполняемых в защищенном от исследования режиме, а также программного модуля на ее основе.

Это позволит повысить защищенность от угроз информационной безопасности и надежность разрабатываемых программных модулей ИТ-систем и средств обработки данных, основанных на алгоритмах машинного обучения и искусственном интеллекте.

Председатель комиссии М.А. Кондрашова

Члены комиссии: Л.В. Плетнев

А.А. Клиновенко

Общество с ограниченной ответственностью
«АИ ЗИОН» (ООО «АИ ЗИОН»)
Телефон: +7 (953) 394-90-54
E-mail: aiconstructor@mail.ru
ОКПО 75754649, ОГРН 1215500030976
ИНН 5507286620, КПП 550701001

АКТ

об использовании результатов докторской диссертации
Сулавко Алексея Евгеньевича

Комиссия в составе: ведущего инженера-исследователя по направлению ИИ, к.т.н. Жумажановой С.С., инженера-аналитика по направлению ИИ, к.т.н. Самотуги А.Е., исследователя-разработчика по направлению ИИ Чобана А.Г. составили настоящий акт о том, что результаты докторской диссертационной работы Сулавко А.Е. легли в основу программного продукта AIC ModelOps Platform, разрабатываемого в рамках проектно-конструкторской деятельности ООО «АИ ЗИОН».

Результаты внедрялись при выполнении НИОКР по теме: Разработка и тестирование опытного образца платформы для оптимизации процесса цифровой трансформации при создании и внедрении доверенного искусственного интеллекта с использованием сетей корреляционных нейронов, регистрационный номер НИОКР 122031400041-2.

Проект по разработке AIC ModelOps Platform получил поддержку Фонда содействия развитию малых форм предприятий в научно-технической сфере (Договор 83ГС1ИИС12-D7/72189от 18.02.2022, заявка С1ИИ-113719).

Результаты докторской диссертации использованы в следующем виде:

1. Концепция защищенного исполнения процедур классификации образов для реализации функции отчуждения искусственного интеллекта от принятия решений.
2. Модель корреляционных нейронов и модель нейросетевого преобразователя образов в код на их основе, а также алгоритм их автоматического синтеза и обучения на малых выборках данных.
3. Адаптивная нейро-иммунная модель искусственного интеллекта и алгоритмы ее обучения с учителем и с подкреплением, позволяющие предупредить или снизить влияние концептуального дрейфа.
4. Технология автоматического синтеза и обучения нейросетевых моделей доверенного искусственного интеллекта, а также программные модули, реализующие данную технологию.

Главный инженер-исследователь
по направлению ИИ, к.т.н. Жумажанова С.С.

Инженер-аналитик
по направлению ИИ, к.т.н. Самотуга А.Е.

Исследователь-разработчик
по направлению ИИ Чобан А.Г.

23.04.2022

МСЧ 4
Центр Здоровья

Бюджетное учреждение здравоохранения
Омской области «Медико-санитарная часть № 4» (МСЧ 4)
644039, г. Омск, Воровского, 62/1
e-mail: msch-4@mail.ru, https://msch4omsk.ru/pages/tsentr-zdorovia тел.: 8 (3812) 46–96–20
ОГРН 1025501176140, ИНН/ КПП 5505019620/550501001, ОКПО 01937424

УТВЕРЖДАЮ:
Главный врач
БУЗОО "МСЧ №4"
Филатов Ю.В.
«29» апреля 2022 г.

АКТ

об использовании результатов докторской диссертационной работы
Сулавко Алексея Евгеньевича

Комиссия в составе:

председатель: главный врач БУЗОО "МСЧ №4" Филатов Ю.В.

члены комиссии: заместитель главного врача по поликлиническому разделу работы Калушина Т.Л.
заведующий центром здоровья-врач-терапевт Канаев С.А.

составили настоящий акт о том, что следующие результаты диссертационной работы, представленной на соискание ученой степени доктора технических наук, использованы в рамках инструментально-лабораторных обследований и тестирования пациентов на аппаратно-программном комплексе в Центре здоровья поликлиники БУЗОО МСЧ№4:

1. Модели корреляционных нейронов, анализирующих корреляционные связи между признаками вместо признаков, а также робастного алгоритма автоматического синтеза и обучения сетей корреляционных нейронов на малых выборках данных.
2. Адаптивной нейро-иммунной модели искусственного интеллекта и алгоритмов ее обучения на малых выборках данных.
3. Технологии автоматического синтеза и обучения нейросетевых моделей доверенного искусственного интеллекта.

На базе указанных технических предложений, а также с использованием алгоритмов кластеризации данных был разработан программный модуль для анализа результатов обследований пациентов в обезличенном виде, проводимых в Центре здоровья. Результаты обследований представляют собой данные, регистрируемые приборами Валеоскан, Кардиовизор, Медасс, а также рекомендации врача. Разработанный программный модуль позволяет:

1. Выявлять условные категории пациентов со схожими проблемами со здоровьем. На основании кластеризации результатов обследований были составлены типовые шаблоны рекомендаций для каждой условной категории пациентов.
2. Классифицировать пациентов, вновь проходящих обследование в Центре здоровья, по условным категориям и предоставлять врачу шаблон типовых рекомендаций, который врач по своему усмотрению может редактировать для каждого случая отдельно (добавлять, удалять, изменять рекомендации в зависимости от особенностей, показаний и истории болезни пациента). Это позволило оптимизировать работу врача, сократив время, затрачиваемое врачом на подготовку рекомендации пациенту (набор текста на компьютере, формулирование заключения), и соответственно общее время приема пациента.

Председатель комиссии Филатов Ю.В.

Члены комиссии: Калушина Т.Л.

Канаев С.А.

УТВЕРЖДАЮ
Проректор по образовательной деятельности
ФГАОУ ВО «Омский государственный
технический университет», к.и.н.
_____ А.С. Полынский
«24» апреля 2023 г.

АКТ
о внедрении в учебный процесс университета результатов докторской диссертации
доцента кафедры «Комплексная защита информации»
Сулавко Алексея Евгеньевича

Мы, нижеподписавшиеся, и.о. декана Радиотехнического факультета, к.т.н. Пузырёв П.И., заместитель заведующего кафедрой «Комплексная защита информации», д.ф-м.н. Магазев А.А. составили настоящий акт о том, что полученные доцентом кафедры «Комплексная защита информации» Сулавко А.Е. результаты докторской диссертации внедрены в учебный процесс университета.

Предложенные в диссертации модель, методы и алгоритмы, а также программный комплекс используются на кафедре «Комплексная защита информации» для подготовки бакалавров по направлению 10.03.01 – «Информационная безопасность», специалистов по направлению 10.05.05 – «Безопасность информационных технологий в правоохранительной сфере», а также магистрантов по направлению 09.04.01 – «Информатика и вычислительная техника» (профиль «Безопасность и этика искусственного интеллекта») при изучении дисциплин «Биометрия и защита информации», «Машинное обучение в приложениях биометрии», «Защищенное исполнение искусственного интеллекта», «Доверенный искусственный интеллект», «Распознавание образов». Кроме этого, результаты диссертационной работы Сулавко А.Е. активно используются в рамках выполнения выпускных работ и научно-исследовательских работ университета.

И.о. декана Радиотехнического факультета,
к.т.н. П.И. Пузырев

Зам. заведующего кафедрой,
профессор кафедры
«Комплексная защита информации»,
д.ф-м.н. А.А. Магазев

УТВЕРЖДАЮ
Проректор по учебной работе
Санкт-Петербургского государственного электротехнического университета «ЛЭТИ» им. В.И.Ульянова,
_____ к.т.н., доцент Галунин С.А.
«16» мая 2022 г.

АКТ

о внедрении в учебный процесс университета результатов докторской диссертации
доцента кафедры «Комплексная защита информации»
Омского государственного технического университета
Сулавко Алексея Евгеньевича

Мы, нижеподписавшиеся, проректор по учебной работе, к.т.н., доцент Галунин С.А., и руководитель программы подготовки магистратуры «Безопасность и этика искусственного интеллекта», к.ф.-м.н., доцент Левина А.Б., составили настоящий акт о том, что полученные Сулавко А.Е. результаты докторской диссертации внедрены в учебный процесс университета.

Предложенные в диссертации концепция защищенного исполнения процедур высоконадежной биометрической аутентификации, модели корреляционных нейронов и нейросетевого преобразователя биометрия-код, адаптивная нейро-иммунная модель искусственного интеллекта и алгоритмы их автоматического синтеза и обучения с учителем и с подкреплением на малых выборках данных, а также методы высоконадежной многофакторной аутентификации с обеспечением защиты биометрических данных от компрометации используются при разработке программы подготовки магистратуры СПбГЭТУ «ЛЭТИ» «Безопасность и этика искусственного интеллекта» в рамках направления 01.04.01 «Информатика и вычислительная техника». Предложенные в докторской диссертации Сулавко А.Е. положения легли в основу следующих дисциплин: «Доверенный искусственный интеллект», «Защищенное исполнение искусственного интеллекта», «Машинное обучение в приложениях биометрии».

Проректор по учебной работе, к.т.н., доцент С.А. Галунин

Руководитель программы подготовки магистратуры
«Безопасность и этика искусственного интеллекта»,
к.ф.-м.н., доцент А.Б. Левина

Приложение 3 Письмо в ТК 164 и список разработчиков стандарта

УДК 004.855.5:006.354 ОКС 35.240.01

Ключевые слова: автоматическое машинное обучение, защищенное исполнение алгоритмов классификации, нейросетевые модели, преобразователи образов в код, корреляционный нейрон, мета-признак Байеса-Минковского

Разработан коллективом сотрудников Федерального государственного бюджетного образовательного учреждения высшего образования «Омский государственный технический университет» (ОмГТУ).

Проректор по НИД ОмГТУ:	подпись	к.х.н., Фефелов В.Ф.
Руководитель разработки:		
доцент каф. комплексной защиты информации	подпись	к.т.н., Сулавко А.Е.
Исполнители:		
зав. каф. комплексной защиты информации	подпись	д.т.н., Ложников П.С.
доцент каф. комплексной защиты информации	подпись	к.т.н., Самотуга А.Е.
магистрант	подпись	Стадников Д.Г.
магистрант	подпись	Чобан А.Г.
аспирант	подпись	Иниватов Д.П.

Приложение 4 План проспект разработанного стандарта

1 Область применения
2 Нормативные ссылки
3 Термины и определения
4 Обозначения и сокращения
5 Угрозы информационной безопасности, возникающие при исполнении искусственного интеллекта в задачах классификации
6 Базовый подход к защите искусственного интеллекта от обозначенных угроз информационной безопасности в задачах классификации
7 Общие положения автоматического синтеза и обучения нейро-корреляционных преобразователей в задачах классификации
    7.1 Архитектурные принципы построения систем защищенного исполнения нейросетевых алгоритмов классификации образов. Связывание класса образов с криптографическим ключом или паролем
    7.2 Классификация корреляционных нейронов по принципу функционирования и уровню защиты данных от компрометации
    7.3 Требования к законам распределения признаков
    7.4 Требования к обучающим данным и контроль независимости тренировочных примеров
8 Синтез и обучение нейро-корреляционных преобразователей для защищенного исполнения процедур бинарной классификации образов
    8.1 Настройка первого слоя преобразований
    8.2 Математическая модель разностного корреляционного нейрона
    8.3 Автоматический синтез и обучение разностного корреляционного нейрона
    8.4 Определение количества и параметров корреляционных нейронов второго слоя
    8.5 Автоматический синтез и обучение корреляционных нейронов второго слоя
9 Использование нейро-корреляционных преобразователей для защищенного исполнения процедур идентификации образов
10 Валидация бинарных последовательностей, возникающих на выходе нейро-корреляционных преобразователей
11 Требования к тестированию качества нейро-корреляционных преобразователей
12 Приложение. Пример реализации НКП на языке C#

Приложение 5 Запрос на вступление в эксперты

УТВЕРЖДАЮ

Председатель ТК 164

С.В.Гарбук

« 25 » января 2022 г.

ПЛАН РАБОТЫ
технического комитета по стандартизации ТК 164 «Искусственный интеллект» на 2022 год

№ п/п	Наименование работ	Ответственный исполнитель	Срок исполнения	Результаты
1. Организационно-методическая работа				
1.1	Приведение состава и структуры ТК в соответствие с ГОСТ Р 1.1-2020	Секретариат ТК	01.06.2022	Приказ Росстандарта
1.2	Приведение положения о ТК в соответствие с ГОСТ Р 1.1-2020	Секретариат ТК	01.06.2022	Положение о ТК
1.3	Разработка типового положения и положений о подкомитетах	Секретариат ТК	01.06.2022	Типовое положение и положения о подкомитетах
1.4	Актуализация ПНС 2022 на основании предложений членов ТК, ФОИВ и заинтересованных организаций	Секретариат ТК	По мере поступления предложений	Предложения в Росстандарт
1.5	Подготовка предложений в Программу национальной стандартизации на 2023 год	Секретариат ТК	01.09.2022	Предложения в Росстандарт

17

№ п/п	Наименование работ	Ответственный исполнитель	Срок исполнения	Результаты
2.48	Рассмотрение первой редакции проекта национального стандарта ГОСТ Р «Искусственный интеллект. Нейросетевые алгоритмы в защищённом исполнении. Автоматическое обучение нейросетевых моделей на малых выборках в задачах классификации» (шифр ПНС 1.11.164-1.127.22)	ОМГТУ, ПК02 Секретариат ТК	10.10.2022	Первая редакция ГОСТ Р Первичная нормативная экспертиза
2.49	Рассмотрение первой редакции проекта национального стандарта ГОСТ Р «Системы искусственного интеллекта в управлении продукцией. Системы искусственного интеллекта для формирования плана маркетинга продукции. Типовые требования к испытанию» (шифр ПНС 1.11.164-1.128.22)	НП РУССОФТ, ПК02 Секретариат ТК	01.03.2022	Первая редакция ГОСТ Р Первичная нормативная экспертиза
2.50	Рассмотрение первой редакции проекта национального стандарта ГОСТ Р «Системы искусственного интеллекта на автомобильном транспорте. Системы для автономного управления движением промышленного транспорта. Общие положения» (шифр ПНС 1.11.164-1.129.22)	ПАО «Газпром нефть», ПК03 Секретариат ТК	01.04.2022	Первая редакция ГОСТ Р Первичная нормативная экспертиза
2.51	Рассмотрение первой редакции проекта национального стандарта ГОСТ Р «Системы искусственного интеллекта на автомобильном транспорте. Системы для автономного управления движением на зимних автомобильных дорогах и ледовых переправах. Общие положения» (шифр ПНС 1.11.164-1.130.22)	ПАО «Газпром нефть», ПК03 Секретариат ТК	01.04.2022	Первая редакция ГОСТ Р Первичная нормативная экспертиза
2.52	Рассмотрение первой редакции проекта национального стандарта ГОСТ Р «Системы искусственного интеллекта на автомобильном транспорте. Системы для автономного управления движением на промысловых объектах. Общие положения» (шифр ПНС 1.11.164-1.131.22)	ПАО «Газпром нефть», ПК03 Секретариат ТК	01.04.2022	Первая редакция ГОСТ Р Первичная нормативная экспертиза

Приложение 6 Предложение войти в состав экспертов от России Международного технического комитета ISO/IEC JTC 1/SC 42 «Artificial intelligence»

ТЕХНИЧЕСКИЙ КОМИТЕТ
ПО СТАНДАРТИЗАЦИИ
«ИСКУССТВЕННЫЙ ИНТЕЛЛЕКТ»
(ТК 164)

Покровский бульвар, д. 11,
Москва, 109028
info@tc164.ru

ФГБОУ ВО
«Омский государственный
технический университет»
(ОмГТУ)

Врио ректора
Маевскому Д.П.

Исх. от 25.05.2021 № 162/2-2021
В ответ на № 242/03-14 от 23.04.2021

Уважаемый Дмитрий Павлович!

В ответ на Ваше письмо № 242/03-14 от 23.04.2021 г. сообщаю следующее.

В соответствии с п. 5.1 ГОСТ Р 1.2-2020 «Стандартизация в Российской Федерации. Стандарты национальные Российской Федерации. Правила разработки, утверждения, обновления, внесения поправок и отмены» (далее – ГОСТ Р 1.2-2020) для организации разработки национального стандарта соответствующая тема должна быть включена в Программу национальной стандартизации (далее – ПНС) на текущий год. О сборе предложений по внесению в ПНС на 2022 год мероприятий по разработке национальных стандартов будет сообщено дополнительно.

В соответствии с разделом 5 ГОСТ Р 1.2-2020 на публичное обсуждение выносится первая редакция проекта национального стандарта, ранее направленная разработчиком в секретариат ТК. Вынесение технического отчета на публичное обсуждение не предусмотрено.

В соответствии с п. 6.6.2 ГОСТ Р 1.1-2020 «Стандартизация в Российской Федерации. Технические комитеты по стандартизации и проектные технические комитеты по стандартизации. Правила создания и деятельности» заявку для включения организации в состав ТК рассматривают на заседании ТК. В случае положительного решения, принятого на заседании,

Исп.: О.С. Миронова
Тел.: (495) 531-00-00*27898

председатель ТК направляет в федеральный орган исполнительной власти в сфере стандартизации данную заявку с соответствующим протоколом заседания комитета. О решении, принятом на заседании ТК, по включению в состав ТК 164 Омского государственного технического университета будет сообщено дополнительно.

Для повышения эффективности работ по сопровождению разработки международного документа ISO/IEC TR 5464 «Artificial intelligence – Functional safety and AI systems» прошу рассмотреть возможность включить в состав экспертов, зарегистрированных в глобальной директории ISO, кандидата технических наук, доцента – Сулавко Алексея Евгеньевича.

Ответственный секретарь О.С. Миронова

Приложение 7 Пример использования AIC desktop для анализа акустических образов уха

Эхо-сигналы сначала подвергались спектральному и кепстральному анализу, из них извлекались наиболее информативные данные, которые нормировались (приводились к единой размерности и области значений). Далее образы подавались на вход автокодировщику – нейронной сети со специальной архитектурой, которая способна сжимать размерность входных данных, кодируя их набором информативных признаков, а также восстанавливать входные данные из вектора признаков. Автокодировщик строился на базе многослойной сверточной нейронной сети. После извлечения вектора признаков (сжатия), данные не восстанавливаются. Вектор признаков может быть обработан с помощью реализованных моделей машинного обучения и классификаторов (в том числе, НПБК на базе корреляционных нейронов, НПБК, обучаемый по ГОСТ 52633.5-2001 и др.). Автокодировщик и классификаторы обучались и тестировались на различных непересекающихся выборках.

Акустические образы наружного уха могут быть загружены в конструктор нейронных сетей AIC. Образы будут представлены в виде набора классов. Так как набор данных сравнительно небольшого объема (2250 образов), перед обучением нейронных сетей следует не только нормировать образы по размерности, но и снизить их размерность, преобразовав в более компактный формат, позволяющий выделить информативные признаки. На основании предыдущих исследований было определено, что спектральный анализ исследуемых эхо-сигналов должен быть двухэтапным: 1 этап – вычисление усредненного амплитудного (энергетического) спектра эхо-сигнала, 2 этап – построение спектрограммы усредненного спектра (такую спектрограмму можно назвать кепстрограммой).

Этап 1. Предварительно из записей эхо-сигналов удалялась неинформативная часть (тишина). Для этого эхо-сигнал разбивался на 1000 интервалов. Интервалы с суммарной энергией меньшей 0,001 доли от общей

энергии эхо-сигнала удалялись в начале и конце записи. Построение усредненного спектра сводилось к вычислению спектрограммы эхо-сигнала с помощью Short Time Fourier Transform (STFT). При этом использовалось окно Хэмминга длиной 4096 (получался спектр из 2048 амплитуд) и шагом 2048 отчетов. Оконная функция Хэмминга часто применяется в обработке речевых сигналов для подавления шумов, связанных с разрывами в начале и конце окна дискретизированного сигнала. Далее спектрограмма интегрировалась (вычислялся усредненный спектр). При интегрировании осуществлялась обрезка спектров (удалялось 200 низкочастотных и 88 высокочастотных амплитуд). Таким образом, не учитывались частоты менее 2 кГц (чтобы возник резонанс на этой частоте, длина ушного канала должна значительно превышать референтный интервал) и более 21 кГц (эти частоты не регистрируются микрофоном и представляют собой шум). В итоге длина усредненных спектров составила 1760. Настройки AIC для этого этапа обработки приведены на рисунке 1 (можно преобразовать весь набор данных и сохранить промежуточный результат обработки в файл).

Этап 2. После преобразования всех данных в усредненные спектры, по ним строились спектрограммы с помощью STFT с использованием прямоугольного окна длиной 64 и шагом 32 отчета. При этом усредненный спектр принимался за временной ряд, поэтому его спектрограмму следует называть кепстрограммой (операция логарифмирования усредненного спектра не применялась, как это выполняется при обработке речевых сигналов). Оптимальность выбора окна Хэмминга на первом этапе и прямоугольного окна на втором обусловлена результатами прошлых исследований. Кепстрограмма характеризует амплитудно-частотную локализацию усредненного спектра, а не исходного сигнала. Исходный эхо-сигнал почти симметричен и гораздо менее информативен с точки зрения выявления признаков для классификации, чем усредненный спектр.

После всех преобразований ширина кепстрограмм составила 32 пикселя и длина 54 пикселя. Эти числа хорошо раскладываются на простые ($32=2^5$, $54=3^3\cdot2$), что желательно при построении автокодировщика на базе сверточной нейронной сети. Таким образом, выбор длины усредненного (энергетического) спектра также

обусловлен возможностью получения кепстрограммы почти оптимального размера с точки зрения дальнейшей нейросетевой обработки.

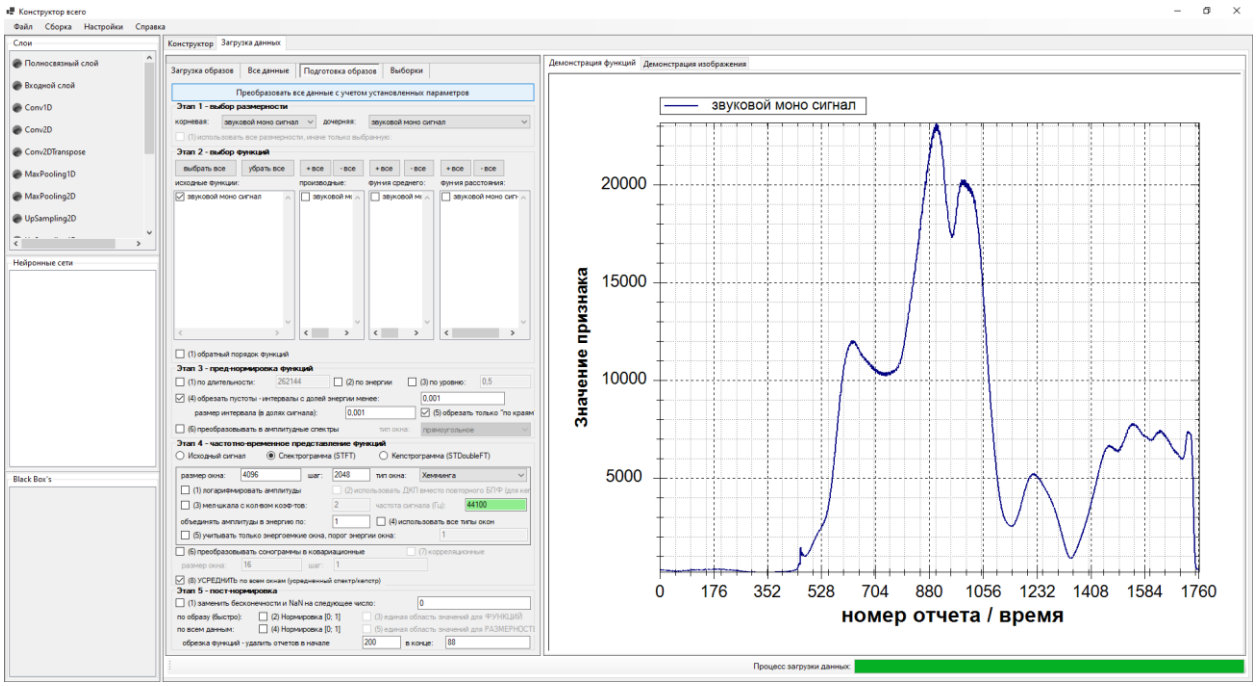


Рисунок 1 – Настройки (слева) параметров предварительной обработки образов для получения усредненного спектра

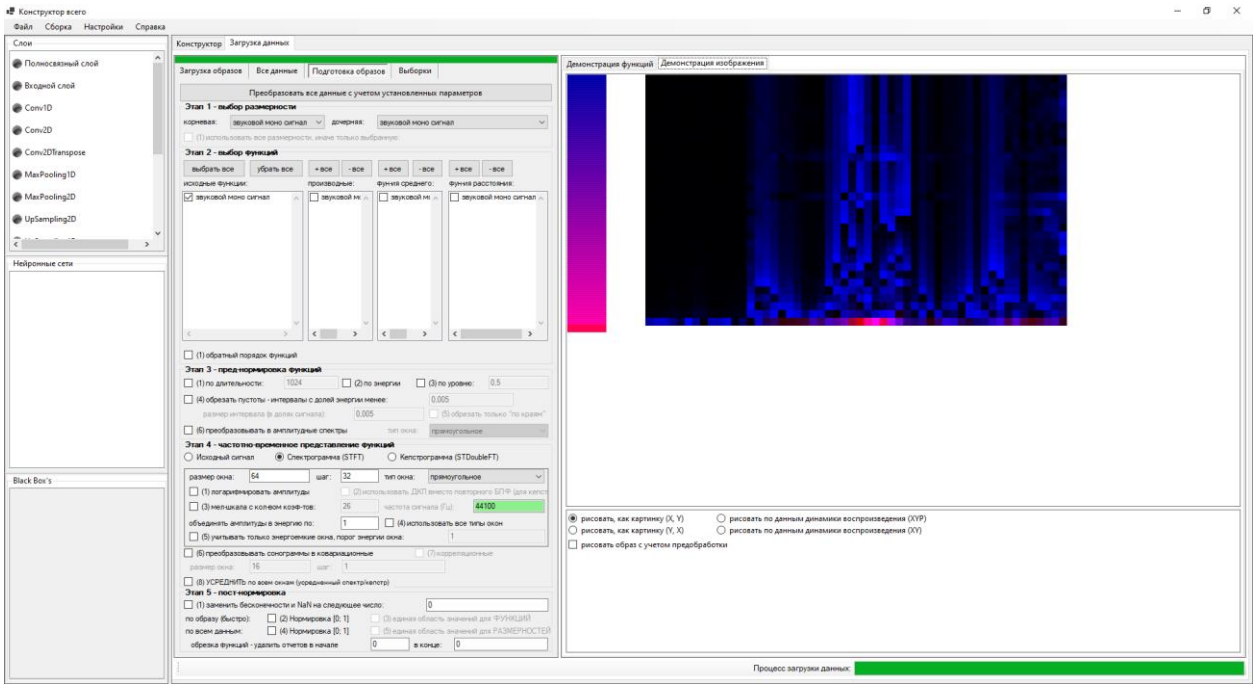


Рисунок 2 – Настройки (слева) параметров предварительной обработки образов для получения кепстрограммы (STFT-спектрограммы усредненного спектра)

Полученные кепстрограммы можно представить в виде изображений в градациях серого, каждый пиксель которых описывается положительным

действительным числом. Настройки AIC для второго этапа предварительной обработки образов приведены на рисунке 2 (цветовая индикация используется с целью лучшего восприятия изображения, на самом деле изображение не имеет RGB каналов).

Аугментация данных, построение и обучение автокодировщика для извлечения признаков.

Имеющегося количества образов достаточно только для обучения малых нейронных сетей (с небольшим числом слоев, нейронов и связей), если использовать итерационные алгоритмы обучения, основанные на градиентном спуске. Для увеличения объемов обучающей выборки можно применить методы аугментации данных (увеличения выборки через модификацию существующих данных). В настоящем исследовании объем выборки был расширен путем парного скрещивания (усреднения) образов (функции аугментации находятся на вкладке «Все данные» и применяются по отношению к выбранным классам образов, рисунок 3):

- скрещено по одной паре случайных образов из разных классов (создан новый класс синтетических образов, пригодный только для обучения автокодировщика);

- скрещены все возможные пары образов внутри каждого исходного класса, после чего количество примеров в классе составило 120 (15 исходных + 105 синтетических).

После аугментации объем обучающей выборки автокодировщика составил 11775 примеров. Опыты по распознаванию образов без аугментации данных также проводились, но результаты оказались значительно хуже (использовался аналогичный автокодировщик, а также автокодировщик с такой же «глубиной», но вдвое меньшим числом нейронов).

Далее значения кепстральных коэффициентов на всех частотах приводились к интервалу [0;1] (использовалась быстрая нормировка, как показано на рисунок 4, однако другие варианты нормирования данных также могут быть

использованы). Эта операция выполнялась с целью подготовки данных к подаче на вход нейронной сети.

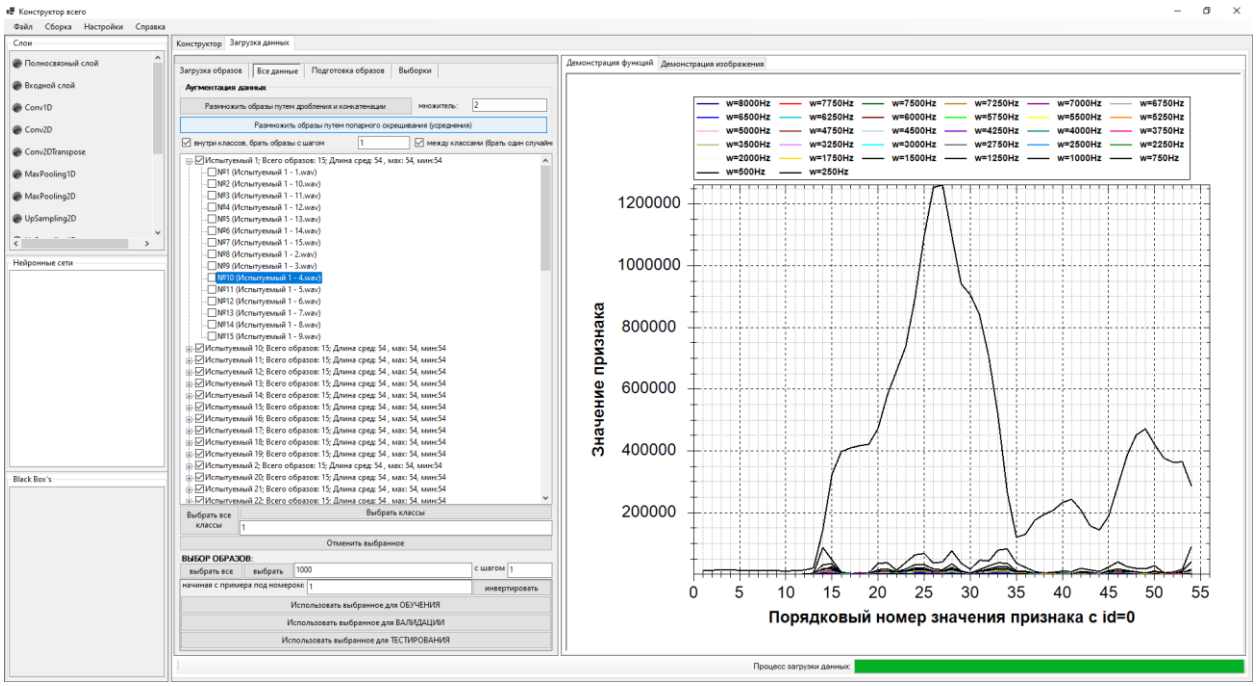


Рисунок 3 – Настройка параметров аугментации данных, выбор образов для просмотра, формирование выборок

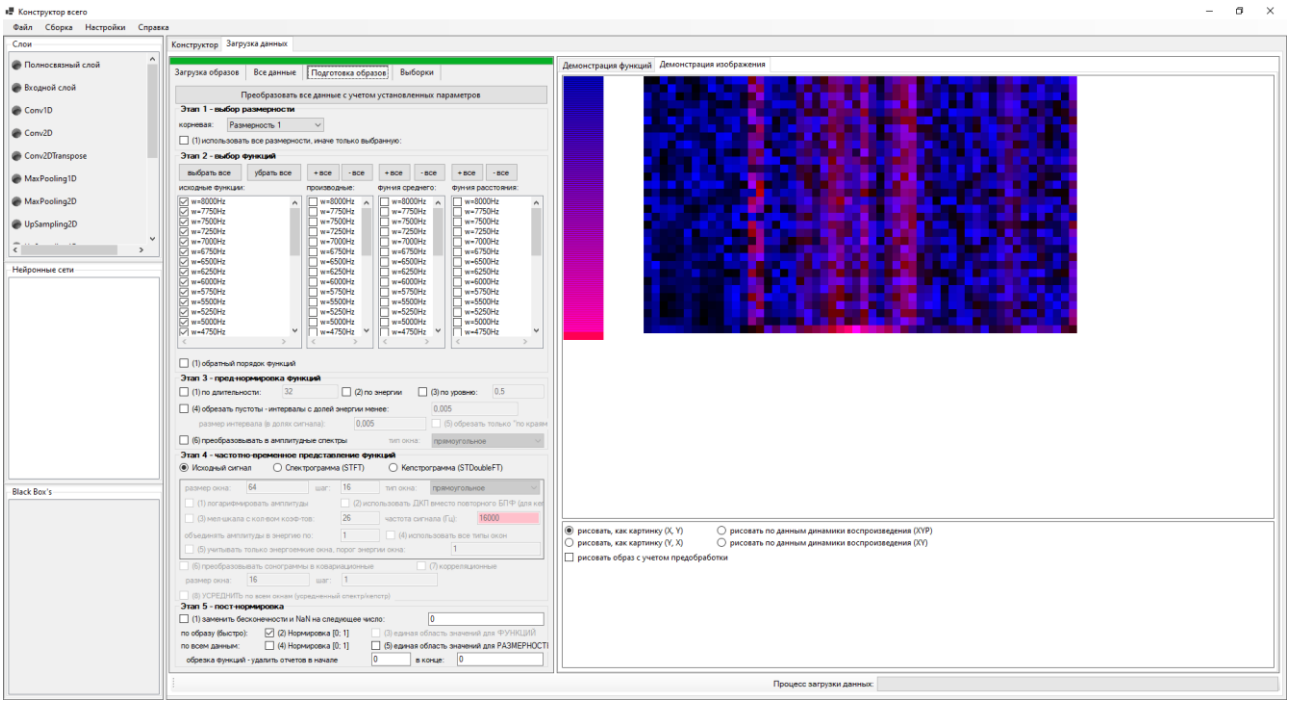


Рисунок 4 – Настройки нормирования образов перед их поступлением в нейронную сеть (после аугментации)

Синтез и обучение автокодировщика для извлечения признаков.

В конструкторе нейронных сетей AIC создан шаблон автокодировщика (меню «Файл->Создать»), состоящий из двух сетей – кодировщика и декодировщика. На рисунке 5а изображен интерфейс конструктора, в котором можно редактировать выбранную сеть, слои являются строительными блоками (слева), они «перетягиваются» на проект архитектуры сети (справа), чтобы настроить параметры слоя следует дважды кликнуть на него левой кнопкой мыши в проекте, чтобы создать связь между слоями – кликнуть правой кнопкой мыши. Созданная конфигурация автокодировщика представлена в таблице 1. После формирования конфигурации, автокодировщик был скомпилирован (меню «Сборка->Скомпилировать», для обучения выбран оптимизатор Adam, в качестве функции ошибки – бинарная кросс-энтропия, рисунок 5б). Перед обучением был установлен режим обработки «Изображения» (меню «Настройки»). На вкладке «Все данные» были выбраны образы левого уха и использованы в качестве обучающей выборки. Обучение проводилось в 5 эпох при размере батча в 64 образа (меню «Сборка->Обучить», рисунок 5б). Далее производилось дообучение по одной эпохе при размере батча в 128 и 256 образов соответственно (всего 7 эпох обучения). Хронологию обучения можно отследить в консоли (рисунок 6). Качество работы автокодировщика (сжатие и восстановление образа) после каждой эпохи можно протестировать (рисунок 7), таким образом контролировалось возникновение переобучения. Промежуточный вариант обученной сети можно сохранить в файл, для того, чтобы начать обучение заново следует перекомпилировать сеть.

Автокодировщик обучается как единая сеть, однако для извлечения признаков используется только кодировщик – сеть, отвечающая за сжатие пространства признаков. Декодировщик – сеть, отвечающая за восстановление данных, необходима только на этапе обучения (без нее невозможно обучить кодировщик). Из рисунка 6 видно, что общее количество параметров кодировщика составляет 65512, а декодировщика – 48953.

Таблица 1 – Конфигурация автокодировщика

Кодировщик		**Декодировщик**	
Тип слоя	Параметры слоя (кол-во фильтров/нейронов; размер ядра; strides; активация)	Тип слоя	Параметры слоя (кол-во фильтров/нейронов; размер ядра; strides; активация)
Сверточный	8; 3,3; 2,2; ReLu	Сверточный	64; 2,1; 2,1; ReLu
Сверточный	16; 3,4; 2,3; ReLu	Сверточный	32; 3,3; 2,2; ReLu
Batch-normalization		Batch-normalization	
Сверточный	32; 3,3; 2,2; ReLu	Сверточный	16; 3,4; 2,3; ReLu
Сверточный	64; 3,3; 2,2; ReLu	Сверточный	8; 3,4; 2,3; ReLu
Batch-normalization		Batch-normalization	
Сверточный	128; 2,1; 2,1; ReLu	Сверточный	1; 3,3; 2,2; ReLu
Полносвязный	128; ; ; linear		

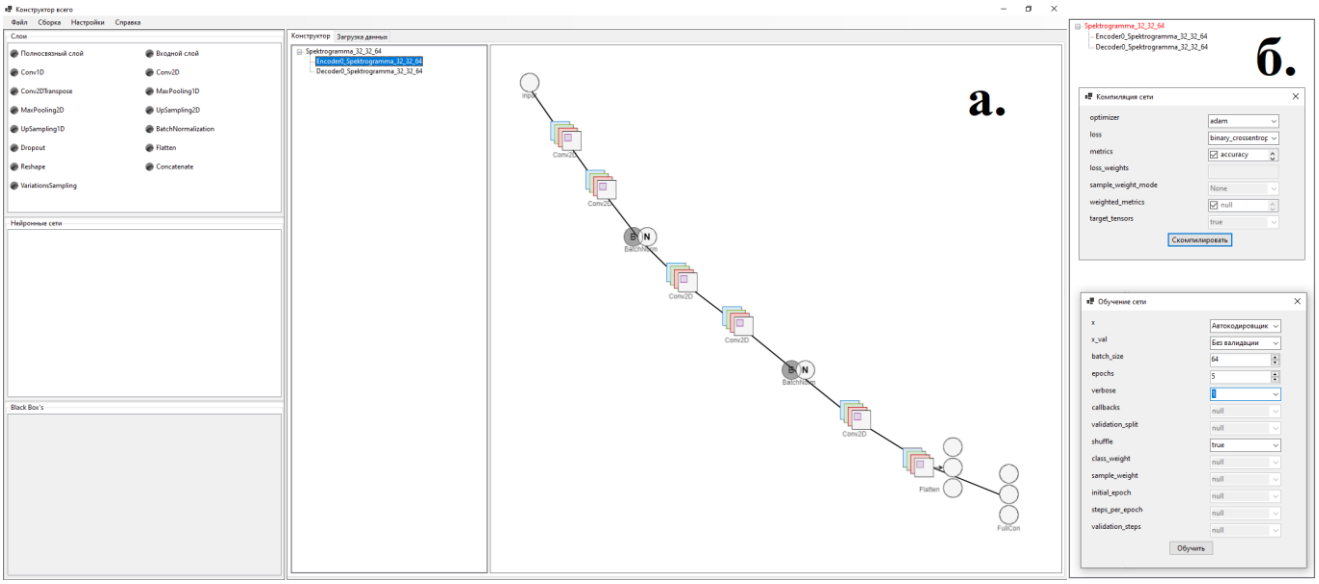


Рисунок 5 – Создание кодировщика: а. проектирование архитектуры; б. настройки параметров компиляции и обучения автокодировщика (в параметре «x» указан автокодировщик, а параметр «verbose» равен 1 – для вывода информации об обучении в консоль)

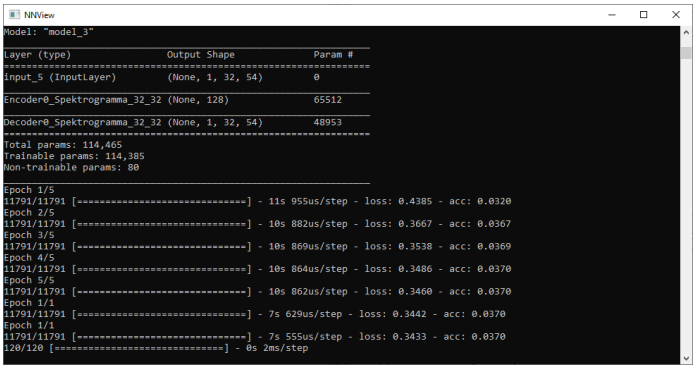


Рисунок 6 – Вывод информации о процессе обучения в консоль

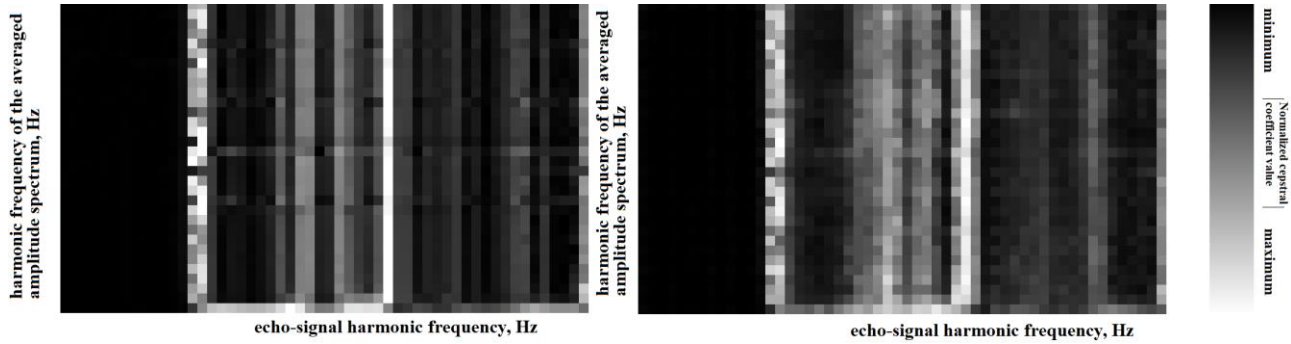


Рисунок 7 – Исходная нормированная кепстрограмма (слева) и восстановленная автокодировщиком кепстрограмма после 7 эпох обучения (справа)

Обучение и тестирование классификаторов.

Образы правого уха не использовались при обучении автокодировщика и были выбраны в качестве тестовой выборки для извлечения признаков (меню «Сборка->Выделить признаки», выбор соответствующего кодировщика осуществлялся на вкладке «Конструктор», рисунок 8). После извлечения векторов признаков они поступают в модуль статистического анализа программного комплекса AIC desktop (рисунок 9а).

Далее данные образов правого уха снова были разделены на обучающую и тестовую выборки (уже относительно классификаторов). Для обучения классификаторов выбрано по 7 примеров образов каждого класса (рисунок 9а), остальные примеры использовались в качестве тестовых. Применялось 4 классификатора:

1. «Наивный» классификатор Байеса (в меню «Модель/метод обучения» выбрана стандартная статистическая модель, которая описывает классы образов с помощью плотностей вероятности признаков, в меню «Классификатор/решающее правило» – «Классический (наивный) Байес», рисунок 9а).
2. Усовершенствованный байесовский классификатор, который позволяет рассчитывать апостериорную вероятность гипотез с учетом информативности признаков (в соответствующих меню выбраны стандартная статистическая модель и «Усовершенствованный (осторожный) Байес», в настройках классификатора установлены параметры, как на рисунок 9б).

3. Искусственная иммунная сеть (ИИС) (в качестве модели обучения выбрана ИИС, в качестве классификатора – взвешенная мера Хэмминга). Для ИИС установлены следующие основные параметры: количество детекторов врожденного иммунитета $N_{ВИ}$=50, количество детекторов приобретенного иммунитета $N_{ПИ}$ =0, количество итераций обучения $I_{ВИ}$=10.

4. Нейросетевой преобразователь биометрия-код (НПБК), обучаемый по ГОСТ Р 52633.5-2011 (в качестве модели обучения выбрана «Neoro-extractor», в качестве классификатора – мера Хэмминга). Для НПБК установлены параметры: 1 слой из 32 нейронов с 4 входами на каждый нейрон. Подробнее с алгоритмом настройки и правилами выбора параметров НПБК можно ознакомиться ГОСТ Р 52633.5-2011.

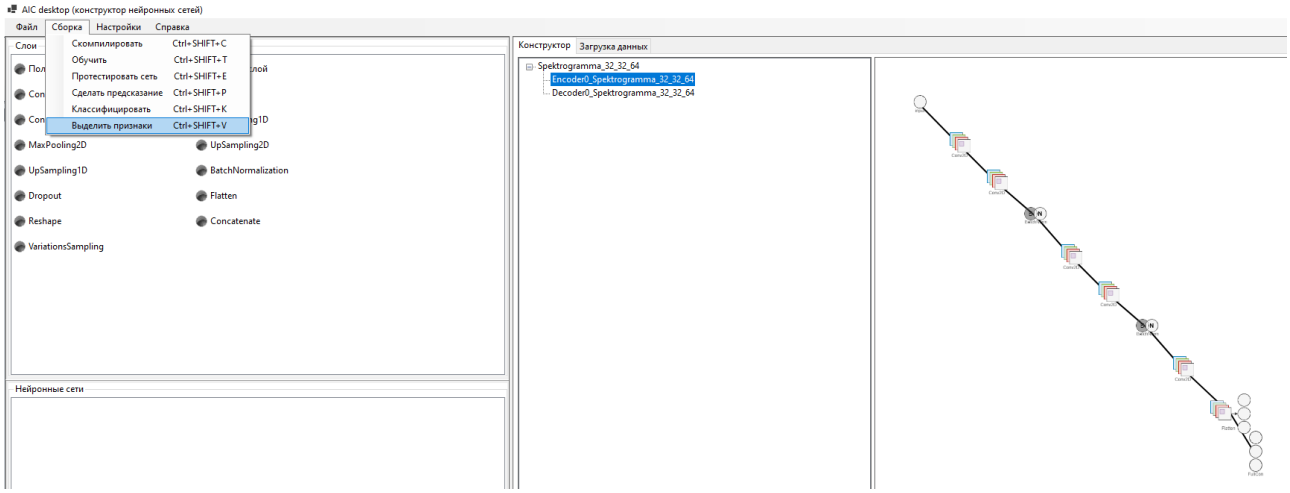


Рисунок 8 – Выбор кодировщика и извлечение признаков

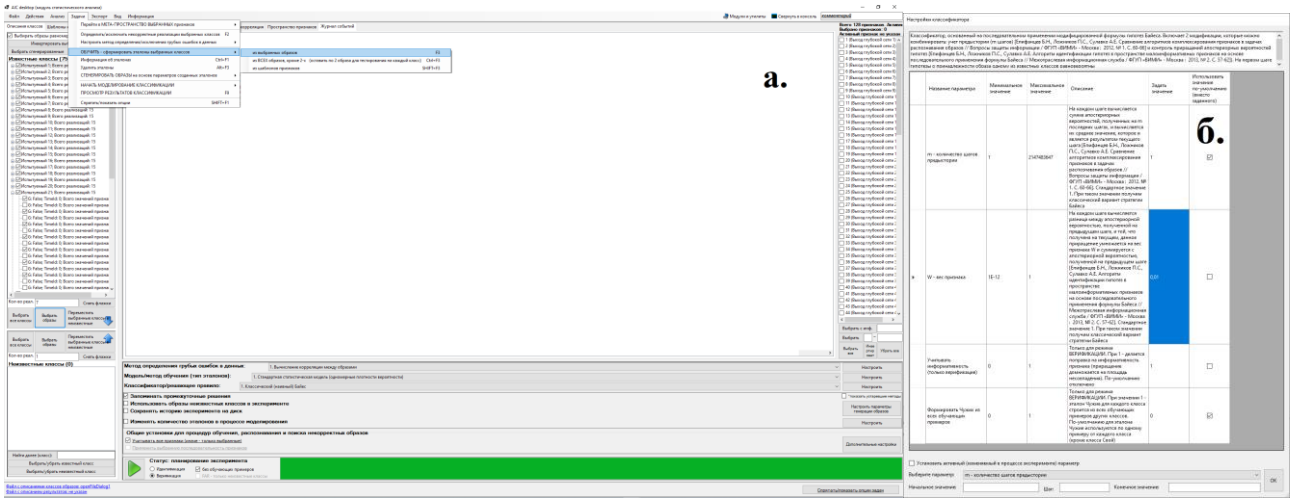


Рисунок 9 – Классификатор: а. обучение; б. настройка

После обучения моделей выбирались соответствующие классификаторы (параметры классификатора можно установить, нажав кнопку «Настроить»), и выполнялся запуск вычислительного эксперимента (меню «Задачи->Начать моделирование классификации->в режиме верификации»). Результаты экспериментов можно видеть на рисунке 10. Построение графиков осуществляется в модуля просмотра результатов моделирования.

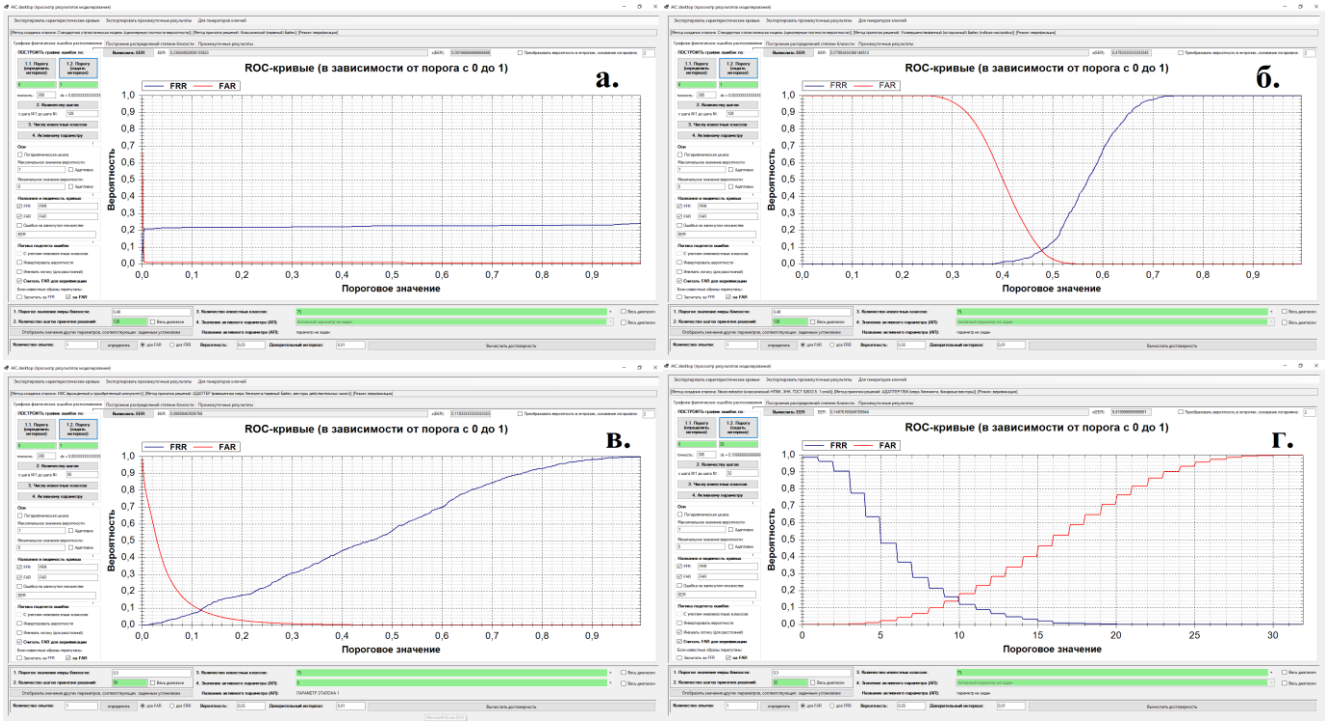


Рисунок 10 – Просмотр результатов моделирования и построения графиков зависимости вероятностей ошибок «ложного отказа» (FRR и «ложного допуска» (FAR) от порогового значения: а. «Наивный» классификатор Байеса, б. Усовершенствованный классификатор Байеса, в. ИИС, г. НПБК.

Рассмотренные классификаторы можно сравнить по коэффициенту равной вероятности ошибок (EER=FRR=FAR), который составляет (рисунок 10):

- «наивный» байесовский классификатор – EER≈0,23;
- усовершенствованный байесовский классификатор – EER≈0,08;
- ИИС – EER≈0,088;
- НПБК – EER≈0,15.

Приложение 8 Реализация нейросетевого преобразователя образов в код на основе корреляционных нейронов на языке C#

Реализация базовых абстрактных классов ядра.

```
namespace SHV.Kernel
{
   // Класс образов
   public class ImageClass
   {
      private String _name; // Название класса
      private Realization[] _realizations; // Реализации
      private FeaturesSpecification _specification; // Спецификация
признаков

      // Внутренняя функция для сокращения кода
      private FeatureCrossSection[] GetFirstCS(FeatureCrossSection[]
csTemp, int count)
      {
         FeatureCrossSection[] cs = new FeatureCrossSection[count];
         for (int i = 0; i < count; i++)
            cs[i] = csTemp[i];
         return cs;
      }

      public ImageClass(String name, FeaturesSpecification specification)
      {
         if ((name == null)||(specification==null)) throw new
ArgumentException();
         _name = name;
         _realizations = new Realization[0];
         _specification = specification;
      }

      #region AddRemoveAndFlags

      public void AddRealizations(List<Realization> reals)
      {
         Realization[] reals_tmp = new Realization[_realizations.Length +
reals.Count];
         for (int i = 0; i < _realizations.Length; i++)
            reals_tmp[i] = _realizations[i];
         for (int i = 0; i < reals.Count; i++)
         {
            if (reals[i] == null)
               throw new ArgumentException("Class: " + Name);
            reals[i].ImageClassName = _name;
            reals_tmp[i + _realizations.Length] = reals[i];
         }
         _realizations = null;
         _realizations = reals_tmp;
      }
      public void AddRealizations(Realization[] reals)
      {
         Realization[] reals_tmp = new Realization[_realizations.Length +
reals.Length];
         for (int i = 0; i < _realizations.Length; i++)
            reals_tmp[i] = _realizations[i];
         for (int i = 0; i < reals.Length; i++)
         {
            if (reals[i] == null)
               throw new ArgumentException("Class: " + Name);
            reals[i].ImageClassName = _name;
            reals_tmp[i + _realizations.Length] = reals[i];
         }
         _realizations = null;
         _realizations = reals_tmp;
      }

      public void RemoveRealizations(List<int> realIndexes)
      {
         List<Realization> reals = new List<Realization>();
         for (int i = 0; i < realIndexes.Count; i++)
            _realizations[realIndexes[i]]=null;
         for (int i = 0; i < _realizations.Length; i++)
            if (_realizations[i] != null)
               reals.Add(_realizations[i]);
         _realizations = null;
         _realizations = reals.ToArray();
      }

      #endregion


      #region Geters

      // Отдать определенные реализации
      public Realization[] GetRealizations(List<int> realizationIndexes)
      {
         if ((realizationIndexes == null) || (realizationIndexes.Count >
_realizations.Length)) throw new ArgumentException();
         Realization[] res = new Realization[realizationIndexes.Count];
         for (int i = 0; i < realizationIndexes.Count; i++)
            res[i] = _realizations[realizationIndexes[i]];
         return res;
      }

      // Отдать все реализации
      public Realization[] GetRealizations(int first, int last_from_end)
      {
         Realization[] reals = new Realization[_realizations.Length - first -
last_from_end];
         last_from_end = _realizations.Length - last_from_end;
         for (int i = first; i < last_from_end; i++)
            reals[i - first] = _realizations[i];
         return reals;
      }

      // Подготовить реализации к эксперименту (если нужно
переставить признаки и/или удалить неиспользуемые признаки)
      public Realization[] GetPreparedRealizations(List<int>
realizationIndexes, ushort[] uniqueFeaturesIds, bool originalPos)
      {
         if ((realizationIndexes == null) || (realizationIndexes.Count >
realizationIndexes.Count)) throw new ArgumentException();
         Realization[] res = new Realization[realizationIndexes.Count];
         for (int i = 0; i < realizationIndexes.Count; i++)
            res[i] =
Manager.GetPreparedRealization(_realizations[realizationIndexes[i]],
uniqueFeaturesIds, originalPos);
         return res;
      }
      public Realization[] GetPreparedRealizations(ushort[]
uniqueFeaturesIds, bool originalPos, int first, int last_from_end)
      {
         Realization[] res = new Realization[_realizations.Length - first -
last_from_end];
         last_from_end = _realizations.Length - last_from_end;
         for (int i = first; i < last_from_end; i++)
            res[i - first] = Manager.GetPreparedRealization(_realizations[i],
uniqueFeaturesIds, originalPos);
         return res;
      }
```

```
        // Дать сечения определенных признаков всех реализаций (отдает только те, что не равны null)
        public FeatureCrossSection[] GetCrossSections(ushort[] uniqueFeaturesIds)
        {
            FeatureCrossSection[] csTemp = new FeatureCrossSection[uniqueFeaturesIds.Length];
            int count = 0;
            for (int i = 0; i < uniqueFeaturesIds.Length; i++)
            {
                double[] tmp = GetCrossSection(uniqueFeaturesIds[i]);
                if (tmp != null)
                {
                    csTemp[count]._id = uniqueFeaturesIds[i];
                    csTemp[count]._values = tmp;
                    count++;
                }
            }
            return GetFirstCS(csTemp, count);
        }

        // Дать сечения определенных признаков и реализаций (отдает только те, что не равны null)
        public FeatureCrossSection[] GetCrossSections(ushort[] uniqueFeaturesIds, List<int> realizationIndexes)
        {
            FeatureCrossSection[] csTemp = new FeatureCrossSection[uniqueFeaturesIds.Length];
            int count = 0;
            for (int i = 0; i < uniqueFeaturesIds.Length; i++)
            {
                double[] tmp = GetCrossSection(uniqueFeaturesIds[i], realizationIndexes);
                if (tmp != null)
                {
                    csTemp[count]._id = uniqueFeaturesIds[i];
                    csTemp[count]._values = tmp;
                    count++;
                }
            }
            return GetFirstCS(csTemp, count);
        }

        // Дать сечения определенных признаков и реализаций (если сечения нет, все равно отдает, но _values = null)
        public FeatureCrossSection[] GetCrossSectionsWithNull(ushort[] uniqueFeaturesIds, List<int> realizationIndexes)
        {
            FeatureCrossSection[] cs = new FeatureCrossSection[uniqueFeaturesIds.Length];
            for (int i = 0; i < uniqueFeaturesIds.Length; i++)
            {
                cs[i]._id = uniqueFeaturesIds[i];
                cs[i]._values = GetCrossSection(uniqueFeaturesIds[i], realizationIndexes);
            }
            return cs;
        }

        // Дать сечения определенных признаков всех реализаций (если сечения нет, все равно отдает, но _values = null)
        public FeatureCrossSection[] GetCrossSectionsWithNull(ushort[] uniqueFeaturesIds)
        {
            FeatureCrossSection[] cs = new FeatureCrossSection[uniqueFeaturesIds.Length];
            for (int i = 0; i < uniqueFeaturesIds.Length; i++)
            {
                cs[i]._id = uniqueFeaturesIds[i];
                cs[i]._values = GetCrossSection(uniqueFeaturesIds[i]);
            }
            return cs;
        }

        // Дать сечение признака всех реализаций (если его нет - отдает null)
        public double[] GetCrossSection(ushort id)
        {
            ushort index = _specification.GetFeatureIndex(id);
            if (index == ushort.MaxValue) return null;
            List<double> values = new List<double>();
            for (int i = 0; i < _realizations.Length; i++)
            {
                List<double> cs = _realizations[i].GetCrossSection(id, new List<double>());
                for (int j = 0; j < cs.Count; j++)
                    values.Add(cs[j]);
            }
            return values.ToArray();
        }

        // Дать сечение признака определенных реализаций (если его нет - отдает null)
        public double[] GetCrossSection(ushort id, List<int> realizationIndexes)
        {
            ushort index = _specification.GetFeatureIndex(id);
            if (index == ushort.MaxValue) return null;
            List<double> values = new List<double>();
            for (int i = 0; i < realizationIndexes.Count; i++)
            {
                List<double> cs = _realizations[realizationIndexes[i]].GetCrossSection(id, new List<double>());
                for (int j = 0; j < cs.Count; j++)
                    values.Add(cs[j]);
            }
            return values.ToArray();
        }

        public List<double> AddValuesToCrossSection(ushort id, List<int> realizationIndexes, List<double> cs)
        {
            ushort index = _specification.GetFeatureIndex(id);
            if (index == ushort.MaxValue) return null;
            for (int i = 0; i < realizationIndexes.Count; i++)
            {
                List<double> tmp = _realizations[realizationIndexes[i]].GetCrossSection(id, new List<double>());
                for (int j = 0; j < tmp.Count; j++)
                    cs.Add(tmp[j]);
            }
            return cs;
        }

        // Вернуть среднее среднеквадратичное отклонение всех признаков
        public double GetMxOfAllFeaturesValues(ushort[] ids)
        {
            FeatureCrossSection[] cs = GetCrossSections(ids);
            double[] sx = new double[cs.Length];
            for (int j = 0; j < cs.Length; j++)
                sx[j] = Statistica.Sx_a(cs[j]._values, cs[j]._values.Length);
            return Statistica.Mx_a(sx, sx.Length);
        }

        #endregion


        #region Properties

        public String Name // Название класса
        {
            get { return _name; }
            set { _name = value; }
        }

        public Realization[] Realizations // Реализации
        {
            get
            {
                return _realizations;
            }
        }
```

```
        #endregion
    }

    // Признак, эталонные характеристики признака образа, представляет собой закон распределения значений признаков и его параметры
    abstract public class IFeature
    {
        protected int _typeId; // Идентификатор типа признака
        // Принадлежит ли сечение признаку? На данный момент реализовано через критерий Хи-квадрат. Если недостаточно информации то также возвращается false
        public bool IsCrossSectionBelongsToFeature(double[] featureValues, float significanceLevel, int accuracy)
        {
            int degreesOfFreedom = accuracy - DistribLawParamQuantity - 1;
            if (degreesOfFreedom < 1) return false;
            int featureValuesNumber = featureValues.Length;
            double[] intervalsLimits = Statistica.GetIntervalsLimits(featureValues, accuracy);
            double[] probabilities = new double[accuracy + 1];
            for(int i=0;i<probabilities.Length;i++)
                probabilities[i]=GetDistributionFunction(intervalsLimits[i]);
            double xiObserved = Statistica.GetXiObserved(featureValues, probabilities, intervalsLimits, accuracy);
            double xiKritical = Statistica.GetXiTheoretical(significanceLevel, degreesOfFreedom);
            if (xiObserved < xiKritical)
                return true;
            else
                return false;
        }

        public int TypeId { get { return _typeId; } } // Идентификатор типа признака

        #region Methods_MustBeImplemented

        // Вернуть значения параметров закона распределения признака
        abstract public Object[] GetDistribLawParam();

        // Вернуть плотность вероятности заначения признака
        abstract public double GetDensityOfProb(double featureValue);

        // Вернуть значение функции распределения от заначения признака
        abstract public double GetDistributionFunction(double featureValue);

        // Вернуть левую границу для построения графика плотности распределения (минимальное вероятное значение признака)
        abstract public double GetMinimumLimit();

        // Вернуть правую границу для построения графика плотности распределения (правую вероятное значение признака)
        abstract public double GetMaximumLimit();

        // Вернуть максимальное значение плотности распределения
        abstract public double GetMaximumValueOfDensity();

        #endregion

        #region Properties_MustBeImplemented

        abstract public int DistribLawParamQuantity { get; } // Количество параметров закона распределения признака
        abstract public double Mx { get; } // Математическое ожидание значений признака
        abstract public double Dx { get; } // Дисперсия значений признака
        abstract public double Sx { get; } // Среднеквадратичное отклонение значений признака
        abstract public ushort Id { get; } // Идентификатор признака
        abstract public String TypeName { get; } // Название типа признака
        abstract public bool CanGenerateFeaturesValues { get; } // Может ли признак генерировать значения?

        #endregion

        #region CanBeImplementedOrNot

        // Пересчитать параметры распределения
        abstract public void Refresh(double mx, double sx);
        // Сгенерировать значение признака/сечение
        // Функции нужно реализовывать, если CanGenerateFeaturesValues=true (поле здесь и в SHV.Services.FeaturesFactory), иначе просто прописать throw new NotImplementedException();
        abstract public double GenerateFeatureValue();
        abstract public double[] GenerateCrossSection(int quantityOfValues);

        #endregion
    }

    // Эталон, эталонные характеристики признаков класса образов (образа)
    abstract public class IEtalon
    {
        // Реализации, из которых был сформирован эталон (если они были переданы в конструктор)
        protected Realization[] _ownRealizations;
        protected IFeature[] _features; // Эталонные описания признаков
        protected String _imageClassName; // Имя класса, которому принадлежит реализация
        protected ushort[] _featuresIndexes; // Индексы признаков для индексации идентификаторов признаков
//        protected bool _remainHistoryOfTraning; // Запоминать историю обучения (если этот функционал реализован, актуально для нейронных сетей)
        protected int _typeId;
        protected ushort _maxFeatureIdInGlobalScope;

        // Вернуть степени соответствия реализации эталону (чатные/условные вероятности, плотности вероятности, значения функций принадлежности для признаков и т.д.)
        // Степень соответствия должна быть тем больше, чем ближе реализация к образу (в отличии от расстояния)
        // СТАНДАРТНАЯ РЕАЛИЗАЦИЯ
        // В качестве степеней близости вектора значений признаков отдает вектор плотностей вероятности этих значений
        // Если какого-то признака нет в эталоне, то плотность вероятности для этого признака равна нулю
        public virtual double[] GetProximityVector(Realization realization, double?[] parameters = null) //, double optionalParameter2 = Double.MinValue)
        {
            ushort[] ids = realization.GetAllFeaturesIds();
            double[] values = realization.GetAllFeaturesValues();
            int countOfVals = values.Length;
            double[] degrees = new double[countOfVals];
            for (int i = 0; i < countOfVals; i++)
                if ((_featuresIndexes.Length <= ids[i]) || (_featuresIndexes[ids[i]] == ushort.MaxValue))
                    degrees[i] = 0;
                else
                    degrees[i] = _features[_featuresIndexes[ids[i]]].GetDensityOfProb(values[i]);
            return degrees;
        }

        // Вернуть битовый вектор соответствия реализации эталону (выходы нейронной сети и т.д.)
        // Первый опциональный параметр - это коэффициент Стьюдента для интервала Хэмминга
        public virtual BitArray GetBinaryProximityVector(Realization realization, double?[] parameters = null)
        {
            ushort[] ids = realization.GetAllFeaturesIds();
            double[] values = realization.GetAllFeaturesValues();
            int countOfVals = values.Length;
            double s = 1;
            if (parameters!=null)
                if (parameters[0] != null)
                    s = (double)parameters[0];
            BitArray degrees = new BitArray(countOfVals);
            for (int i = 0; i < countOfVals; i++)
                if ((_featuresIndexes.Length <= ids[i]) || (_featuresIndexes[ids[i]] == ushort.MaxValue))
```

```
                degrees[i] = true;
            else
            {
                if (((Features[_featuresIndexes[ids[i]]].Mx -
(s*Features[_featuresIndexes[ids[i]]].Sx)) < values[i]) &&
((Features[_featuresIndexes[ids[i]]].Mx +
(s*Features[_featuresIndexes[ids[i]]].Sx) > values[i]))
                    degrees[i] = false;
                else
                    degrees[i] = true;
            }
        return degrees;
    }
    public virtual BitArray GetCorrectBinaryCode(int
featuresVectorLenth)
    {
        if (featuresVectorLenth == 0)
        {
            if (HaveOwnRealizations())
                for (int i = 0; i < OwnRealizations.Length; i++)
                    if (OwnRealizations[i].GetAllFeaturesValues().Length >
featuresVectorLenth)
                        featuresVectorLenth =
OwnRealizations[i].GetAllFeaturesValues().Length;
        }
        else
            featuresVectorLenth = Features.Length;
        BitArray res = new BitArray(featuresVectorLenth);
        return res;
    }

    // Показать историю обучения, если она есть
    public virtual List<String> GetHistoryOfTraining() { return null; } //
Текстовое описание
    public virtual List<double> GetHistoryOfTraining(int
errorFunctionType) { return null; } // Описание в виде значений
функции ошибки или характеристик нейронов
    public virtual String[] GetErrorFunctionName() { return null; } //
Название функции ошибки / характеристики нейронов



    #region Implemented

    // Обязательными параметрами для создания эталона являются
массив IFeature и ImageClass, остальное - реализации и сечения могут
быть переданы нулевые (null)
    // Все признаки (и сечения) индексируются, предполагается, что
порядок признаков и сечений (если featureCrossSections не равно null)
в передаваемых параметрах одинаков
    public IEtalon(String imageClassName, IFeature[] features,
Realization[] realizations, FeatureCrossSection[] featureCrossSections,
ushort maxFeatureIdInGlobalScope, double?[] parameters, int typeId)
    {
        //          _ownCrossSections = featureCrossSections;
        if (realizations == null)
            _ownRealizations = new Realization[0];
        else
            _ownRealizations = realizations;
        if ((features == null) || (features.Length < 1)) throw new
ArgumentException();
        _features = features;
        _imageClassName = imageClassName;
        _typeId = typeId;
        _maxFeatureIdInGlobalScope = maxFeatureIdInGlobalScope;

        // Индексация значений признаков
        _featuresIndexes = new ushort[maxFeatureIdInGlobalScope + 1];
        for (int i = 0; i < maxFeatureIdInGlobalScope; i++)
            _featuresIndexes[i] = ushort.MaxValue;
        for (ushort i = 0; i < features.Length; i++)
            _featuresIndexes[features[i].Id] = i;
    }

    // Имеет ли эталон собственные реализации (или только может
генерировать их)?
    public bool HaveOwnRealizations()
    {
        if ((_ownRealizations == null) || (_ownRealizations.Length == 0))
            return false;
        return true;
    }

    // Вернуть идентификаторы эталонных признаков
    public ushort[] GetFeaturesIds()
    {
        ushort[] ids = new ushort[_features.Length];
        for (int i = 0; i < _features.Length; i++)
            ids[i] = _features[i].Id;
        return ids;
    }

    protected Realization GenerateRealization(int sizeOfRealization, long
timeId, int[] genFeaturesIndexes, int countOfGenFeatures)
    {
        int count = 0;
        double[] values = new double[countOfGenFeatures *
sizeOfRealization];
        ushort[] ids = new ushort[countOfGenFeatures *
sizeOfRealization];
        for (int j = 0; j < sizeOfRealization; j++)
            for (int i = 0; i < countOfGenFeatures; i++)
            {
                values[count] =
_features[genFeaturesIndexes[i]].GenerateFeatureValue();
                ids[count] = _features[genFeaturesIndexes[i]].Id;
                count++;
            }
        genFeaturesIndexes = null;
        return new Realization(values, ids, timeId, true);
    }

    // Сгенерировать реализацию, sizeOfRealization - количество
итераций генерации, можно генерировать длинные реализации
    // Эту функцию целесообразно использовать, если нужно
имитировать несколько процессов ввода реализаций, каждый ввод -
одна сессия распознавания
    public Realization GenerateRealization(int sizeOfRealization, long
timeId)
    {
        int[] genFeaturesIndexes = new int[_features.Length];
        int countOfGenFeatures = 0;
        for (int i = 0; i < _features.Length; i++)
            if (_features[i].CanGenerateFeaturesValues == true)
            {
                genFeaturesIndexes[countOfGenFeatures] = i;
                countOfGenFeatures++;
            }
        if ((countOfGenFeatures == 0) || (sizeOfRealization < 1)) return
null;
        genFeaturesIndexes = null;
        return GenerateRealization(sizeOfRealization, timeId,
genFeaturesIndexes, countOfGenFeatures);
    }

    // Сгенерировать реализации, quantityOfRealizations - количество
генерируемых "одинарных" реализаций, т.е. параметр
sizeOfRealization всегда равен 1
    // Эту функцию целесообразно использовать, если нужно
имитировать несколько процессов ввода реализаций, каждый ввод -
одна сессия распознавания
    public Realization[] GenerateRealizations(int quantityOfRealizations,
int sizeOfRealization, long timeId)
    {
        int[] genFeaturesIndexes = new int[_features.Length];
        int countOfGenFeatures = 0;
        for (int i = 0; i < _features.Length; i++)
            if (_features[i].CanGenerateFeaturesValues == true)
            {
                genFeaturesIndexes[countOfGenFeatures] = i;
                countOfGenFeatures++;
            }
        if ((countOfGenFeatures == 0) || (quantityOfRealizations < 1))
return null;
        Realization[] reals = new Realization[quantityOfRealizations];
        for (int j = 0; j < quantityOfRealizations; j++)
```

```
            reals[j] = GenerateRealization(sizeOfRealization, timeId,
genFeaturesIndexes, countOfGenFeatures);
            genFeaturesIndexes = null;
            return reals;
        }

        // Сгенерировать реализации, quantityOfRealizations - количество
генерируемых "одинарных" реализаций, т.е. параметр
sizeOfRealization всегда равен 1
        // featuresIdSequance - последовательность входа признаков в
реализациях
        // Эту функцию целесообразно использовать, если нужно
имитировать процесс непрерывного ввода реализаций и
распознавания в реальном времени
        public Realization[] GenerateRealizations(int quantityOfRealizations,
int sizeOfRealization, long timeId, ushort[] featuresIdSequance)
        {
            int[] genFeaturesIndexes = new int[featuresIdSequance.Length];
            int countOfGenFeatures = 0;
            for (int i = 0; i < featuresIdSequance.Length; i++)
                if (GetFeatureIndex(featuresIdSequance[i]) != null)
                    if
(_features[(int)GetFeatureIndex(featuresIdSequance[i])].CanGenerateFeat
uresValues == true)
                    {
                        genFeaturesIndexes[countOfGenFeatures] =
(int)GetFeatureIndex(featuresIdSequance[i]);
                        countOfGenFeatures++;
                    }
            if ((countOfGenFeatures == 0) || (quantityOfRealizations < 1))
return null;
            Realization[] reals = new Realization[quantityOfRealizations];
            for (int j = 0; j < quantityOfRealizations; j++)
                reals[j] = GenerateRealization(sizeOfRealization, timeId,
genFeaturesIndexes, countOfGenFeatures);
            genFeaturesIndexes = null;
            return reals;
        }

        // Вернуть собственное сечение (совокупность значений
определенного признака, из которых было сформировано эталонное
описание признака, если он собирался из них)
        public double[] GetOwnCrossSection(ushort featureId)
        {
            // Если неверный Id признака, то возвращает null
            if (_featuresIndexes[featureId] == ushort.MaxValue) return null;
            ushort index = (ushort)_featuresIndexes[featureId];
            // Если есть сечение в исходном виде - возвращает сечение
//            if (_ownCrossSections != null)
//                return _ownCrossSections[index]._values;
            // Если нет сечения и реализаций, то возвращает null
            if ((OwnRealizations == null) || (OwnRealizations.Length == 0))
                return null;
            // Если нет сечений, но есть реализации - то составляет
сечение и возвращает его
            List<double> res = new List<double>();
            for (int i = 0; i < OwnRealizations.Length; i++)
            {
                int pos = 0;
                double? value =
OwnRealizations[i].FindFeatureValue(featureId, ref pos);
                while (value != null)
                {
                    res.Add((double)value);
                    value = OwnRealizations[i].FindFeatureValue(featureId, ref
pos);
                }
            }
            return res.ToArray();
        }
        public List<double> GetOwnCrossSection(ushort featureId, int
maxValuesCount)
        {
            // Если неверный Id признака, то возвращает null
            if (_featuresIndexes[featureId] == ushort.MaxValue) return null;
            ushort index = (ushort)_featuresIndexes[featureId];
            // Если нет сечения и реализаций, то возвращает null
            if ((OwnRealizations == null) || (OwnRealizations.Length == 0))
                return null;
            Random r = UniformDistribution.GetRandom();
            // Если нет сечений, но есть реализации - то составляет
сечение и возвращает его
            List<double> res = new List<double>();
            double tmp = Double.MinValue;
            int counter = 0;
            for (int i = 0; i < OwnRealizations.Length; i++)
            {
                int pos = 0;
                double? value =
OwnRealizations[i].FindFeatureValue(featureId, ref pos);
                while (value != null)
                {
                    double tmp2 = (double)value;
                    if (tmp != tmp2)
                    {
                        counter++;
                        tmp = tmp2;
                    }
//                    if (r.Next(3) > 0)
                    res.Add(tmp2);
                    if ((res.Count >= maxValuesCount) && (counter > 1))
                        return res;
                    value = OwnRealizations[i].FindFeatureValue(featureId, ref
pos);
                }
            }
            return res;
        }
        public List<double>[] Get2CoherentOwnMiddleCrossSections(ushort
featureId_1, ushort featureId_2)
        {
            // Если неверный Id хотябы одного из признаков, то
возвращает null
            if ((_featuresIndexes[featureId_1] == ushort.MaxValue) ||
(_featuresIndexes[featureId_2] == ushort.MaxValue)) return null;
            ushort index1 = (ushort)_featuresIndexes[featureId_1];
            ushort index2 = (ushort)_featuresIndexes[featureId_1];
            // Если нет реализаций, то возвращает null
            if ((OwnRealizations == null) || (OwnRealizations.Length == 0))
                return null;
            // Если есть реализации - то составляет 2 когерентных
(совпадающих по фазе сечения) усредненных сечения и возвращает
их
            List<double>[] res = new List<double>[2];
            res[0] = new List<double>();
            res[1] = new List<double>();
            for (int i = 0; i < OwnRealizations.Length; i++)
            {
                /*                List<double> values1 =
OwnRealizations[i].GetCrossSection(featureId_1, new List<double>());
                    List<double> values2 =
OwnRealizations[i].GetCrossSection(featureId_2, new List<double>());
                    if ((values1.Count < 1) && (values2.Count < 1))
continue;
                    if (values1.Count == 1)
                        res[0].Add(values1[0]);
                    else
                        if (values1.Count > 1)
                            res[0].Add(Statistica.Mx_a(values1.ToArray(),
values1.Count));
                    if (values2.Count == 1)
                        res[1].Add(values2[0]);
                    else
                        if (values2.Count > 1)
                            res[1].Add(Statistica.Mx_a(values2.ToArray(),
values2.Count));*/
                int tmp = 0;
                double? value1 =
OwnRealizations[i].FindFeatureValue(featureId_1, ref tmp);
                if (value1 == null) continue;
                double? value2 =
OwnRealizations[i].FindFeatureValue(featureId_2, ref tmp);
                if (value2 == null) continue;
                res[0].Add((double)value1);
                res[1].Add((double)value2);
            }
```

```
        return res;
    }

    // Индексация признаков для быстрого доступа без поиска
    public ushort? GetFeatureIndex(ushort featureId)
    {
        if (_featuresIndexes[featureId] == ushort.MaxValue)
            return null;
        return _featuresIndexes[featureId];
    }

    public int GetFeatureIndex_a(ushort featureId)
    {
        return _featuresIndexes[featureId];
    }

    public IFeature GetFeature(uint featureId)
    {
        if (_featuresIndexes[featureId] == ushort.MaxValue)
            return null;
        return _features[_featuresIndexes[featureId]];
    }

    public double[] GetVectorOfMx(ushort[] ids, double insteadNull)
    {
        double[] res = new double[ids.Length];
        for (int i = 0; i < ids.Length; i++)
            if (_featuresIndexes[ids[i]] == ushort.MaxValue)
                res[i] = insteadNull;
            else
                res[i] = _features[_featuresIndexes[ids[i]]].Mx;
        return res;
    }

    public double[] GetVectorOfSx(ushort[] ids, double insteadNull)
    {
        double[] res = new double[ids.Length];
        for (int i = 0; i < ids.Length; i++)
            if (_featuresIndexes[ids[i]] == ushort.MaxValue)
                res[i] = insteadNull;
            else
                res[i] = _features[_featuresIndexes[ids[i]]].Sx;
        return res;
    }

    public double[] GetVectorOfMx()
    {
        double[] res = new double[_features.Length];
        for (int i = 0; i < _features.Length; i++)
            res[i] = _features[i].Mx;
        return res;
    }

    public double[] GetVectorOfSx()
    {
        double[] res = new double[_features.Length];
        for (int i = 0; i < _features.Length; i++)
            res[i] = _features[i].Sx;
        return res;
    }

    public double[] GetVectorOfMaxAbsValuesOfFeatures(ushort[] ids,
double insteadNull)
    {
        double[] res = new double[ids.Length];
        for (int i = 0; i < ids.Length; i++)
            if (_featuresIndexes[ids[i]] == ushort.MaxValue)
                res[i] = insteadNull;
            else
            {
                res[i] =
Math.Abs(_features[_featuresIndexes[ids[i]]].GetMaximumLimit());
                if (res[i] <
Math.Abs(_features[_featuresIndexes[ids[i]]].GetMinimumLimit()))
                    res[i] =
Math.Abs(_features[_featuresIndexes[ids[i]]].GetMinimumLimit());
            }
        return res;
    }

    // Перемешать реализации
    public void MixRealizations()
    {
        Random rand = UniformDistribution.GetRandom();
        for (int i = _ownRealizations.Length - 1; i >= 1; i--)
        {
            int j = rand.Next(i + 1);
            // обменять значения data[j] и data[i]
            var temp = _ownRealizations[j];
            _ownRealizations[j] = _ownRealizations[i];
            _ownRealizations[i] = temp;
        }
    }

    #endregion

    #region ImplementedRounding

    public byte[] ToBytes_2(ushort[] ids)
    {
        byte[] res = new byte[ids.Length];
        for (int i = 0; i < ids.Length; i++)
            if ((_featuresIndexes[ids[i]] != ushort.MaxValue) &&
(_features[_featuresIndexes[ids[i]]].Mx != 0))
            {
                double max =
Math.Abs(_features[_featuresIndexes[ids[i]]].GetMaximumLimit());
                double min =
Math.Abs(_features[_featuresIndexes[ids[i]]].GetMinimumLimit());
                if (max < min)
                    max = min;
                res[i] =
OperationsOfRoundingWithNumbers.GetNormalizedMantissa(_features[_
featuresIndexes[ids[i]]].Mx, max, 7, 2);
                if (_features[_featuresIndexes[ids[i]]].Mx < 0)
                    res[i] = (byte)(res[i] + 128);
            }
            else
                res[i] = 0;
        return res;
    }

    public byte[] ToBytes(ushort[] ids)
    {
        byte[] res = new byte[ids.Length];
        for (int i = 0; i < ids.Length; i++)
            if ((_featuresIndexes[ids[i]] !=
ushort.MaxValue)&&(_features[_featuresIndexes[ids[i]]].Mx!=0))
            {
                res[i] =
OperationsOfRoundingWithNumbers.GetNormalizedMantissa(_features[_
featuresIndexes[ids[i]]].Mx, _features[_featuresIndexes[ids[i]]].Mx, 7, 2);
                if (_features[_featuresIndexes[ids[i]]].Mx < 0)
                    res[i] = (byte)(res[i] + 128);
            }
            else
                res[i] = 0;
        return res;
    }

    #endregion

    #region ImplementedProperties

    public Realization[] OwnRealizations { get { return
_ownRealizations; } } // Реализации, из которых был сформирован
эталон (если они были переданы в конструктор - иначе return null)
    public IFeature[] Features { get { return _features; } } // Эталонные
описания признаков
    public String ImageClassName { get { return _imageClassName; } }
// Имя класса, к которому относится эталон
//      public bool RemainHistoryOfTraining { set {
_remainHistoryOfTraning = value; } get { return
_remainHistoryOfTraning; } } // Запоминать историю обучения
(актуально для нейронок)
```

```
        public int TypeId { get { return _typeId; } } // Идентификатор типа эталона
        public ushort MaxFeatureIdInGlobalScope { get { return _maxFeatureIdInGlobalScope; } } // Максимальный идентификатор признака среди всех эталонов


        #endregion

        #region MustBeImplemented

        // Вернуть параметры метода создания эталона (если они есть), иначе вернуть null
        abstract public double[] GetParameters();

        // Вычисляет параметры эталона с учетом Чужих
        // среднеквадратичные отклонения и мат. ожидания значений признаков всех известных образов (Чужих/Других)
        // Если обучение эталона (нейронной сети) зависит от данных других эталонов, то обучение должно происходить в теле данной функции
        // ownIndex - это индекс текущего эталона (из которого вызывается данная функция)
        abstract public void TrainingWithGlobalScopeData(IEtalon[] etalons, int ownIndex);

        #endregion
    }

    // Классификатор, возвращающий степени близости (лучше нормировать от нуля до единицы, т.е. [0; 1]) эталонов и реализаций, а также генерируемых им ключей
    abstract public class IClassificator
    {
        #region ImplementedAndVirtual

        protected IEtalon[] _etalons;
        protected FeaturesSpecification _specification;
        protected int _typeId; // Идентификатор типа классификатора
        protected List<String>[] _information_own;
        protected List<String>[] _information_stranger;
        protected String[] _comments;
        protected bool _saveHistory;
        public IClassificator(IEtalon[] etalons, FeaturesSpecification specification, int typeId, bool saveHistory)
        {
            _etalons = etalons;
            _specification = specification;
            _typeId = typeId;
            _saveHistory = saveHistory;
            if(saveHistory)
            {
                _information_own = new List<string>[etalons.Length];
                _information_stranger = new List<string>[etalons.Length];
                _comments = new string[etalons.Length];
            }
        }
        // Вернуть эталоны известных образов
        public IEtalon[] GetEtalons() { return _etalons; }
        public int TypeId { get { return _typeId; } } // Идентификатор типа классификатора
        public bool NeedSaveHistory { get { return _saveHistory; } } // Сохранить историю операций в файлы или не сохранять
        // Добавить в историю операций
        protected void AddHistory(String information, int etalonIndex, int etalonIndexRealizationFrom)
        {
            if (_saveHistory)
            {
                if (etalonIndex == etalonIndexRealizationFrom)
                    _information_own[etalonIndex].Add(information);
                else
                    _information_stranger[etalonIndex].Add(information);
            }
        }
        protected void CreateHistoryForEtalon(String comment, int etalonIndex)
        {
            if (_saveHistory)
            {
                _comments[etalonIndex] = comment;
                _information_own[etalonIndex] = null;
                _information_stranger[etalonIndex] = null;
                _information_own[etalonIndex] = new List<string>();
                _information_stranger[etalonIndex] = new List<string>();
            }
        }
        // Сохранить историю операций в файлы
        public void SaveHistory(int iter, int iter2, int gen, String expComment, String ethMethod, String decitionMethod, bool useOnlyUnknownForFAR)
        {
            if (_saveHistory)
            {
                try
                {
                    // Записываем описания параметров
                    String dir = Inviroment._dir_name_base + "\\" + gen.ToString() + "_" + iter.ToString() + "_" + iter2.ToString() + "_" + expComment + "\\";
                    Directory.CreateDirectory(dir);

                    String desc = dir + Inviroment._desc_file_name_base + ".txt";
                    String par_str = "";
                    if (useOnlyUnknownForFAR)
                        par_str = "FAR - only unknown images\r\n";
                    else
                        par_str = "FAR - all images\r\n";
                    par_str = par_str+"Эталоны: " + ethMethod + "\r\n";
                    double[] par = _etalons[0].GetParameters();
                    for (int i = 0; i < par.Length; i++)
                        par_str += par[i].ToString() + "\r\n";
                    par_str += "Классификатор: " + decitionMethod + "\r\n";
                    double[] par_ = GetParameters();
                    for (int i = 0; i < par_.Length; i++)
                        par_str += par_[i].ToString() + "\r\n";
                    File.WriteAllText(desc, par_str);

                    // Записываем комментарии
                    String comm = dir + Inviroment._comments_file_name_base + ".txt";
                    File.WriteAllLines(comm, _comments);

                    // Записываем историю эксперимента
                    String names = "";
                    for (int i = 0; i < _etalons.Length; i++)
                    {
                        names += _etalons[i].ImageClassName + "\r\n";
                        String own = dir + Inviroment._own_file_name_base + "_" + (i+1).ToString()+"_"+".txt";
                        String stranger = dir + Inviroment._stranger_file_name_base + "_" + (i+1).ToString() + ".txt";
                        File.WriteAllLines(own, _information_own[i]);
                        File.WriteAllLines(stranger, _information_stranger[i]);
                    }
                    File.WriteAllText(dir + Inviroment._classes_file_name_base + ".txt", names);
                }
                catch { }
            }
        }

        // Отдать длину генерируемого ключа в битах, соответсвующую параметрам
        virtual public long GetKeyLength(double[] parameters, int featuresQuantity, int knownImagesQuantity, int ethIndex)
        {
            throw new NotImplementedException();
        }

        #endregion

        #region MustBeImplemented

        // Установить параметры классификатора (если значение параметра равно null, то принимается стандартное значение)
```

```
        // Если параметры некорректны, можно преобразовать их к наиболее близким корректным либо вызвать throw new ArgumentException()
        // Необходимо предусмотреть параметры классификатора по-умолчанию (оптимальные или просто стандартные параметры)
        abstract public void SetParameters(double?[] parameters);
        // Вернуть параметры классификатора (если они корректировались при вызове функции SetParameters, то вернуть откорректированные)
        abstract public double[] GetParameters();

        // - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

        // Функции идентификации. Если классификатор не работает в режиме идентификации, то не реализовывать и просто прописать throw new NotImplementedException();
        // Тогда _supportIdentification=false, иначе _supportIdentification=true (в SHV.Services.ClassificatorFactory)

        // Возвращает степени схожести с эталонами, вычисленные на последнем шаге алгоритма принятия решений
        abstract public double[][] Identify_lastStep(Realization realization);
        // Возвращает степени схожести с эталонами для каждого шага алгоритма принятия решений (шаги выбираются автором классификатора произвольно, например каждый признак может означать отдельный шаг)
        // Первый уровень (List) - содержит разделение данных по эталонам, второй (массив) - по признакам
        abstract public double[][] Identify_allSteps(Realization realization);

        // - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

        // Функции верификации (аутентификации). Если классификатор не работает в режиме верификации, то не реализовывать и просто прописать throw new NotImplementedException();
        // Тогда _supportVerification=false, иначе _supportVerification=true (в SHV.Services.ClassificatorFactory)

        // Возвращает степень схожести с эталоном, вычисленную на последнем шаге алгоритма принятия решений
        abstract public double[] Verify_lastStep(Realization realization, int etalonIndex, int etalonIndexRealizationFrom);
        // Возвращает степени схожести с эталоном для каждого шага алгоритма принятия решений (шаги выбираются автором классификатора произвольно, например каждый признак может означать отдельный шаг)
        abstract public double[] Verify_allSteps(Realization realization, int etalonIndex, int etalonIndexRealizationFrom);

        #endregion
    }


    // Нормированная реализация, содержащая значения только выбранных признаков в заданной последовательности
    // Предполагается, что значения одного признака в нормализованной реализации расположены последовательно (рядом), т.е. сгруппированы в виде сечений _featuresCrossSections
    // В нормализованных реализациях можно менять последовательность признаков
    public class Realization
    {
        #region PrivateFields

        private long _timeId; // Идентификатор времени получения реализации
        private double[] _allFeaturesValues; // Значения признаков в порядке следования
        private ushort[] _allFeaturesIds; // Идентификаторы признаков в порядке следования
        private String _name; // Имя реализации может произвольно изменяться, но при добавлении в класс присваивается имя класса (используется для вывода на графиках, а не в вычислениях)
        private bool _synthetic; // Флаг, если равен true, то реализация сгенерирована
        public List<int>  _partsCounters;
        public List<String> _partsNames;

        #endregion

        public Realization(double[] featuresValues, ushort[] featuresIds, long timeId, bool synthetic, List<int> partsCounters=null, List<String> partsNames=null)
        {
            if (featuresValues == null) throw new ArgumentException("Realization");
            if (featuresIds == null) throw new ArgumentException("Realization");
            _timeId = timeId;
            _allFeaturesValues = featuresValues;
            _allFeaturesIds = featuresIds;
            _synthetic = synthetic;
            _partsCounters = partsCounters;
            _partsNames = partsNames;
        }


        #region Methods

        // Вернуть последовательность всех значений всех признаков в порядке их следования в реализации
        public double[] GetAllFeaturesValues()
        {
            return _allFeaturesValues;
        }

        // Вернуть последовательность идентификаторов всех признаков в порядке их следования в реализации
        public ushort[] GetAllFeaturesIds()
        {
            return _allFeaturesIds;
        }

        // Найти и вернуть первое значение заданного признака
        public double? FindFeatureValue(ushort featureId, ref int searchPos)
        {
            for (; searchPos < _allFeaturesValues.Length; searchPos++)
                if (_allFeaturesIds[searchPos] == featureId)
                {
                    searchPos++;
                    return _allFeaturesValues[searchPos-1];
                }
            return null;
        }

        public double FindFeatureValue(ushort featureId, int searchPos, double insteedNull)
        {
            int start = searchPos;
            for (; searchPos < _allFeaturesValues.Length; searchPos++)
                if (_allFeaturesIds[searchPos] == featureId)
                {
                    searchPos++;
                    return _allFeaturesValues[searchPos - 1];
                }
            for (searchPos=0; searchPos < start; searchPos++)
                if (_allFeaturesIds[searchPos] == featureId)
                {
                    searchPos++;
                    return _allFeaturesValues[searchPos - 1];
                }
            return insteedNull;
        }

        public List<double> GetCrossSection(ushort featureId, List<double> cs)
        {
            for (int j = 0; j < _allFeaturesIds.Length; j++)
                if (featureId == _allFeaturesIds[j])
                    cs.Add(_allFeaturesValues[j]);
            return cs;
        }

        #endregion
```

```
        #region Properties

        public long TimeId { get { return _timeId; } } // Время создания реализации в произвольном виде
        public String ImageClassName { get { return _name; } set { _name = value; } } // Имя реализации может произвольно изменяться, но при добавлении в класс присваивается имя класса (используется для вывода на графиках, а не в вычислениях)
        public bool Synthetic { get { return _synthetic; } } // Флаг, если равен true, то реализация сгенерирована

        #endregion
    }


    // Хранилище классов образов и реализаций
    public interface IStorage
    {
================================================================================================================================================================
        // ИНИЦИАЛИЗАЦИЯ - работа с контейнерами данных
        bool SetDatasetIfItIsCorrect(String resurceName); // Устанавливает контейнер с данными. Возвращает true, если контейнер существует и имеет правильный формат и false - если нет
        String GetDatasetName(); // Возвращает имя активного контейнера с данными (из которого хранилище берет данные)
        FeaturesSpecification GetSpecification(); // Вернуть спецификацию - описание физического значения признаков
        // ------------------------------------------------------------------------------------------------------------------------------------------------
        // ЗАГРУЗКА КЛАССОВ
        List<ImageClass> GetImageClasses(int maxNumberOfClasses, int maxNumberOfRealizations); // при maxNumberOfClasses=0 грузятся все классы, при maxNumberOfRealizations=0 грузятся все реализации
        // ------------------------------------------------------------------------------------------------------------------------------------------------
        // ЗАГРУЗКА ШАБЛОНОВ - эталонных описаний признаков и классов (если формат не предусматривает хранение готовых шаблонов, то функции можно не реализовывать)
        List<ImagePattern> GetImagePatterns(int maxNumberOfClasses); // при maxNumberOfClasses=0 загружаются все данные
        // ------------------------------------------------------------------------------------------------------------------------------------------------
        // СОХРАНЕНИЕ (если сохранение в данный формат не требуется, то функции можно не реализовывать)
        bool CreateDataset(String resurceName, FeaturesSpecification specification, bool saveOnlyUsed); // Создает новый ресурс на основе спецификации (создание и подготовка новых контейнеров с данными)
        // СОХРАНЕНИЕ КЛАССОВ
        bool PutImageClass(ImageClass imageClass); // Положить конкретный класс образов (при успехе операции, должна вернуть true, иначе false)
        // СОХРАНЕНИЕ ШАБЛОНОВ КЛАССОВ
        bool PutImagePattern(String imageClassName, FeaturePattern[] imagePattern); // Положить шаблоны признаков класса образов (шаблон класса образов)
        //
================================================================================================================================================
    }


    // Спецификация признаков
    public class FeaturesSpecification
    {
        private ushort[] _ids; // Идентификаторы признаков
        private String[] _descriptions; // Описания признаков
        private bool[] _used; // Флаги использования признаков в эксперименте
        private ushort[] _featuresIdsIndexes; // Индексация признаков
        private ushort _maxFeatureIdInGlobalScope; // Максимальный идентификатор признака (среди всех вообще)
        private String _descriptionOfExp; // Описание эксперимента
        private int[] _typeIds; // Типы признаков (идентификатор типа: закона распределения и метода генерации значений)

        private void ReIndex()
        {
            _featuresIdsIndexes = null;
            _featuresIdsIndexes = new ushort[_maxFeatureIdInGlobalScope+1];
            for (ushort i = 0; i < _maxFeatureIdInGlobalScope; i++)
                _featuresIdsIndexes[i] = ushort.MaxValue;
            for (ushort i = 0; i < _ids.Length; i++)
                _featuresIdsIndexes[_ids[i]] = i;
        }

        public FeaturesSpecification(ushort[] ids, String[] descriptions, bool[] used, String descriptionOfExp)
        {
            if ((ids == null) || (ids.Length < 1) || (descriptions == null) || (descriptions.Length < 1) || (ids.Length != descriptions.Length))
                throw new ArgumentException();
            _ids = ids;
            _descriptions = descriptions;
            if ((used != null) || (used.Length == ids.Length) || (used.Length == descriptions.Length))
                _used = used;
            else
            {
                _used = new bool[ids.Length];
                for (int i = 0; i < ids.Length; i++)
                    _used[i] = true;
            }
            _descriptionOfExp = descriptionOfExp;
            _maxFeatureIdInGlobalScope = Statistica.MaxValue(ids);
            ReIndex();
            _typeIds = new int[ids.Length];
            for (int i = 0; i < ids.Length; i++)
                _typeIds[i] = 0;
        }

        public String GetExperimentDescription()
        {
            return _descriptionOfExp;
        }

        // Дать индекс признака
        public ushort GetFeatureIndex(ushort id)
        {
            if (id > _featuresIdsIndexes.Length) return ushort.MaxValue;
            else return _featuresIdsIndexes[id];
        }

        // Дать описание признака
        public String GetFeatureDescription(ushort id)
        {
            if ((id>_featuresIdsIndexes.Length)||(_featuresIdsIndexes[id] == ushort.MaxValue)) return "";
            else return _descriptions[_featuresIdsIndexes[id]];
        }

        // Дать флаг использования признака
        public bool GetUsed(ushort id)
        {
            if ((id > _featuresIdsIndexes.Length) || (_featuresIdsIndexes[id] == ushort.MaxValue)) return false;
            else return _used[_featuresIdsIndexes[id]];
        }

        // Установить флаг использования признака
        public bool SetUsed(ushort id, bool usedFlag)
        {
            if ((id > _featuresIdsIndexes.Length) || (_featuresIdsIndexes[id] == ushort.MaxValue)) return false;
            else _used[_featuresIdsIndexes[id]] = usedFlag;
            return true;
        }
```

```
    // Установить флаг использования признака
    public void SetUsedAll(bool usedFlag)
    {
       for (int i = 0; i < _used.Length; i++)
          _used[i] = usedFlag;
    }

    // Дать максимальный идентификатор признака (среди всех
вообще)
    public ushort GetMaxFeatureId()
    {
       return _maxFeatureIdInGlobalScope;
    }

    // Вернуть все идентификаторы
    public ushort[] GetFeaturesIds()
    {
       return _ids;
    }

    // Вернуть все описания
    public String[] GetFeaturesDescriptions()
    {
       return _descriptions;
    }

    // Вернуть типы признаков
    public int[] GetTypesIds()
    {
       int[] ids = new int[_ids.Length];
       for (int i = 0; i < _ids.Length; i++)
             ids[i]=GetTypeId(_ids[i]);
       return ids;
    }

    // Вернуть идентификаторы используемых признаков
    public ushort[] GetUsedFeaturesIds()
    {
       List<ushort> ids = new List<ushort>();
       for (int i = 0; i < _ids.Length; i++)
          if (_used[i])
             ids.Add(_ids[i]);
       return ids.ToArray();
    }

    // Вернуть типы первых нескольких признаков
    public ushort[] GetFirstFeaturesIds(int featuresCount)
    {
       if (featuresCount > _ids.Length)
          featuresCount = _ids.Length;
       ushort[] ids = new ushort[featuresCount];
       for (int i = 0; i < ids.Length; i++)
          ids[i] = _ids[i];
       return ids;
    }

    // Вернуть описания используемых признаков
    public String[] GetUsedFeaturesDescriptions()
    {
       List<String> descriptions = new List<string>();
       for (int i = 0; i < _descriptions.Length; i++)
          if (_used[i])
             descriptions.Add(_descriptions[i]);
       return descriptions.ToArray();
    }

    // Вернуть типы используемых признаков
    public int[] GetUsedTypesIds()
    {
       List<int> ids = new List<int>();
       for (int i = 0; i < _ids.Length; i++)
          if (_used[i])
             ids.Add(GetTypeId(_ids[i]));
       return ids.ToArray();
    }

    // Вернуть тип признака
    public int GetTypeId(ushort id)
    {
       return _typeIds[_featuresIdsIndexes[id]];
    }

    // Установить тип признака
    public bool SetTypeId(ushort id, int typeId)
    {
       if ((id > _featuresIdsIndexes.Length) || (_featuresIdsIndexes[id] ==
ushort.MaxValue)) return false;
       _typeIds[_featuresIdsIndexes[id]] = typeId;
       return true;
    }

    // Количество различных признаков в глобальном пространстве
    public ushort FeaturesCountInGlobalScope { get { return
(ushort)_ids.Length; } }
  }


  static public class Manager
  {
    #region Features_MetaSpace

    public static Realization GetRealizationInMetaSpace(Realization
realization, ushort[] uniqueFeaturesIds, int normalize, bool
mixWithOriginalFeatures, double[] mx, double[] sx, int n, ushort[]
new_ids, double p, ushort[] allUniqueFeaturesIds)
    {
       double[] values = null;
       if (normalize==1)
          values =
Manager.GetVectorOfNormalizedUniqueFeaturesValues(realization,
uniqueFeaturesIds, 0, sx, p);
       else
       if (normalize == 2)
          values =
Manager.GetVectorOfNormalizedUniqueFeaturesValues(realization,
uniqueFeaturesIds, 0, mx, sx, p);
       else
       values = Manager.GetVectorOfUniqueFeaturesValues(realization,
uniqueFeaturesIds, 0, false);
       double[] valueNew = new double[n];
       int count = 0;
       for (int i = 0; i < values.Length; i++)
          for (int j = i + 1; j < values.Length; j++)
          {
             valueNew[count] = Math.Abs(values[i] - values[j]);
             count++;
          }
       if (mixWithOriginalFeatures)
       {
          values =
Manager.GetVectorOfUniqueFeaturesValues(realization,
allUniqueFeaturesIds, 0, false);
          for (int i = 0; i < values.Length; i++)
          {
             valueNew[count] = values[i];
             count++;
          }
       }
       return new Realization(valueNew, new_ids, realization.TimeId,
realization.Synthetic);
    }

    #endregion


    #region Information

    static private bool CompareEvaluateFRR(double min, double max,
double st, bool distanceMeasure)
    {
       if (distanceMeasure)
       {
          if (st < max)
             return true;
          else
```

```
            return false;
        }
        else
        {
            if (st > min)
                return true;
            else
                return false;
        }
    }

    static private bool CompareEvaluateFAR(double min, double max,
double st, bool distanceMeasure)
    {
        if (!distanceMeasure)
        {
            if (st < max)
                return true;
            else
                return false;
        }
        else
        {
            if (st > min)
                return true;
            else
                return false;
        }
    }

    // Измерение вероятностей ошибок 1-ого рода в зависимости от
ошибок 2-ого рода
    static public double[] EvaluateFRR_bySquare(IFeature own, IFeature
stranger, bool entropy, double entropyUnit, long FARaccuracy, bool
distanceMeasure)
    {
        if (FARaccuracy < 10) throw new ArgumentException();
        long accuracy = FARaccuracy * 100;
        double[] res = new double[FARaccuracy];
        double probStep = (double)1 / FARaccuracy;
        double min = own.GetMinimumLimit();
        double max = own.GetMaximumLimit();
        double min2 = stranger.GetMinimumLimit();
        double max2 = stranger.GetMaximumLimit();
        if (min > min2) min = min2;
        if (max < max2) max = max2;
        double s = 0;
        double o = 0;
        double step = (max - min) / accuracy;
        if (step == 0)
            throw new ArgumentException();
        int counter = 0;
        int count = -1;
        double st = max;
        if (distanceMeasure)
            st = min;
        while (CompareEvaluateFRR(min, max, st, distanceMeasure))
        {
            double t_o = own.GetDensityOfProb(st);
            double t_s = stranger.GetDensityOfProb(st);
            if (!Double.IsNaN(t_o))
                o += t_o * step;
            if (!Double.IsNaN(t_s))
                s += t_s * step;
            if ((s > probStep * counter) && (s < probStep * (counter + 1)))
                count++;
            while ((s > probStep * counter) && (s < probStep * (counter +
1)))
            {
                res[count] = 1 - o;
                if (entropy)
                    res[count] = -Math.Log(1 - o, entropyUnit);
                counter++;
                if (counter >= res.Length)
                    return res;
            }
            if (distanceMeasure)
                st += step;
            else
                st -= step;
        }
        return res;
    }

    // Измерение вероятностей ошибок 2-ого рода в зависимости от
ошибок 1-ого рода
    static public double[] EvaluateFAR_bySquare(IFeature own, IFeature
stranger, bool entropy, double entropyUnit, long FARaccuracy, bool
distanceMeasure)
    {
        if (FARaccuracy < 10) throw new ArgumentException();
        long accuracy = FARaccuracy * 100;
        double[] res = new double[FARaccuracy];
        double probStep = (double)1 / FARaccuracy;
        double min = own.GetMinimumLimit();
        double max = own.GetMaximumLimit();
        double min2 = stranger.GetMinimumLimit();
        double max2 = stranger.GetMaximumLimit();
        if (min > min2) min = min2;
        if (max < max2) max = max2;
        double s = 0;
        double o = 0;
        double step = (max - min) / accuracy;
        if (step == 0)
            throw new ArgumentException();
        int counter = 0;
        int count = -1;
        double st = min;
        if (distanceMeasure)
            st = max;
        while (CompareEvaluateFAR(min, max, st, distanceMeasure))
        {
            double t_o = own.GetDensityOfProb(st);
            double t_s = stranger.GetDensityOfProb(st);
            if (!Double.IsNaN(t_o))
                o += t_o * step;
            if (!Double.IsNaN(t_s))
                s += t_s * step;
            if ((o > probStep * counter) && (o < probStep * (counter + 1)))
                count++;
            while ((o > probStep * counter) && (o < probStep * (counter +
1)))
            {
                res[count] = 1 - s;
                if (entropy)
                    res[count] = -Math.Log(1 - s, entropyUnit);
                counter++;
                if (counter >= res.Length)
                    return res;
            }
            if (distanceMeasure)
                st -= step;
            else
                st += step;
        }
        return res;
    }

    static public double EvaluateSquare(IFeature ft, IFeature ft2, uint
accuracy, bool bits)
    {
        double min = ft.GetMinimumLimit();
        double max = ft.GetMaximumLimit();
        double min2 = ft2.GetMinimumLimit();
        double max2 = ft2.GetMaximumLimit();
        if (min > min2) min = min2;
        if (max < max2) max = max2;
        double e = 0;
        double step = (max - min) / accuracy;
        if (step == 0)
            return 1;
        double st = min;
        int counter = 0;
        while ((st < max) && (counter < 10000))
        {
            double t_i = ft.GetDensityOfProb(st);
```

```
            double t_j = ft2.GetDensityOfProb(st);
            double T1 = t_j > t_i ? t_i : t_j;
            if ((!Double.IsNaN(T1)) && (!Double.IsInfinity(T1)))
               e += T1 * step;
            st += step;
            counter++;
         }
         // Проверка на ноль
         if (e == 0) e = e + ((double)1 / (100 * accuracy));
         if (bits) e = -Math.Log(e, 2);
         return e;
      }

      // Расчет ошибок первого и второго рода
      static public double[] EvaluateABE(IFeature[] ft, uint accuracy, bool
log)
      {
         double T, T1;
         List<double> E = new List<double>();
         double[] min = new double[ft.Length];
         double[] max = new double[ft.Length];
         for (int i = 0; i < ft.Length; i++)
            if (ft[i] != null)
            {
               min[i] = ft[i].GetMinimumLimit();
               max[i] = ft[i].GetMaximumLimit();
            }

         // Найти все пересекающиеся области классов и
просуммировать их
         for (int i = 0; i < ft.Length; i++) // Рассмотрим последовательно
каждую плотность распределения вероятности
            if (ft[i] != null)
            {
               double e = 0;
               double step = (max[i] - min[i]) / accuracy;
               for (double st = min[i]; st < max[i]; st += step) // Рассмотрим
область от -3сигма до +3сигма
               {
                  T = 0;
                  for (int j = 0; j < ft.Length; j++) // Рассмотрим следующее
за i распределение j (i+1)
                     if (ft[j] != null)
                        if (j != i && st >= min[j] && st <= max[j])
                        {
                           double t_i = ft[i].GetDensityOfProb(st);
                           double t_j = ft[j].GetDensityOfProb(st);
                           T1 = t_j > t_i ? t_i : t_j;
                           if (T1 > T) T = T1;
                        }
                  e += T * step;
               }
               // Проверка на ноль
               if (e == 0) e = e + ((double)1 / (100 * accuracy));
               if (log) e = -Math.Log(e, 2);
               E.Add(e);
            }
            else
            {
               if (log) E.Add(0);
               else
                  E.Add(1);
            }
         return E.ToArray();
      }

      static public double[] EvaluateABE_midlePair(IFeature[] ft, uint
accuracy, bool log)
      {
         List<double> E = new List<double>();
         double[] min = new double[ft.Length];
         double[] max = new double[ft.Length];
         for (int i = 0; i < ft.Length; i++)
            if (ft[i] != null)
            {
               min[i] = ft[i].GetMinimumLimit();
               max[i] = ft[i].GetMaximumLimit();
            }

         // Найти все пересекающиеся области классов и
просуммировать их
         for (int i = 0; i < ft.Length; i++) // Рассмотрим последовательно
каждую плотность распределения вероятности
            if (ft[i] != null)
            {
               double e = 0;
               double step = (max[i] - min[i]) / accuracy;
               for (double st = min[i]; st < max[i]; st += step) // Рассмотрим
область от -3сигма до +3сигма
               {
                  List<double> T = new List<double>();
                  for (int j = 0; j < ft.Length; j++) // Рассмотрим следующее
за i распределение j (i+1)
                     if (ft[j] != null)
                        if (j != i && st >= min[j] && st <= max[j])
                        {
                           double t_i = ft[i].GetDensityOfProb(st);
                           double t_j = ft[j].GetDensityOfProb(st);
                           double T1 = t_j > t_i ? t_i : t_j;
                           if ((!Double.IsNaN(T1)) &&
(!Double.IsInfinity(T1))) T.Add(T1); ;
                        }
                  if (T.Count > 0)
                     e += Statistica.Mx_a(T.ToArray(), T.Count) * step;
               }
               // Проверка на ноль
               if (e == 0) e = e + ((double)1 / (100 * accuracy));
               if (log) e = -Math.Log(e, 2);
               E.Add(e);
            }
            else
            {
               if (log) E.Add(0);
               else
                  E.Add(1);
            }

         return E.ToArray();
      }

      static public double[] EvaluateABE_ownOrAlien(IFeature[] ft,
IFeature[] ftAlien, uint accuracy, bool log)
      {
         List<double> E = new List<double>();
         double[] min = new double[ft.Length];
         double[] max = new double[ft.Length];
         for (int i = 0; i < ft.Length; i++)
            if (ft[i] != null)
            {
               min[i] = ft[i].GetMinimumLimit();
               double min_tmp = ftAlien[i].GetMinimumLimit();
               if (min_tmp < min[i]) min[i] = min_tmp;
               max[i] = ft[i].GetMaximumLimit();
               double max_tmp = ftAlien[i].GetMaximumLimit();
               if (max_tmp > max[i]) max[i] = max_tmp;
            }

         // Найти все пересекающиеся области классов и
просуммировать их
         for (int i = 0; i < ft.Length; i++) // Рассмотрим последовательно
каждую плотность распределения вероятности
            if (ft[i] != null)
            {
               if (ftAlien[i] != null)
               {
                  double e = 0;
                  double step = (max[i] - min[i]) / accuracy;
                  for (double st = min[i]; st < max[i]; st += step) //
Рассмотрим область от -3сигма до +3сигма
                  {
                     double t_i = ft[i].GetDensityOfProb(st);
                     double t_a = ftAlien[i].GetDensityOfProb(st);
                     double T1 = t_a > t_i ? t_i : t_a;
                     if ((!Double.IsNaN(T1)) && (!Double.IsInfinity(T1)))
                        e += T1 * step;
                  }
```

```
                // Проверка на ноль
                if (e == 0) e = e + ((double)1 / (100 * accuracy));
                if (log) e = -Math.Log(e, 2);
                E.Add(e);
            }
            else
            {
                if (log) E.Add(0);
                else
                    E.Add(1);
            }
        }
        else
        {
            if (log) E.Add(0);
            else
                E.Add(1);
        }

    return E.ToArray();
}

#endregion


#region Corralation

    // Вычисление коэффициентов парной корреляции между
признаками
    static public double[][] CalcCorrelationMatrix(IEtalon etalon)
{
    double[][] res = new double[etalon.Features.Length][];
    for (ushort i = 0; i < etalon.Features.Length; i++)
        res[i] = new double[etalon.Features.Length];
    for (ushort i = 0; i < etalon.Features.Length; i++)
    {
        res[i][i] = 1;
        for (ushort j = (ushort)(i + 1); j < etalon.Features.Length; j++)
        {
            List<double>[] cs =
etalon.Get2CoherentOwnMiddleCrossSections(etalon.Features[i].Id,
etalon.Features[j].Id);
            if ((cs[0].Count > 2) && (cs[0].Count == cs[1].Count))
            {
                res[i][j] = Statistica.CoefCor_a(cs[0].ToArray(),
cs[1].ToArray(), cs[0].Count);
                res[j][i] = res[i][j];
            }
        }
    }
    return res;
}

static public double[][] CalcCorrelationMatrix(double[][] cs)
{
    double[][] res = new double[cs.Length][];
    for (ushort i = 0; i < cs.Length; i++)
        res[i] = new double[cs.Length];
    for (ushort i = 0; i < cs.Length; i++)
    {
        res[i][i] = 1;
        for (ushort j = (ushort)(i + 1); j < cs.Length; j++)
        {
            res[i][j] = Statistica.CoefCor_a(cs[j], cs[i], cs[j].Length);
            res[j][i] = res[i][j];
        }
    }
    return res;
}

// Вычисление коэффициентов парной корреляции между
признаками
static public double[][][] CalcCorrelationMatrix(IEtalon[] etalons)
{
    double[][][] res = new double[etalons.Length][][];
    for (int k = 0; k < etalons.Length; k++)
        res[k] = CalcCorrelationMatrix(etalons[k]);
    return res;
}

#endregion


#region Realizations_CrossSections_Etalons

static public FeatureCrossSection[] GetCrossSections(Realization
realization, ushort[] uniqueFeaturesIds)
{
    FeatureCrossSection[] res = new
FeatureCrossSection[uniqueFeaturesIds.Length];
    for (int i = 0; i < uniqueFeaturesIds.Length; i++)
    {
        FeatureCrossSection cs;
        cs._id = uniqueFeaturesIds[i];
        List<double> values = new List<double>();
        int k = 0;
        double? value = realization.FindFeatureValue(cs._id, ref k);
        while (value != null)
        {
            values.Add((double)value);
            value = realization.FindFeatureValue(cs._id, ref k);
        }
        cs._values = null;
        if (values.Count > 0)
            cs._values = values.ToArray();
        res[i] = cs;
    }
    return res;
}

static public double[][] GetCrossSections(double[][] realizations)
{
    double[][] res = new double[realizations[0].Length][];
    for (int j = 0; j < realizations[0].Length; j++)
    {
        res[j] = new double[realizations.Length];
        for (int i = 0; i < realizations.Length; i++)
            res[j][i] = realizations[i][j];
    }
    return res;
}

static public List<double> GetCrossSection(List<Realization>
realizations, ushort id)
{
    List<double> res = new List<double>();
    for (int i = 0; i < realizations.Count; i++)
        res = realizations[i].GetCrossSection(id, res);
    return res;
}

static public List<double> GetCrossSection(Realization[]
realizations, ushort id)
{
    List<double> res = new List<double>();
    for (int i = 0; i < realizations.Length; i++)
        res = realizations[i].GetCrossSection(id, res);
    return res;
}

// Создать зависимость признаков в реализациях (если
inAscending=true, то значения
static public Realization[]
CreateDependancyRealizations(Realization[] realizations, ushort[] ids,
bool directAndInverseCorrelation)
{
    Realization[] res = new Realization[realizations.Length];
    List<List<int>> allCsSizes = new List<List<int>>();
    List<double[]> AllCsSorted = new List<double[]>();
    for (int j = 0; j < realizations.Length; j++)
    {
        allCsSizes.Add(new List<int>());
        for (int i = 0; i < ids.Length; i++)
            allCsSizes[j].Add(0);
    }
    bool dirrectInverse = true;
```

```
            for (int i = 0; i < ids.Length; i++)
            {
                List<List<double>> cs = new List<List<double>>();
                for (int j = 0; j < realizations.Length; j++)
                {
                    cs.Add(realizations[j].GetCrossSection(ids[i], new
List<double>()));
                    allCsSizes[j][i] = cs[j].Count;
                }
                List<double> csTmp = new List<double>();
                for (int j = 0; j < cs.Count; j++)
                    for (int k = 0; k < cs[j].Count; k++)
                        csTmp.Add(cs[j][k]);
                double[] csSorted = csTmp.ToArray();
                Array.Sort(csSorted);
                dirrectInverse = !dirrectInverse;
                if (directAndInverseCorrelation)
                    if (dirrectInverse)
                        Array.Reverse(csSorted);
                AllCsSorted.Add(csSorted);
            }

            int[] count = new int[ids.Length];
            for (int i = 0; i < ids.Length; i++)
                count[i] = 0;
            int[] count2 = new int[realizations.Length];
            for (int i = 0; i < realizations.Length; i++)
                count2[i] = 0;
            double[][] newValues = new double[realizations.Length][];
            ushort[][] newIds = new ushort[realizations.Length][];
            for (int j = 0; j < realizations.Length; j++)
            {
                newValues[j] = new
double[realizations[j].GetAllFeaturesValues().Length];
                newIds[j] = new
ushort[realizations[j].GetAllFeaturesValues().Length];
            }

            for (int i = 0; i < ids.Length; i++)
                for (int j = 0; j < realizations.Length; j++)
                    for (int k = 0; k < allCsSizes[j][i]; k++)
                    {
                        newValues[j][count2[j]] = AllCsSorted[i][count[i]];
                        newIds[j][count2[j]] = ids[i];
                        count[i]++;
                        count2[j]++;
                    }

            for (int j = 0; j < realizations.Length; j++)
            {
                res[j] = new Realization(newValues[j], newIds[j],
realizations[j].TimeId, realizations[j].Synthetic);
                res[j].ImageClassName = realizations[j].ImageClassName;
            }
            return res;
        }

        static public List<double> GetFeaturesValues(Realization real,
ushort[] uniqueFeaturesIds)
        {
            List<double> res = new List<double>();
            ushort[] allFeaturesIds = real.GetAllFeaturesIds();
            double[] allFeaturesValues = real.GetAllFeaturesValues();
            for (int i = 0; i < uniqueFeaturesIds.Length; i++)
            {
                for (int j = 0; j < allFeaturesIds.Length; j++)
                    if (uniqueFeaturesIds[i] == allFeaturesIds[j])
                        res.Add(allFeaturesValues[j]);
            }
            return res;
        }

        // Вернуть значения уникальных признаков (каждого по одному)
        static public double?[]
GetVectorOfUniqueFeaturesValues(Realization real, ushort[]
uniqueFeaturesIds, bool mx)
        {
            double?[] vectorOfFeatures = new
double?[uniqueFeaturesIds.Length];
            ushort[] allFeaturesIds = real.GetAllFeaturesIds();
            double[] allFeaturesValues = real.GetAllFeaturesValues();
            if (mx)
                for (int i = 0; i < uniqueFeaturesIds.Length; i++)
                {
                    vectorOfFeatures[i] = null;
                    int count = 0;
                    for (int j = 0; j < allFeaturesIds.Length; j++)
                        if (uniqueFeaturesIds[i] == allFeaturesIds[j])
                        {
                            count++;
                            if (count == 1)
                                vectorOfFeatures[i] = allFeaturesValues[j];
                            else
                                vectorOfFeatures[i] =
Statistica.Mx_a_rct((double)vectorOfFeatures[i], allFeaturesValues[j],
count);
                        }
                }
            else
                for (int i = 0; i < uniqueFeaturesIds.Length; i++)
                {
                    vectorOfFeatures[i] = null;
                    for (int j = 0; j < allFeaturesIds.Length; j++)
                        if (uniqueFeaturesIds[i] == allFeaturesIds[j])
                        {
                            vectorOfFeatures[i] = allFeaturesValues[j];
                            break;
                        }
                }
            return vectorOfFeatures;
        }
        // При mx=true возвращает среднее значение каждого признака,
иначе первое встречное
        static public double[] GetVectorOfUniqueFeaturesValues(Realization
real, ushort[] uniqueFeaturesIds, double insteadNull, bool mx)
        {
            double[] vectorOfFeatures = new
double[uniqueFeaturesIds.Length];
            ushort[] allFeaturesIds = real.GetAllFeaturesIds();
            double[] allFeaturesValues = real.GetAllFeaturesValues();
            if (mx)
                for (int i = 0; i < uniqueFeaturesIds.Length; i++)
                {
                    vectorOfFeatures[i] = insteadNull;
                    int count = 0;
                    for (int j = 0; j < allFeaturesIds.Length; j++)
                        if (uniqueFeaturesIds[i] == allFeaturesIds[j])
                        {
                            count++;
                            if (count == 1)
                                vectorOfFeatures[i] = allFeaturesValues[j];
                            else
                                vectorOfFeatures[i] =
Statistica.Mx_a_rct(vectorOfFeatures[i], allFeaturesValues[j], count);
                        }
                }
            else
                for (int i = 0; i < uniqueFeaturesIds.Length; i++)
                {
                    vectorOfFeatures[i] = insteadNull;
                    for (int j = 0; j < allFeaturesIds.Length; j++)
                        if (uniqueFeaturesIds[i] == allFeaturesIds[j])
                        {
                            vectorOfFeatures[i] = allFeaturesValues[j];
                            break;
                        }
                }
            return vectorOfFeatures;
        }
        // При mx=true возвращает среднее значение каждого признака,
иначе первое встречное
        static public double[]
GetVectorOfNormalizedUniqueFeaturesValues(Realization real, ushort[]
uniqueFeaturesIds, double insteadNull, bool mx, IEtalon stranger_etalon)
        {
```

```
        double[] vectorOfFeatures = new
double[uniqueFeaturesIds.Length];
        ushort[] allFeaturesIds = real.GetAllFeaturesIds();
        double[] allFeaturesValues = real.GetAllFeaturesValues();
        if (mx)
            for (int i = 0; i < uniqueFeaturesIds.Length; i++)
            {
                vectorOfFeatures[i] = insteadNull;
                int count = 0;
                for (int j = 0; j < allFeaturesIds.Length; j++)
                    if (uniqueFeaturesIds[i] == allFeaturesIds[j])
                    {
                        count++;
                        if (count == 1)
                            vectorOfFeatures[i] = allFeaturesValues[j];
                        else
                            vectorOfFeatures[i] =
Statistica.Mx_a_rct(vectorOfFeatures[i], allFeaturesValues[j], count);
                        IFeature feature =
stranger_etalon.GetFeature(allFeaturesIds[j]);
                        vectorOfFeatures[i] = (vectorOfFeatures[i] - feature.Mx)
/ feature.Sx;
                    }
            }
        else
            for (int i = 0; i < uniqueFeaturesIds.Length; i++)
            {
                vectorOfFeatures[i] = insteadNull;
                for (int j = 0; j < allFeaturesIds.Length; j++)
                    if (uniqueFeaturesIds[i] == allFeaturesIds[j])
                    {
                        IFeature feature =
stranger_etalon.GetFeature(allFeaturesIds[j]);
                        vectorOfFeatures[i] = (allFeaturesValues[j] - feature.Mx)
/ feature.Sx;
                        break;
                    }
            }
        return vectorOfFeatures;
    }
    static public double[]
GetVectorOfNormalizedUniqueFeaturesValues(Realization real, ushort[]
uniqueFeaturesIds, double insteadNull, double[] sx, double p)
    {
        double[] vectorOfFeatures = new
double[uniqueFeaturesIds.Length];
        ushort[] allFeaturesIds = real.GetAllFeaturesIds();
        double[] allFeaturesValues = real.GetAllFeaturesValues();
        for (int i = 0; i < uniqueFeaturesIds.Length; i++)
        {
            vectorOfFeatures[i] = insteadNull;
            for (int j = 0; j < allFeaturesIds.Length; j++)
                if (uniqueFeaturesIds[i] == allFeaturesIds[j])
                {
                    vectorOfFeatures[i] =
Math.Pow(Math.Abs(allFeaturesValues[j]) / sx[i], p);
                    break;
                }
        }
        return vectorOfFeatures;
    }
    static public double[]
GetVectorOfNormalizedUniqueFeaturesValues(Realization real, ushort[]
uniqueFeaturesIds, double insteadNull, double[] mx, double[] sx, double
p)
    {
        double[] vectorOfFeatures = new
double[uniqueFeaturesIds.Length];
        ushort[] allFeaturesIds = real.GetAllFeaturesIds();
        double[] allFeaturesValues = real.GetAllFeaturesValues();
        for (int i = 0; i < uniqueFeaturesIds.Length; i++)
        {
            vectorOfFeatures[i] = insteadNull;
            for (int j = 0; j < allFeaturesIds.Length; j++)
                if (uniqueFeaturesIds[i] == allFeaturesIds[j])
                {
                    vectorOfFeatures[i] = Math.Pow(Math.Abs(mx[j]-
allFeaturesValues[j]) / sx[i], p);
                    break;
                }
        }
        return vectorOfFeatures;
    }

    // Вернуть реализацию, подготовленную к эксперименту (если
необходимо изменить последовательность признаков и/или убрать
неиспользуемые признаки)
    static public Realization GetPreparedRealization(Realization real,
ushort[] uniqueFeaturesIds, bool originalPos)
    {
        int count = 0;
        ushort[] allFeaturesIds = real.GetAllFeaturesIds();
        double[] allFeaturesValues = real.GetAllFeaturesValues();
        double[] vals = new double[allFeaturesValues.Length];
        ushort[] ids = new ushort[allFeaturesValues.Length];
        if (originalPos)
        {
            for (int i = 0; i < allFeaturesIds.Length; i++)
                for (int j = 0; j < uniqueFeaturesIds.Length; j++)
                    if (allFeaturesIds[i] == uniqueFeaturesIds[j])
                    {
                        vals[count] = allFeaturesValues[i];
                        ids[count] = allFeaturesIds[i];
                        count++;
                        break;
                    }
        }
        else
        {
            for (int j = 0; j < uniqueFeaturesIds.Length; j++)
                for (int i = 0; i < allFeaturesIds.Length; i++)
                    if (allFeaturesIds[i] == uniqueFeaturesIds[j])
                    {
                        vals[count] = allFeaturesValues[i];
                        ids[count] = allFeaturesIds[i];
                        count++;
                    }
        }
        if (count == 0) return null;

        double[] vals_ = new double[count];
        ushort[] ids_ = new ushort[count];
        for (int i = 0; i < count; i++)
        {
            vals_[i] = vals[i];
            ids_[i] = ids[i];
        }
        vals = null;
        ids = null;
        return new Realization(vals_, ids_, real.TimeId, real.Synthetic);
    }

    #endregion


    #region MixRealizations

    // Скрещевание реализаций
    static public Realization MixRealizations(Realization real1,
Realization real2, int weight_real1, int weight_real2)
    {
        double[] values1 = real1.GetAllFeaturesValues();
        ushort[] ids1 = real1.GetAllFeaturesIds();
        double[] values2 = real2.GetAllFeaturesValues();
        ushort[] ids2 = real2.GetAllFeaturesIds();
        bool[] was = new bool[values2.Length];
        double[] values = new double[values1.Length];
        double w = weight_real1 + weight_real2;
        double w1 = (double)weight_real1 / w;
        double w2 = (double)weight_real2 / w;
        for (int i = 0; i < ids1.Length; i++)
        {
            values[i] = values1[i];
            for (int j = 0; j < ids2.Length; j++)
                if ((ids1[i] == ids2[j]) && (!was[j]))
```

```
                { values[i] = w1 * values1[i] + w2 * values2[j]; was[j] = true;
break; }
            }
            long timeId = real1.TimeId;
            if (timeId < real2.TimeId) timeId = real2.TimeId;
            timeId++;
            Realization real = new Realization(values, ids1, timeId, true);
            return real;
        }

        static public List<Realization>
CreateNextRealizationsGeneration(List<Realization> reals, int
everyPairMixCount, bool addOriginal)
        {
            List<Realization> res = new List<Realization>();
            for (int i = 0; i < reals.Count; i++)
                for (int j = i + 1; j < reals.Count; j++)
                    for (int k = 1; k <= everyPairMixCount; k++)
                        res.Add(MixRealizations(reals[i], reals[j], k,
everyPairMixCount + 1 - k));
            if (addOriginal)
                for (int i = 0; i < reals.Count; i++)
                    res.Add(reals[i]);
            return res;
        }

        static public Realization[]
CreateNextRealizationsGeneration(Realization[] reals, int
everyPairMixCount, bool addOriginal)
        {
            int quantity = ((reals.Length * reals.Length - reals.Length) / 2) *
everyPairMixCount;
            if (addOriginal)
                quantity = quantity + reals.Length;
            Realization[] res = new Realization[quantity];
            int count = 0;
            for (int i = 0; i < reals.Length; i++)
                for (int j = i + 1; j < reals.Length; j++)
                    for (int k = 1; k <= everyPairMixCount; k++)
                    {
                        res[count] = MixRealizations(reals[i], reals[j], k,
everyPairMixCount + 1 - k);
                        count++;
                    }
            if (addOriginal)
                for (int i = 0; i < reals.Length; i++)
                {
                    res[count] = reals[i];
                    count++;
                }
            return res;
        }

        static public Realization[]
CreateNextRealizationsGeneration(Realization[] reals, int
strongStrangersCount, int everyPairMixCount, bool addOriginal)
        {
            int quantity = ((strongStrangersCount * strongStrangersCount -
strongStrangersCount) / 2) * everyPairMixCount;
            if (addOriginal)
                quantity = quantity + reals.Length;
            Realization[] res = new Realization[quantity];
            int count = 0;
            for (int i = 0; i < strongStrangersCount; i++)
                for (int j = i + 1; j < strongStrangersCount; j++)
                    for (int k = 1; k <= everyPairMixCount; k++)
                    {
                        res[count] = MixRealizations(reals[i], reals[j], k,
everyPairMixCount + 1 - k);
                        count++;
                    }
            if (addOriginal)
                for (int i = 0; i < reals.Length; i++)
                {
                    res[count] = reals[i];
                    count++;
                }
            return res;
        }

        static public List<Realization>
CreateNextRealizationsGeneration(List<Realization> reals, int
strongStrangersCount, int generationSize, bool addOriginal)
        {
            List<Realization> res = new List<Realization>();
            int pairs = (strongStrangersCount * strongStrangersCount -
strongStrangersCount) / 2;
            int everyPairMixCount = generationSize / pairs;
            if (strongStrangersCount > reals.Count) strongStrangersCount =
reals.Count;
            if (generationSize % pairs == 0)
            {
                for (int i = 0; i < strongStrangersCount; i++)
                    for (int j = i + 1; j < strongStrangersCount; j++)
                        for (int k = 1; k <= everyPairMixCount; k++)
                            res.Add(Manager.MixRealizations(reals[i], reals[j], k,
everyPairMixCount + 1 - k));
            }
            else
            {
                everyPairMixCount++;
                for (int i = 0; i < strongStrangersCount; i++)
                    for (int j = i + 1; j < strongStrangersCount; j++)
                        for (int k = 1; k <= everyPairMixCount; k++)
                            if (res.Count < generationSize)
                                res.Add(Manager.MixRealizations(reals[i], reals[j], k,
everyPairMixCount + 1 - k));

            }
            if (addOriginal)
            {
                for (int i = 0; i < res.Count; i++)
                    reals.Add(res[i]);
                return reals;
            }
            return res;
        }

        #endregion


        #region ImagesAndRealization

        // Предобработка орбразов - нормировки
        static private List<List<double>> PrepareImageFunctions(Image
image, PrepareImageParameters parameters, bool
imageFunctionsMustBeNormalized, int dimensionIndex)
        {
            List<List<double>> res = new List<List<double>>();
            if (parameters.inverseFunctions)
            {
                if (parameters.needAvarege)

res.Add(Statistica.Average(image._functions[dimensionIndex],
parameters.avaregeFuncIndexes));
                if (parameters.needDistance)

res.Add(Statistica.Distance_n_space(image._functions[dimensionIndex],
parameters.distancesFuncIndexes));
                for (int i = parameters.derivativesFuncIndexes.Count - 1; i >= 0;
i--)

res.Add(Statistica.Derivative(image._functions[dimensionIndex][parameters.derivativesFuncIndexes[i]]));
                for (int i = parameters.initialAuncIndexes.Count - 1; i >= 0; i--)

res.Add(image._functions[dimensionIndex][parameters.initialAuncIndexes[i]]);
            }
            else
            {
                for (int i = 0; i < parameters.initialAuncIndexes.Count; i++)

res.Add(image._functions[dimensionIndex][parameters.initialAuncIndexes[i]]);
                for (int i = 0; i < parameters.derivativesFuncIndexes.Count; i++)
```

```
res.Add(Statistica.Derivative(image._functions[dimensionIndex][parameters.derivativesFuncIndexes[i]]));
            if (parameters.needDistance)

res.Add(Statistica.Distance_n_space(image._functions[dimensionIndex], parameters.distancesFuncIndexes));
            if (parameters.needAvarege)

res.Add(Statistica.Average(image._functions[dimensionIndex], parameters.avaregeFuncIndexes));
        }
        if ((parameters.needNormalizeEnergy) || parameters.clip || parameters.needNormalizeStart)
            for (int i = 0; i < res.Count; i++)
            {
                double energy = SignalsAnalisis.GetEnergy(res[i]);
                if (parameters.needNormalizeEnergy && parameters.needNormalizeStart)
                    res[i] = SignalsAnalisis.NormalizeByEnergyAndLevel(res[i], ref energy, parameters.newStartValue);
                else
                if (parameters.needNormalizeEnergy)
                    res[i] = SignalsAnalisis.NormalizeByEnergy(res[i], ref energy);
                else
                if (parameters.needNormalizeStart)
                    res[i] = SignalsAnalisis.NormalizeByLevel(res[i], ref energy, parameters.newStartValue);

                if (parameters.clip)
                {
                    double mx = Statistica.Mx_a(res[i], res[i].Count);
                    if (parameters.clip_beginEnd)
                        res[i] = SignalsAnalisis.ClipSignal_begin_end(res[i], energy, parameters.sampleSize, parameters.threshold, mx);
                    else
                        res[i] = SignalsAnalisis.ClipSignal(res[i], energy, parameters.sampleSize, parameters.threshold, mx);
                }
            }
        bool wasNorm = false;
        if (parameters.needNormalize || imageFunctionsMustBeNormalized || parameters.spectrBeforeSpectrogram)
            for (int i = 0; i < res.Count; i++)
            {
                if (res[i].Count < parameters.normalizeDuration)
                {
                    while (res[i].Count < parameters.normalizeDuration)
                        res[i].Add(0);
                }
                else
                    res[i] = SignalsAnalisis.interpolation_a(res[i], parameters.normalizeDuration);
                    wasNorm = true;
            }
        if (parameters.spectrBeforeSpectrogram)
            for (int i = 0; i < res.Count; i++)
            {
                if(!wasNorm)
                {
                    int pow = (int)Math.Log(res[i].Count, 2);
                    int value = (int)Math.Pow(2, pow);
                    res[i] = SignalsAnalisis.interpolation_a(res[i], value);
                }
                res[i] = SpectralSignalAnalysis.GetWindow(res[i], parameters.window_beforeSpectrogram, 0, res[i].Count);
                res[i] = SpectralSignalAnalysis.getSpektreA_FromProcess(res[i], 0, res[i].Count, 1, false);
            }

        return res;
    }
    static private List<List<List<double>>> AddSpectrograms(Image image, int residualDimmentionIndex, int width, int step, bool firstLowFreq, int windowType, int combine, List<List<List<double>>> newFunctions,
        bool covariation, bool cor, int cor_win_size, int cor_step, bool log, bool mel, int melCoef, double freq, bool kepstr, bool diskretCosTransform, bool averageSpectr, bool smartWindows, double energyThreshold, bool freqFromHighToLow)
    {
        for (int i = 0; i < image._functions[residualDimmentionIndex].Count; i++)

newFunctions.Add(SpectralSignalAnalysis.GetSpectrogram(image._functions[residualDimmentionIndex][i], width, step, firstLowFreq, windowType, combine, covariation, cor, cor_win_size, cor_step,
            log, mel, melCoef, freq, kepstr, diskretCosTransform, averageSpectr, smartWindows, energyThreshold, freqFromHighToLow));
        return newFunctions;
    }
    static public Image ConvertImageToSpectrogram(Image image, int residualDimmentionIndex, int width, int step, bool firstLowFreq, int windowType, int windowType_spectrumBeforeSpectrogram, bool spectrumBeforeSpectrogram, int combine, bool allWindows,

        bool covariation, bool cor, int cor_win_size, int cor_step, bool log, bool mel, int melCoef, double freq, bool kepstr, bool diskretCosTransform, bool averageSpectr, bool smartWindows, double energyThreshold, bool freqFromHighToLow)
    {
        List<List<List<double>>> newFunctions = new List<List<List<double>>>();
        if (allWindows)
            for (int i = 0; i < SpectralSignalAnalysis.GetWindowTypesCount(); i++)
                newFunctions = AddSpectrograms(image, residualDimmentionIndex, width, step, firstLowFreq, i, combine, newFunctions, covariation, cor, cor_win_size, cor_step, log, mel, melCoef, freq, kepstr, diskretCosTransform, averageSpectr, smartWindows, energyThreshold, freqFromHighToLow);
        else
            newFunctions = AddSpectrograms(image, residualDimmentionIndex, width, step, firstLowFreq, windowType, combine, newFunctions, covariation, cor, cor_win_size, cor_step, log, mel, melCoef, freq, kepstr, diskretCosTransform, averageSpectr, smartWindows, energyThreshold, freqFromHighToLow);
        image._functions = newFunctions;
        return image;
    }
    static public Image PrepareImage(Image image, PrepareImageParameters parameters, bool imageFunctionsMustBeNormalized)
    {
        Image img = new Image();
        img._name = image._name;
        //          img._content = image._content;
        img._sources = image._sources;
        img._functions = new List<List<List<double>>>();
        if (parameters.allDimentions)
            for (int i = 0; i < image._functions.Count; i++)
                img._functions.Add(PrepareImageFunctions(image, parameters, imageFunctionsMustBeNormalized, i));
        else
            img._functions.Add(PrepareImageFunctions(image, parameters, imageFunctionsMustBeNormalized, parameters.dimmentionIndex));
        if ((!parameters.allDimentions) && parameters.spectrogram)
            img = ConvertImageToSpectrogram(img, parameters.dimmentionIndex, parameters.spectrogram_width, parameters.spectrogram_step, !parameters.inverseFunctions, parameters.window, parameters.window_beforeSpectrogram,
                parameters.spectrBeforeSpectrogram, parameters.spectr_combine, parameters.allWindowsTypes, parameters.covariation, parameters.cor, parameters.cor_win_size, parameters.cor_step, parameters.log, parameters.mel,
                parameters.melCoef, parameters.freqOfSampling, parameters.kepstr, parameters.diskretCosTransform, parameters.averageSpectr, parameters.smartSpectrogram, parameters.energyThresholdForSpectrogram, parameters.freqFromHighToLow);
        if (parameters.replaceInfinity)
            img = ReplaceInfinity(img);
```

```
            if (parameters.clip_samples_begin > 0 ||
parameters.clip_samples_end > 0)
                img._functions = SignalsAnalisis.ClipSamples(img._functions,
parameters.clip_samples_begin, parameters.clip_samples_end);
            if (parameters.norm_0_1)
            {
                if (parameters.norm_0_1_oneDiapazon)
                    for (int i = 0; i < img._functions.Count; i++)
                        img._functions[i] =
SignalsAnalisis.NormalizeBy_0_1(img._functions[i]);
                else
                    for (int i = 0; i < img._functions.Count; i++)
                        for (int j = 0; j < img._functions[i].Count; j++)
                            img._functions[i][j] =
SignalsAnalisis.NormalizeBy_0_1(img._functions[i][j]);
            }
            return img;
        }
        static public Image ReplaceInfinity(Image img)
        {
            for (int i = 0; i < img._functions.Count; i++)
                for (int j = 0; j < img._functions[i].Count; j++)
                    for (int k = 0; k < img._functions[i][j].Count; k++)
                        if (Double.IsInfinity(img._functions[i][j][k]) ||
Double.IsNaN(img._functions[i][j][k]))
                            img._functions[i][j][k] = 0;
            return img;
        }

        // Эта функция делает реализации для сохранения
        static public List<Realization> GetRealizations(List<Image> images,
List<ImageAnalizerTask> tasks)
        {
            if (tasks.Count == 0) return null;
            List<Realization> res = new List<Realization>();
            for (int i = 0; i < images.Count; i++)
            {
                FeaturesVector fv = new FeaturesVector();
                fv.featuresValues = new List<double>();
                fv.featuresIds = new List<ushort>();
                fv.partsCounters = new List<int>();
                fv.partsNames = new List<String>();
                ushort firstId = 1;
                try
                {
                    for (int j = 0; j < tasks.Count; j++)
                    {
                        IImageToRealizationConverter itr =
(IImageToRealizationConverter)tasks[j].imageAnalizer;
                        fv =
itr.GetFeaturesValues(Manager.PrepareImage(images[i],
tasks[j].prepareParameters, itr.ImageFunctionsMustBeNormalized()),
firstId, fv, tasks[j].analizerParameters,
tasks[j].prepareParameters.dim_name.ToArray(),
                            tasks[j].prepareParameters.func_names.ToArray(),
tasks[j].prepareParameters.dimmentionIndex);
                        firstId = fv.featuresIds[fv.featuresIds.Count - 1];
                        firstId++;
                    }
                }
                catch { continue; }
                Realization real = new Realization(fv.featuresValues.ToArray(),
fv.featuresIds.ToArray(), DateTime.Now.Ticks, false, fv.partsCounters,
fv.partsNames);
                real.ImageClassName = images[i]._name;
                res.Add(real);
            }
            return res;
        }

        // Эта функция делает реализации "на лету" для просмотра
        static public List<Realization> GetRealizations(List<Image> images,
ImageAnalizerTask task)
        {
            List<Realization> res = new List<Realization>();
            for (int i = 0; i < images.Count; i++)
            {
                FeaturesVector fv = new FeaturesVector();
                fv.featuresValues = new List<double>();
                fv.featuresIds = new List<ushort>();
                fv.partsCounters = new List<int>();
                fv.partsNames = new List<String>();
                ushort firstId = 1;
                IImageToRealizationConverter itr =
(IImageToRealizationConverter)task.imageAnalizer;
                fv = itr.GetFeaturesValues(Manager.PrepareImage(images[i],
task.prepareParameters, false), firstId, fv, task.analizerParameters,
task.prepareParameters.dim_name.ToArray(),
task.prepareParameters.func_names.ToArray(),
task.prepareParameters.dimmentionIndex);
                if (fv.featuresIds.Count == 0) continue;
                firstId = fv.featuresIds[fv.featuresIds.Count - 1];
                firstId++;
                Realization real = new Realization(fv.featuresValues.ToArray(),
fv.featuresIds.ToArray(), DateTime.Now.Ticks, false);
                real.ImageClassName = images[i]._name;
                res.Add(real);
            }
            return res;
        }

        #endregion
    }

    public delegate void BeginTask(Object owner, int
minimumProgressValue, int maximumProgressValue, int operationType);
    public delegate void ProgressTask(Object owner, int
numberOfDoneIteration, Object intermediateResult, bool success, String
className);
    public delegate void EndEtalonsCreationTask(Object owner, IEtalon[]
knownEtalons, IEtalon[] unknownEtalons);
    public delegate void EndExperimentTask(Object owner,
AllRecognizeResults results);
    public delegate void EndGenerationOfRealizationsTask(Object owner,
Realization[][] realizationsKnown, Realization[][] realizationsUnknown);
    public delegate void EndExclutionOfErrorsInRealizationsTask(Object
owner, List<List<int>> realizationsKnownIndexes, List<List<int>>
realizationsUnknownIndexes);
    public delegate void EndTask(Object owner);

    public interface IExperiment
    {
        #region Properties

        IStorage DataSource { get; set; }
        ImageClass[] KnowClasses { get; }
        ImageClass[] UnknowClasses { get; }
        ImagePattern[] KnowPatterns { get; }
        ImagePattern[] UnknowPatterns { get; }
        IEtalon[] KnowClassesEth { get; }
        IEtalon[] UnknowClassesEth { get; }
        int AllEtalonsQuantity { get; }
        FeaturesSpecification Specification { get; } // Отдать
спецификацию признаков

        #endregion


        #region Data

        // Загрузить/сохранить данные
        bool LoadData(String resourceName, int maxClasses, int maxReals);
        bool LoadData(List<ImageClass> classes, FeaturesSpecification
specification, IStorage storage);
        bool SaveData(String resourceName, bool saveOnlyUsedFeatures);
        // Создать шаблоны классов
        bool CreateImagesPatterns(List<int> knownClassesIndexes,
List<int> unknownClassesIndexes, ushort[] ids, int[] typeIds);
        bool CreateImagesPatterns(List<int> knownClassesIndexes,
List<int> unknownClassesIndexes, ushort[] ids, int[] typeIds,
            List<List<int>> knownRealizationsIndexes, List<List<int>>
unknownRealizationsIndexes);
        // Перемещение классов между категориями
известных/неизвестных
        bool MakeUnknown(bool[] whatKnownClassesIndexes);
        bool MakeKnown(bool[] whatUnknownClassesIndexes);
```

```
        bool MakeUnknownPatterns(bool[] whatKnownClassesIndexes);
        bool MakeKnownPatterns(bool[] whatUnknownClassesIndexes);
        // Объединение классов
        bool CombineClasses(List<int> classesIndexes, String name, bool
unknown);
        // Удаление данных
        bool DeleteClasses(List<int> knownClassesIndexes, List<int>
unknownClassesIndexes);
        bool DeleteRealizations(List<int> knownClassesIndexes, List<int>
unknownClassesIndexes,
            List<List<int>> knownRealizationsIndexes, List<List<int>>
unknownRealizationsIndexes);
        bool DeleteEtalons();
        bool DeletePatterns();

        #endregion


        // Задачи (моделирование, создание эталонов, исключение
ошибок, генерация реализаций и функция выполнения задач)
        #region Tasks

        // Перейти в пространство мета-признаков, normalize = 0 - без
нормировки, normalize=1 - нормировка только по откл, normalize=2 -
норм по мат ожид и откл
        double[][] ConvertFeatureSpaceToMetaspace(int normalize, bool
mixWithOriginalFeatures, ushort[] uniqueFeaturesIds, List<List<int>>
knownRealizationsIndexes, double p, bool calcCor);

        // Функции создания и удаления эталонов. Если allFeatures=true,
то uniqueFeaturesIds может быть равным null
        void CreateEtalons(int methodId, bool allFeatures, ushort[]
uniqueFeaturesIds, int[] featuresMethodsIds, bool originalPosOfFeatures,
bool storeCrossSections,
            List<int> knownClassesIndexes, List<int>
unknownClassesIndexes, double?[] parameters);
        void CreateEtalons(int methodId, bool allFeatures, ushort[]
uniqueFeaturesIds, int[] featuresMethodsIds, bool originalPosOfFeatures,
bool storeCrossSections,
            List<int> knownClassesIndexes, List<int>
unknownClassesIndexes, List<List<int>> knownRealizationsIndexes,
List<List<int>> unknownRealizationsIndexes, double?[] parameters,
List<List<int>> knownRealizationsIndexes_forTest);
        void CreateEtalons(int methodId, List<int> knownClassesIndexes,
List<int> unknownClassesIndexes, double?[] parameters);

        // Функции для проведения эксперимента
        void StartModelingIdentification(int classificatorId, bool
useAllFeatures, ushort[] uniqueFeaturesIds, bool originalPosOfFeatures,
            double?[] parametersValues, bool haveActiveParameter, int
indexOfActiveParameter, double[] activeParametersValues, String
activeParameterName,
            bool increaseFeaturesQuantitiy, int startEthQuantity, int
ethQuantityStep, bool useUnknownRealizations, String
unknownClassesName, String ethMethod,
            bool doNotUseTraned, bool saveHistory, String comment,
List<int> unknownClassesIndexes);
        void StartModelingVerification(int classificatorId, bool
useAllFeatures, ushort[] uniqueFeaturesIds, bool originalPosOfFeatures,
            double?[] parametersValues, bool haveActiveParameter, int
indexOfActiveParameter, double[] activeParametersValues, String
activeParameterName,
            bool increaseFeaturesQuantitiy, int startEthQuantity, int
ethQuantityStep, bool useUnknownRealizations, String
unknownClassesName, String ethMethod,
            bool doNotUseTraned, bool saveHistory, String comment, bool
noAnothers, List<int> unknownClassesIndexest);

        // Исключение грубых ошибок (если
maxQuantityOfRealsForEveryClass<=0, то количество анализируемых
реализаций не ограничено)
        void FindGrossErrorsInRealizations(int excluderId, List<int> classes,
List<int> unknownClasses, double?[] parameters,
            int maxQuantityOfRealsForEveryClass, bool
analisisAllRealizations, bool useAllFeatures, ushort[] uniqueFeaturesIds,
bool originalPosOfFeatures);

        // Функция для генерации реализаций
        // Первый пераметр - количество реализаций от каждого эталона
(т.е. общее количество реализаций равно количеству эталонов
умножить на realizationCount),
        // Второй параметр - количество значений каждого признака в
каждой реализации (т.е. значений признаков с одним и тем же id)
        void GenerateRealizations(int realizationCount, int
eachFeatureValuesCount, bool useSpecialSequanceOfNotUniqueFeatures,
ushort[] sequanceOfNotUniqueFeatures,
            bool createDependencyOfFeatures, bool
descendinglyDependency);

        int ExecuteTask(Task task, bool saveHistory);

        #endregion


        // События модели
        #region Events

        event BeginTask _etalonsCreationBegan;
        event ProgressTask _etalonsCreationInProgress;
        event EndEtalonsCreationTask _etalonsCreationEnd;
        event BeginTask _incorrectRealizationsEliminationBegan;
        event ProgressTask _incorrectRealizationsEliminationInProgress;
        event EndExclutionOfErrorsInRealizationsTask
_incorrectRealizationsEliminationEnd;
        event BeginTask _generationRealizationsBegan;
        event ProgressTask _generationRealizationsInProgress;
        event EndGenerationOfRealizationsTask
_generationRealizationsEnd;
        event BeginTask _experimentBegan;
        event ProgressTask _experimentInProgress;
        event EndExperimentTask _experimentEnd;
        event BeginTask _featureSpaceConvertionBegan;
        event ProgressTask _featureSpaceConvertionInProgress;
        event EndTask _featureSpaceConvertionEnd;

        #endregion
    }
}
```

Реализация тестового класса НПБК на базе корреляционных нейронов.

```
namespace SHV.Extentions.Etalons
{
    public struct MetaFeature
    {
        public int j; // номер первого признака в паре
        public int t; // номер второго признака в паре
    }

    public class CNeuroExtractor : IEtalon
    {
        // Описание параметров обученных нейронов
        private List<MetaFeature[]> _synapses;
        private List<int> _tablesIndexes;
        private List<double[]> _thresholds;
        private double[] _sx_stranger;
        private double[] _mx_stranger;
        private List<double[]> _w;

        // Параметры НПБК
        private int _neuronsNumber;
        private int _neuronsInputsNumber;
        private double _p;
        static private bool[][][] _tables_patterns = new bool[][][] {
            new bool[][] { new bool[] { true, true }, new bool[] { false, false },
new bool[] { true, false }, new bool[] { false, true } },
            new bool[][] { new bool[] { true, true }, new bool[] { true, false },
new bool[] { false, false }, new bool[] { false, true } },
```

```
            new bool[][] { new bool[] { false, false }, new bool[] { true, true },
new bool[] { true, false }, new bool[] { false, true } },
            new bool[][] { new bool[] { false, false }, new bool[] { true, false },
new bool[] { true, true }, new bool[] { false, true } },
            new bool[][] { new bool[] { true, false }, new bool[] { false, false },
new bool[] { true, true }, new bool[] { false, true } },
            new bool[][] { new bool[] { true, false }, new bool[] { true, true },
new bool[] { false, false }, new bool[] { false, true } },
            new bool[][] { new bool[] { false, true }, new bool[] { false, false },
new bool[] { true, false }, new bool[] { true, true } },
            new bool[][] { new bool[] { false, true }, new bool[] { true, false },
new bool[] { false, false }, new bool[] { true, true } },
            new bool[][] { new bool[] { false, false }, new bool[] { false, true },
new bool[] { true, false }, new bool[] { true, true } },
            new bool[][] { new bool[] { false, false }, new bool[] { true, false },
new bool[] { false, true }, new bool[] { true, true } },
            new bool[][] { new bool[] { true, false }, new bool[] { false, false },
new bool[] { false, true }, new bool[] { true, true } },
            new bool[][] { new bool[] { true, false }, new bool[] { false, true },
new bool[] { false, false }, new bool[] { true, true } },
            new bool[][] { new bool[] { true, true }, new bool[] { false, true },
new bool[] { true, false }, new bool[] { false, false } },
            new bool[][] { new bool[] { true, true }, new bool[] { true, false },
new bool[] { false, true }, new bool[] { false, false } },
            new bool[][] { new bool[] { false, true }, new bool[] { true, true },
new bool[] { true, false }, new bool[] { false, false } },
            new bool[][] { new bool[] { false, true }, new bool[] { true, false },
new bool[] { true, true }, new bool[] { false, false } },
            new bool[][] { new bool[] { true, false }, new bool[] { false, true },
new bool[] { true, true }, new bool[] { false, false } },
            new bool[][] { new bool[] { true, false }, new bool[] { true, true },
new bool[] { false, true }, new bool[] { false, false } },
            new bool[][] { new bool[] { true, true }, new bool[] { false, false },
new bool[] { false, true }, new bool[] { true, false } },
            new bool[][] { new bool[] { true, true }, new bool[] { false, true },
new bool[] { false, false }, new bool[] { true, false } },
            new bool[][] { new bool[] { false, false }, new bool[] { true, true },
new bool[] { false, true }, new bool[] { true, false } },
            new bool[][] { new bool[] { false, false }, new bool[] { false, true },
new bool[] { true, true }, new bool[] { true, false } },
            new bool[][] { new bool[] { false, true }, new bool[] { false, false },
new bool[] { true, true }, new bool[] { true, false } },
            new bool[][] { new bool[] { false, true }, new bool[] { true, true },
new bool[] { false, false }, new bool[] { true, false } } };

        // Далее идут параметры, которые НА ПРАКТИКЕ ДОЛЖНЫ
УДАЛЯТЬСЯ ПОСЛЕ ОБУЧЕНИЯ, некоторые из них нужны только
на этапе исследований
        private double _cor_min; // Границы интервалов
коррелированности признаков
        private double _cor_max; // Границы интервалов
коррелированности признаков
        private BitArray _key; // Ключ пользователя
        private int _neuronsAssimetry; // Допустимая разница между
количеством нейронов разных типов
        private double _thresholdCoef; // Коэффициент влияния на пороги
(используемый при обучении)
        private bool _normalizeByMx; // Нормализация по мат. ожиданиям
значений признаков для образов Чужих
        private bool _deleteWeakNeurons; // Удалять нестабильные
нейроны (балансировка порогов)
        private double _minimalSquare;
        private bool _calc_w; // Вычислять веса или нет

        private int GetTableIndex(int rightIntervalIndex, bool[] rightBits)
        {
            List<int> indexes = new List<int>();
            for (int i = 0; i < _tables_patterns.Length; i++)
                if (_tables_patterns[i][rightIntervalIndex][0] == rightBits[0] &&
_tables_patterns[i][rightIntervalIndex][1] == rightBits[1])
                    indexes.Add(i);
            int index = UniformDistribution.GetRandom().Next(0,
indexes.Count);
            return indexes[index];
        }

        // Если возвращает null, то нейрон невозможно создать
        private double[] GetThresholds_andSetTableIndex_inside(double
left_own, double right_own, double delta1, double delta2, double delta3,
IFeature ownFeature, IFeature aliensFeature, bool deleteUnstable)
        {
            double[] res = new double[3];
            int intervalsIndex = 0;
            if (delta2 < 0.1 && deleteUnstable) // 0.1
                return null;
            if (delta2 > 0.4 && deleteUnstable) // 0.4
                return null;
            else
            if (delta1 < 0.1) // 0.1
            {
                res[0] = right_own;
                if (delta3 < 0.6 && deleteUnstable) // 0.6
                    return null;
                else
                {
                    intervalsIndex = 0;
                    double right_aliens = aliensFeature.Mx + 4 *
aliensFeature.Sx;
                    double delta_aliens = (right_aliens - right_own) / 4;
                    res[1] = res[0] + delta_aliens;
                    res[2] = res[1] + delta_aliens;
                }
            }
            else
            if (delta1 > 0.4)
            {
                if (delta3 < 0.1)
                {
                    if (delta3 + delta2 > 0.4 && deleteUnstable) // 0.4
                        return null;
                    res[2] = left_own;
                    double left_aliens = aliensFeature.Mx - 4 * aliensFeature.Sx;
                    double delta_aliens = (left_own - left_aliens) / 4;
                    res[1] = res[2] - delta_aliens;
                    res[0] = res[1] - delta_aliens;
                    intervalsIndex = 3;
                }
                else
                {
                    double left_aliens = aliensFeature.Mx - 4 * aliensFeature.Sx;
                    double delta_aliens = (left_own - left_aliens) / 3;
                    res[2] = right_own;
                    res[1] = left_own;
                    res[0] = left_own - delta_aliens;
                    intervalsIndex = 2;
                }
            }
            else
            {
                double right_aliens = aliensFeature.Mx + 4 * aliensFeature.Sx;
                double delta_aliens = (right_aliens - right_own) / 3;
                res[0] = left_own;
                res[1] = right_own;
                res[2] = res[1] + delta_aliens;
                intervalsIndex = 1;
            }
            bool[] bits = new bool[2];
            bits[0] = _key[_synapses.Count * 2];
            bits[1] = _key[_synapses.Count * 2 + 1];
            _tablesIndexes.Add(GetTableIndex(intervalsIndex, bits));
            return res;
        }

        // Если возвращает null, то нейрон невозможно создать
        private double[] GetThresholds_andSetTableIndex(MetaFeature[]
neuron, double[][] own, double[][] aliens, double p, bool deleteUnstable,
double minimalSquare, double[] w, double thresholdCoef_2 = 4, bool
simple = false)
        {
            double[] ownOutputs = new double[own.Length];
            double[] aliensOutputs = new double[aliens.Length];
            for (int i = 0; i < own.Length; i++)
                ownOutputs[i] = GetNeuronOutput(neuron, own[i], p, _calc_w,
w);
            for (int i = 0; i < aliens.Length; i++)
```

```
            aliensOutputs[i] = GetNeuronOutput(neuron, aliens[i], p,
_calc_w, w);
        IFeature ownFeature = new
NormalLawFeature_GenMK(ownOutputs, 0, "", 0);
        IFeature aliensFeature = new
NormalLawFeature_GenMK(aliensOutputs, 0, "", 0);
        if (minimalSquare < 1)
        {
            double square = Manager.EvaluateSquare(ownFeature,
aliensFeature, 100, false);
            if (square > minimalSquare)
                return null;
        }
        double thresholdCoef = 1;
        double left_own = ownFeature.Mx - thresholdCoef_2 *
thresholdCoef * ownFeature.Sx;
        double right_own = ownFeature.Mx + thresholdCoef_2 *
thresholdCoef * ownFeature.Sx;
        double delta1 = aliensFeature.GetDistributionFunction(left_own);
        double delta = aliensFeature.GetDistributionFunction(right_own);
        double delta2 = delta - delta1;
        if (simple)
        {
            int intervalsIndex=0;
            if (delta2 > 0.8)
                return null;
            double[] res = new double[3];
            if (delta1 > 1 - delta)
            {
                if (delta1 > 0.3)
                {
                    res[0] = ownFeature.Mx - 2 * thresholdCoef_2 *
thresholdCoef * ownFeature.Sx;
                    res[1] = ownFeature.Mx - 1.5 * thresholdCoef_2 *
thresholdCoef * ownFeature.Sx;
                    res[2] = left_own;
                    intervalsIndex = 3;
                }
                else
                {
                    res[0] = ownFeature.Mx - 1.5 * thresholdCoef_2 *
thresholdCoef * ownFeature.Sx;
                    res[1] = left_own;
                    res[2] = right_own;
                    intervalsIndex = 2;
                }
            }
            else
            {
                if (1 - delta < 0.3)
                {
                    res[0] = left_own;
                    res[1] = right_own;
                    res[2] = ownFeature.Mx + 1.5 * thresholdCoef_2 *
thresholdCoef * ownFeature.Sx;
                    intervalsIndex = 1;
                }
                else
                {
                    res[0] = right_own;
                    res[1] = ownFeature.Mx + 1.5 * thresholdCoef_2 *
thresholdCoef * ownFeature.Sx;
                    res[2] = ownFeature.Mx + 2 * thresholdCoef_2 *
thresholdCoef * ownFeature.Sx;
                    intervalsIndex = 0;
                }
            }
            bool[] bits = new bool[2];
            bits[0] = _key[_synapses.Count * 2];
            bits[1] = _key[_synapses.Count * 2 + 1];
            _tablesIndexes.Add(GetTableIndex(intervalsIndex, bits));
            return res;
        }

        if (deleteUnstable)
            while (delta2 < 0.1) // 0.1
            {
                //                thresholdCoef_2 = thresholdCoef_2 + 0.1;
                thresholdCoef = thresholdCoef * 1.05;
                left_own = ownFeature.Mx - thresholdCoef_2 *
thresholdCoef * ownFeature.Sx;
                right_own = ownFeature.Mx + thresholdCoef_2 *
thresholdCoef * ownFeature.Sx;
                delta1 = aliensFeature.GetDistributionFunction(left_own);
                delta = aliensFeature.GetDistributionFunction(right_own);
                delta2 = delta - delta1;
                /*                double add = 0.1;
                                    if (delta2 > 0.4) // 0.4
                                    {
                                        thresholdCoef_2 = thresholdCoef_2 -
add;
                                        add = add / 10;
                                        thresholdCoef_2 = thresholdCoef_2
+ add;
                                        left_own = ownFeature.Mx -
thresholdCoef_2 * thresholdCoef * ownFeature.Sx;
                                        right_own = ownFeature.Mx +
thresholdCoef_2 * thresholdCoef * ownFeature.Sx;
                                        delta1 =
aliensFeature.GetDistributionFunction(left_own);
                                        delta =
aliensFeature.GetDistributionFunction(right_own);
                                        delta2 = delta - delta1;
                                    }*/
            }

        double delta3 = 1 - delta;
        return GetThresholds_andSetTableIndex_inside(left_own,
right_own, delta1, delta2, delta3, ownFeature, aliensFeature,
deleteUnstable);
    }

    private double[][] GetMetaFeaturesOfSecondOrder(MetaFeature[]
neuron, double[][] normalizedFetures, double p)
    {
        double[][] res = new double[normalizedFetures.Length][];
        for (int j = 0; j < normalizedFetures.Length; j++)
        {
            double[] metaFeatures = new double[neuron.Length];
            double my =
Math.Abs(Math.Pow(Math.Abs(normalizedFetures[j][neuron[0].j]), p) -
Math.Pow(Math.Abs(normalizedFetures[j][neuron[0].t]), p));
            metaFeatures[0] = my;
            for (int i = 1; i < neuron.Length; i++)
            {
                metaFeatures[i] =
Math.Abs(Math.Pow(Math.Abs(normalizedFetures[j][neuron[i].j]), p) -
Math.Pow(Math.Abs(normalizedFetures[j][neuron[i].t]), p));
                my = Statistica.Mx_a_rct(my, metaFeatures[i], i + 1);
            }
            for (int i = 0; i < neuron.Length; i++)
                metaFeatures[i] = Math.Pow(metaFeatures[i] - my, 2);
            res[j] = metaFeatures;
        }
        return res;
    }

    private double[] GetW(MetaFeature[] neuron, double[][] own,
double[][] aliens, double p)
    {
        double[][] own_meta = GetMetaFeaturesOfSecondOrder(neuron,
own, p);
        double[][] aliens_meta = GetMetaFeaturesOfSecondOrder(neuron,
aliens, p);
        double[] res = new double[neuron.Length];
        for (int i = 0; i < neuron.Length; i++)
        {
            double m_own = own_meta[0][i];
            m_own = Statistica.Mx_a_rct(m_own, own_meta[1][i], 2);
            double d_own = (Math.Pow(own_meta[0][i] - m_own, 2) +
Math.Pow(own_meta[1][i] - m_own, 2)) / 2;
            for (int j = 2; j < own_meta.Length; j++)
            {
                m_own = Statistica.Mx_a_rct(m_own, own_meta[j][i], j + 1);
                d_own = Statistica.Dx_a_rct(d_own, m_own, own_meta[j][i],
j + 1);
```

```
            }
            double m_aliens = aliens_meta[0][i];
            m_aliens = Statistica.Mx_a_rct(m_aliens, aliens_meta[1][i], 2);
            double d_aliens = (Math.Pow(aliens_meta[0][i] - m_aliens, 2) +
Math.Pow(aliens_meta[1][i] - m_aliens, 2)) / 2;
            for (int j = 2; j < aliens_meta.Length; j++)
            {
               m_aliens = Statistica.Mx_a_rct(m_aliens, aliens_meta[j][i], j
+ 1);
               d_aliens = Statistica.Dx_a_rct(d_aliens, m_aliens,
aliens_meta[j][i], j + 1);
            }
            d_own = Math.Sqrt(d_own);
            d_aliens = Math.Sqrt(d_aliens);
            res[i] = Math.Abs(m_own - m_aliens) / (d_own * d_aliens);
         }
         return res;
      }

      private double GetNeuronOutput_no_w(MetaFeature[] neuron,
double[] normalizedFetureVector, double p)
      {
         double[] metaFeatures = new double[neuron.Length];
         double my =
Math.Abs(Math.Pow(Math.Abs(normalizedFetureVector[neuron[0].j]), p)
- Math.Pow(Math.Abs(normalizedFetureVector[neuron[0].t]), p));
         metaFeatures[0] = my;
         for (int i = 1; i < neuron.Length; i++)
         {
            metaFeatures[i] =
Math.Abs(Math.Pow(Math.Abs(normalizedFetureVector[neuron[i].j]), p) -
Math.Pow(Math.Abs(normalizedFetureVector[neuron[i].t]), p));
            my = Statistica.Mx_a_rct(my, metaFeatures[i], i + 1);
         }
         double res = Statistica.Sx_a(metaFeatures, my);
         return res;
      }

      private double GetNeuronOutput_w(MetaFeature[] neuron, double[]
normalizedFetureVector, double p, double[] w)
      {
         double[] metaFeatures = new double[neuron.Length];
         double my =
Math.Abs(Math.Pow(Math.Abs(normalizedFetureVector[neuron[0].j]), p)
- Math.Pow(Math.Abs(normalizedFetureVector[neuron[0].t]), p));
         metaFeatures[0] = my;
         for (int i = 1; i < neuron.Length; i++)
         {
            metaFeatures[i] =
Math.Abs(Math.Pow(Math.Abs(normalizedFetureVector[neuron[i].j]), p) -
Math.Pow(Math.Abs(normalizedFetureVector[neuron[i].t]), p));
            my = Statistica.Mx_a_rct(my, metaFeatures[i], i + 1);
         }
         double my___= Math.Pow(metaFeatures[0] - my, 2) * w[0];
         for (int i = 1; i < neuron.Length; i++)
         {
            metaFeatures[i] = Math.Pow(metaFeatures[i]-my, 2)*w[i];
            my___ = Statistica.Mx_a_rct(my___, metaFeatures[i], i + 1);
         }
         my___ = Math.Sqrt(my___);
         return my___;
      }

      private double GetNeuronOutput(MetaFeature[] neuron, double[]
normalizedFetureVector, double p, bool calc_w, double[] w)
      {
         if (calc_w)
            return GetNeuronOutput_w(neuron, normalizedFetureVector, p,
w);
         return GetNeuronOutput_no_w(neuron, normalizedFetureVector,
p);
      }

      private bool[] GetNeuronActivation(double y, double[] thresholds, int
tableIndex)
      {
         if (y < thresholds[0])
            return _tables_patterns[tableIndex][0];
         if ((thresholds[0] <= y) && (y < thresholds[1]))
            return _tables_patterns[tableIndex][1];
         if ((thresholds[1] <= y) && (y < thresholds[2]))
            return _tables_patterns[tableIndex][2];
         return _tables_patterns[tableIndex][3];
      }

      public CNeuroExtractor(String imageClassName, IFeature[] features,
Realization[] realizations, FeatureCrossSection[] featureCrossSections,
ushort maxFeatureIdInGlobalScope, double?[] parameters, int typeId)
         : base(imageClassName, features, realizations,
featureCrossSections, maxFeatureIdInGlobalScope, parameters, typeId)
      {
         if (OwnRealizations.Length < 2) throw new
ArgumentException("Для создания эталонов данного типа необходимы
примеры образов (более одного)");
         _neuronsNumber = 128;
         _neuronsInputsNumber = 4;
         _cor_min = -0.5;
         _cor_max = 0.5;
         _neuronsAssimetry = _neuronsNumber / 10;
         _thresholdCoef = 1;
         _normalizeByMx = false;
         _p = 0.9;
         _deleteWeakNeurons = false;
         _minimalSquare = 1;
         _calc_w = true;

         if (parameters != null)
         {
            if (parameters[0] != null)
               _neuronsNumber = (int)parameters[0];
            if (parameters[1] != null)
               _neuronsInputsNumber = (int)parameters[1];
            if (parameters[2] != null)
               _cor_min = (double)parameters[2];
            if (parameters[3] != null)
               _cor_max = (double)parameters[3];
            if (parameters[4] != null)
               _neuronsAssimetry = (int)parameters[4];
            if (parameters[5] != null)
               _thresholdCoef = (double)parameters[5];
            if (parameters[6] != null)
               if ((int)parameters[6] == 1)
                  _normalizeByMx = true;
            if (parameters[7] != null)
               _p = (double)parameters[7];
            if (parameters[8] != null)
               if ((int)parameters[8] == 1)
                  _deleteWeakNeurons = true;
            if (parameters[9] != null)
               _minimalSquare = (double)parameters[9];
            if (parameters[10] != null)
               if ((int)parameters[10] == 0)
                  _calc_w = false;
         }
         _key = new BitArray(_neuronsNumber * 2);
         Random rand = UniformDistribution.GetRandom();
         for (int i = 0; i < _neuronsNumber * 2; i++)
            if (rand.Next(0, 2) == 0)
               _key[i] = true;
            else
               _key[i] = false;
      }

      public override BitArray GetCorrectBinaryCode(int
realizationFeaturesValuesCount)
      {
         return _key;
      }

      public override double[] GetParameters()
      {
         double[] par = new double[10];
         par[0] = _neuronsNumber;
         par[1] = _neuronsInputsNumber;
         par[2] = _cor_min;
         par[3] = _cor_max;
```

```
        par[4] = _neuronsAssimetry;
        par[5] = _thresholdCoef;
        if (_normalizeByMx)
            par[6] = 1;
        else
            par[6] = 0;
        par[7] = _p;
        if (_deleteWeakNeurons)
            par[8] = 1;
        else
            par[8] = 0;
        par[9] = _minimalSquare;
        return par;
    }

    public override void TrainingWithGlobalScopeData(IEtalon[]
etalons, int ownIndex)
    {
        // Готовим нормирующие коэффициенты
        _sx_stranger = new double[Features.Length];
        _mx_stranger = null;
        if (_normalizeByMx)
            _mx_stranger = new double[Features.Length];
        for (int j = 0; j < Features.Length; j++)
        {
            List<double> cs = new List<double>();
            for (int i = 0; i < etalons.Length; i++)

cs.Add(etalons[i].OwnRealizations[0].FindFeatureValue(Features[j].Id, j,
0));
            if (!_normalizeByMx)
                _sx_stranger[j] = Statistica.Sx_a(cs);
            else
            {
                _mx_stranger[j] = Statistica.Mx_a(cs);
                _sx_stranger[j] = Statistica.Sx_a(cs, _mx_stranger[j]);
            }
        }

        // Готовим обучающие выборки
        ushort[] ids = GetFeaturesIds();
        double[][] owns = owns = new
double[_ownRealizations.Length][];
        double[][] aliens = new double[etalons.Length][];

        if (_normalizeByMx)
        {
            for (int i = 0; i < _ownRealizations.Length; i++)
                owns[i] =
Manager.GetVectorOfNormalizedUniqueFeaturesValues(_ownRealization
s[i], ids, 0, _mx_stranger, _sx_stranger, _p);
            for (int i = 0; i < etalons.Length; i++)
                aliens[i] =
Manager.GetVectorOfNormalizedUniqueFeaturesValues(etalons[i].OwnR
ealizations[1], ids, 0, _mx_stranger, _sx_stranger, _p);
        }
        else
        {
            for (int i = 0; i < _ownRealizations.Length; i++)
                owns[i] =
Manager.GetVectorOfNormalizedUniqueFeaturesValues(_ownRealization
s[i], ids, 0, _sx_stranger, _p);
            for (int i = 0; i < etalons.Length; i++)
                aliens[i] =
Manager.GetVectorOfNormalizedUniqueFeaturesValues(etalons[i].OwnR
ealizations[1], ids, 0, _sx_stranger, _p);
        }

        // Вычисление матрицы корреляции между признаками для
всех образов Свой
        double[][] cor =
Manager.CalcCorrelationMatrix(Manager.GetCrossSections(owns));

        // Синтезируем diff neuro extractor
        _tablesIndexes = new List<int>();
        _synapses = new List<MetaFeature[]>();
        _w = new List<double[]>();
        _thresholds = new List<double[]>();
        if (_calc_w)
            _w = new List<double[]>();
        while (_synapses.Count < _neuronsNumber / 2)
        {
            _tablesIndexes.Clear();
            _synapses.Clear();
            _thresholds.Clear();
            int count_minus_neurons = 0;
            int count_plus_neurons = 0;
            int count_minus = 0;
            int count_plus = 0;
            MetaFeature[] neuron_minus = new
MetaFeature[_neuronsInputsNumber];
            MetaFeature[] neuron_plus = new
MetaFeature[_neuronsInputsNumber];
            for (int j = 0; j < _features.Length; j++)
                for (int t = j + 1; t < _features.Length; t++)
                {
                    if ((count_minus_neurons < _neuronsNumber / 2 +
_neuronsAssimetry) && (count_minus_neurons < _neuronsNumber -
count_plus_neurons) && (cor[j][t] < _cor_min))
                    {
                        MetaFeature meta_f = new MetaFeature();
                        meta_f.j = j;
                        meta_f.t = t;
                        neuron_minus[count_minus] = meta_f;
                        count_minus++;
                        if (count_minus == _neuronsInputsNumber)
                        {
                            double[] w = null;
                            if (_calc_w)
                                w = GetW(neuron_minus, owns, aliens, _p);
                            double[] thresholds =
GetThresholds_andSetTableIndex(neuron_minus, owns, aliens, _p,
_deleteWeakNeurons, _minimalSquare, w, _thresholdCoef);
                            if (thresholds != null)
                            {
                                _synapses.Add(neuron_minus);
                                _w.Add(w);
                                _thresholds.Add(thresholds);
                                count_minus_neurons++;
                            }
                            neuron_minus = new
MetaFeature[_neuronsInputsNumber];
                            count_minus = 0;
                        }
                    }
                    else
                    if ((count_plus_neurons < _neuronsNumber / 2 +
_neuronsAssimetry) && (count_plus_neurons < _neuronsNumber -
count_minus_neurons) && (cor[j][t] > _cor_max))
                    {
                        MetaFeature meta_f = new MetaFeature();
                        meta_f.j = j;
                        meta_f.t = t;
                        neuron_plus[count_plus] = meta_f;
                        count_plus++;
                        if (count_plus == _neuronsInputsNumber)
                        {
                            double[] w = null;
                            if (_calc_w)
                                w = GetW(neuron_plus, owns, aliens, _p);
                            double[] thresholds =
GetThresholds_andSetTableIndex(neuron_plus, owns, aliens, _p,
_deleteWeakNeurons, _minimalSquare, w, _thresholdCoef);
                            if (thresholds != null)
                            {
                                _synapses.Add(neuron_plus);
                                _w.Add(w);
                                _thresholds.Add(thresholds);
                                count_plus_neurons++;
                            }
                            neuron_plus = new
MetaFeature[_neuronsInputsNumber];
                            count_plus = 0;
                        }
                    }
                }
```

```
            bool ok = false;
            if (count_minus_neurons < _neuronsNumber / 2 +
_neuronsAssimetry)
                if (_cor_min < -0.35)
                {
                    _cor_min = _cor_min + 0.05;
                    ok = true;
                }
            if (count_plus_neurons < _neuronsNumber / 2 +
_neuronsAssimetry)
                if (_cor_max > 0.35)
                {
                    _cor_max = _cor_max - 0.05;
                    ok = true;
                }
            if (!ok) return;
        }
    }

    public override List<String> GetHistoryOfTraining()
    {
        List<String> res = new List<string>();
        res.Add("Всего нейронов " + _synapses.Count.ToString());
        return res;
    }

    public override BitArray GetBinaryProximityVector(Realization
realization, double?[] parameters = null)
    {
        BitArray res = new BitArray(2 * _synapses.Count);
        ushort[] ids = GetFeaturesIds();
        double[] realization_norm = null;
        if (_normalizeByMx)
            realization_norm =
Manager.GetVectorOfNormalizedUniqueFeaturesValues(realization, ids,
0, _mx_stranger, _sx_stranger, _p);
        else
            realization_norm =
Manager.GetVectorOfNormalizedUniqueFeaturesValues(realization, ids,
0, _sx_stranger, _p);
        for (int i = 0; i < _synapses.Count; i++)
        {
            double y = GetNeuronOutput(_synapses[i], realization_norm,
_p, _calc_w, _w[i]);
            bool[] bits = GetNeuronActivation(y, _thresholds[i],
_tablesIndexes[i]);
            res[i * 2] = bits[0];
            res[i * 2 + 1] = bits[1];
        }
        return res;
    }

    /*
    private double GetCloseness(double[][] realizations, int
synapseIndex)
    {
        double res = 1;
        for (int i = 0; i < realizations.Length; i++)
        {
            double y = GetNeuronOutput(_synapses[synapseIndex],
realizations[i], _p, _calc_w, _w[i]);
            bool[] bits = GetNeuronActivation(y,
_thresholds[synapseIndex], _tablesIndexes[synapseIndex]);
            if (_key[synapseIndex * 2] != bits[0] && _key[synapseIndex * 2
+ 1] != bits[1])
                res = res + 2;
            else
            if (_key[synapseIndex * 2] != bits[0] || _key[synapseIndex * 2 +
1] != bits[1])
                res = res + 1;
        }
        res = res / (realizations.Length+1);
        return res;
    }*/
  }
}
```

Реализация простейшего классификатора на базе меры Хэмминга для тестирования НПБК.

```
namespace SHV.Extentions.Classificators
{
  public class NeuralNetworkAdapter_Mark1 : IClassificator
  {
    private double?[] _par; // Внутренние параметры
эталоновпараметра (пороговые значения)
    private bool _normilize; // при true - нормировать расстояние
Хэмминга по количеству бит

//=================================================
=================================================
===

    public NeuralNetworkAdapter_Mark1(IEtalon[] etalons,
FeaturesSpecification specification, int typeId, bool saveHistory)
        : base(etalons, specification, typeId, saveHistory)
    {

    }

    public override void SetParameters(double?[] parameters)
    {
        _par = null;
        _normilize = false;
        if (parameters[0] == 1) _normilize = true;
        _par = new double?[parameters.Length - 1];
        for (int i = 1; i < parameters.Length; i++)
            _par[i - 1] = parameters[i];
        // Узнаем верные коды эталонов
        for (int i = 0; i < _etalons.Length; i++)
        {
// ТЕПЕРЬ ВСЕ ИНАЧЕ, НО МОЖНО ВЕРНУТЬ GetCorrectBinaryCode
//            _ethCodes[i] =
_etalons[i].GetBinaryProximityVector(_etalons[i].GetVectorOfMxAsReali
zation(), _par);
            BitArray correctCode = _etalons[i].GetCorrectBinaryCode(0);
            if (NeedSaveHistory)
            {
                String str = "";
                for (int j = 0; j < /*_ethCodes[i]*/correctCode.Length; j++)
                    str +=
(Convert.ToInt32(/*_ethCodes[i]*/correctCode[j])).ToString();
                CreateHistoryForEtalon(str, i);
            }
        }
    }

    public override double[] GetParameters()
    {
        double[] res = null;
        if (_par != null)
        {
            res = new double[_par.Length + 1];
            for (int i = 0; i < _par.Length; i++)
                if (_par[i] == null) res[i + 1] = Double.MinValue;
                else
                    res[i + 1] = (double)_par[i];
        }
        else
            res = new double[1];
        if (_normilize)
            res[0] = 1;
```

```
            else
                res[0] = 0;
            return res;
        }

        public override long GetKeyLength(double[] parameters, int
featuresQuantity, int knownImagesQuantity, int ethIndex)
        {
            try
            {
                BitArray code =
_etalons[ethIndex].GetCorrectBinaryCode(featuresQuantity);
//_etalons[ethIndex].GetBinaryProximityVector(_etalons[ethIndex].GetVe
ctorOfMxAsRealization(), _par);
                return code.Length;
            }
            catch { return 0; }
        }

        public override double[] Verify_lastStep(Realization realization, int
etalonIndex, int etalonIndexRealizationFrom)
        {
            BitArray code =
_etalons[etalonIndex].GetBinaryProximityVector(realization, _par);
            if (code == null) return null;
//==================================================
==========================================================
===
            BitArray correctCode =
_etalons[etalonIndex].GetCorrectBinaryCode(realization.GetAllFeaturesV
alues().Length);
            int min=/*_ethCodes[etalonIndex]*/correctCode.Length;
//==================================================
==========================================================
===
            if (min > code.Length)
                min=code.Length;
            int hemmingDistance = Math.Abs(code.Length -
/*_ethCodes[etalonIndex]*/correctCode.Length);
            String code_str = "";
            for (int i = 0; i < min; i++)
            {
                code_str += (Convert.ToInt32(code[i])).ToString();
                if (code[i] != /*_ethCodes[etalonIndex]*/correctCode[i])
                    hemmingDistance++;
            }
            double[] res = new double[1];
            if (_normilize)
                res[0] = (double)hemmingDistance / min;
            else
                res[0] = (double)hemmingDistance;
            if (NeedSaveHistory)
                AddHistory(code_str, etalonIndex,
etalonIndexRealizationFrom);
            return res;
        }
        public override double[] Verify_allSteps(Realization realization, int
etalonIndex, int etalonIndexRealizationFrom)
        {
            BitArray code =
_etalons[etalonIndex].GetBinaryProximityVector(realization, _par);
            if (code == null) return null;
//==================================================
==========================================================
===
            BitArray correctCode =
_etalons[etalonIndex].GetCorrectBinaryCode(realization.GetAllFeaturesV
alues().Length);
            int min =/*_ethCodes[etalonIndex]*/correctCode.Length;
//==================================================
==========================================================
===
            if (min > code.Length)
                min = code.Length;
            int hemmingDistance = Math.Abs(code.Length -
/*_ethCodes[etalonIndex]*/correctCode.Length);
            double[] res = new double[min];
            String code_str = "";
            for (int i = 0; i < min; i++)
            {
                code_str += (Convert.ToInt32(code[i])).ToString();
                if (code[i] != /*_ethCodes[etalonIndex]*/correctCode[i])
                    hemmingDistance++;
                if (_normilize)
                    res[i] = (double)hemmingDistance / min;
                else
                    res[i] = (double)hemmingDistance;
            }
            if (NeedSaveHistory)
                AddHistory(code_str, etalonIndex,
etalonIndexRealizationFrom);
            return res;
        }

        // - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

        #region NotImplemented

        public override double[][] Identify_lastStep(Realization realization)
        {
            throw new NotImplementedException();
        }
        public override double[][] Identify_allSteps(Realization realization)
        {
            throw new NotImplementedException();
        }

        #endregion
    }
}
```

Приложение 9 Патент на изобретение

РОССИЙСКАЯ ФЕДЕРАЦИЯ

ПАТЕНТ

НА ИЗОБРЕТЕНИЕ

№ 2543927

СПОСОБ ИДЕНТИФИКАЦИИ ЛИЧНОСТИ ПО ОСОБЕННОСТЯМ ДИНАМИКИ НАПИСАНИЯ ПАРОЛЯ

Патентообладатель(ли): *Епифанцев Борис Николаевич (RU), Ложников Павел Сергеевич (RU), Самотуга Александр Евгеньевич (RU), Сулавко Алексей Евгеньевич (RU)*

Автор(ы): *см. на обороте*

Заявка № **2014116281**

Приоритет изобретения **22 апреля 2014 г.**

Зарегистрировано в Государственном реестре изобретений Российской Федерации ***03 февраля 2015 г.***

Срок действия патента истекает **22 апреля 2034 г.**

Врио руководителя Федеральной службы по интеллектуальной собственности

Л.Л. Кирий

Приложение 10 Свидетельства о регистрации программ для ЭВМ и электронных ресурсов

РОССИЙСКАЯ ФЕДЕРАЦИЯ

СВИДЕТЕЛЬСТВО

о государственной регистрации программы для ЭВМ

№ 2021660512

AIC desktop

Правообладатель: ***Сулавко Алексей Евгеньевич (RU)***

Авторы: ***Сулавко Алексей Евгеньевич (RU), Стадников Денис Геннадьевич (RU), Чобан Адиль Гаврилович (KZ), Иниватов Даниил Павлович (RU)***

Заявка № **2021617236**
Дата поступления **17 мая 2021 г.**
Дата государственной регистрации
в Реестре программ для ЭВМ ***28 июня 2021 г.***

Руководитель Федеральной службы по интеллектуальной собственности

ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ
Сертификат 0x02A5CFBC00B1ACF59A40A2F08092E9A118
Владелец Ивлиев Григорий Петрович
Действителен с 15.01.2021 по 15.01.2035

Г.П. Ивлиев

РОССИЙСКАЯ ФЕДЕРАЦИЯ

СВИДЕТЕЛЬСТВО

о государственной регистрации программы для ЭВМ

№ 2017616888

Среда для имитационного моделирования экспериментов и проверки гипотез по распознаванию образов «SHV-kernel»

Правообладатель: ***Сулавко Алексей Евгеньевич (RU)***

Автор: ***Сулавко Алексей Евгеньевич (RU)***

Заявка № **2017614035**
Дата поступления **24 апреля 2017 г.**
Дата государственной регистрации
в Реестре программ для ЭВМ ***19 июня 2017 г.***

Руководитель Федеральной службы
по интеллектуальной собственности

Г.П. Ивлиев

РОССИЙСКАЯ ФЕДЕРАЦИЯ

СВИДЕТЕЛЬСТВО

о государственной регистрации программы для ЭВМ

№ 2010610473

Программный модуль для обеспечения безопасности бухгалтерских информационных систем «TEOFRAST-B»

Правообладатель(ли): ***Общество с ограниченной ответственностью «Научно-технический центр «КАСИБ» (RU)***

Автор(ы): ***Ложников Павел Сергеевич, Еременко Александр Валериевич, Перевальский Виктор Александрович, Сулавко Алексей Евгеньевич (RU)***

Заявка № **2009616252**

Дата поступления **10 ноября 2009 г.**

Зарегистрировано в Реестре программ для ЭВМ

11 января 2010 г.

Руководитель Федеральной службы по интеллектуальной собственности, патентам и товарным знакам

Б.П. Симонов

РОССИЙСКАЯ ФЕДЕРАЦИЯ

СВИДЕТЕЛЬСТВО

о государственной регистрации программы для ЭВМ

№ 2012611310

Мультифакторная система аутентификации «TEOFRAST-M»

Правообладатель(ли): ***Общество с ограниченной ответственностью «Научно-технический центр «КАСИБ» (RU)***

Автор(ы): ***Ложников Павел Сергеевич, Перевальский Виктор Александрович, Сулавко Алексей Евгеньевич (RU)***

Заявка № **2011619263**

Дата поступления **5 декабря 2011 г.**

Зарегистрировано в Реестре программ для ЭВМ

1 февраля 2012 г.

Руководитель Федеральной службы по интеллектуальной собственности

Б.П. Симонов

РОССИЙСКАЯ ФЕДЕРАЦИЯ

СВИДЕТЕЛЬСТВО

о государственной регистрации программы для ЭВМ

№ 2011613340

Система безопасности компьютера на основе регистрируемых событий в компьютерных сетях

Правообладатель(ли): ***Общество с ограниченной ответственностью «Пегас инжиниринг» (RU)***

Автор(ы): ***Сулавко Алексей Евгеньевич,***
Голованов Сергей Анатольевич (RU)

Заявка № **2011611363**
Дата поступления **3 марта 2011 г.**
Зарегистрировано в Реестре программ для ЭВМ
28 апреля 2011 г.

Руководитель Федеральной службы по интеллектуальной собственности, патентам и товарным знакам

Б.П. Симонов

РОССИЙСКАЯ ФЕДЕРАЦИЯ

СВИДЕТЕЛЬСТВО

о государственной регистрации программы для ЭВМ

№ 2012615385

Распределенная система управления доступом к ресурсам компьютера на основе регистрируемых событий

Правообладатель(ли): ***Общество с ограниченной ответственностью «Пегас инжиниринг» (RU)***

Автор(ы): ***Сулавко Алексей Евгеньевич, Бородин Антон Александрович (RU)***

Заявка № **2012612981**

Дата поступления **18 апреля 2012 г.**

Зарегистрировано в Реестре программ для ЭВМ

15 июня 2012 г.

Руководитель Федеральной службы по интеллектуальной собственности

Б.П. Симонов

РОССИЙСКАЯ ФЕДЕРАЦИЯ

СВИДЕТЕЛЬСТВО

о государственной регистрации программы для ЭВМ

№ 2019663412

«Программный модуль для цифрового подписания PDF-документов «PdfDigiSign»

Правообладатель: ***Федеральное государственное бюджетное образовательное учреждение высшего образования «Омский государственный технический университет» (RU)***

Авторы: ***Ложников Павел Сергеевич (RU), Семиколенов Михаил Андреевич (RU), Сулавко Алексей Евгеньевич (RU)***

Заявка № **2019662174**
Дата поступления **07 октября 2019 г.**
Дата государственной регистрации
в Реестре программ для ЭВМ ***16 октября 2019 г.***

Руководитель Федеральной службы
по интеллектуальной собственности

Г.П. Ивлиев

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное бюджетное научное учреждение
ИНСТИТУТ УПРАВЛЕНИЯ ОБРАЗОВАНИЕМ РОССИЙСКОЙ АКАДЕМИИ ОБРАЗОВАНИЯ

ОБЪЕДИНЕННЫЙ ФОНД ЭЛЕКТРОННЫХ РЕСУРСОВ "НАУКА И ОБРАЗОВАНИЕ"
(основан в 1991 году)

СВИДЕТЕЛЬСТВО О РЕГИСТРАЦИИ ЭЛЕКТРОННОГО РЕСУРСА

№ 23578

Настоящее свидетельство выдано на электронный ресурс, отвечающий требованиям новизны и приоритетности:

Разрез файлов формата WAV

Дата регистрации: **26 апреля 2018 года**

Авторы: **Иниватов Д.П., Сулавко А.Е.**

Организация-разработчик: **ФГБОУ ВО «Омский государственный технический университет»**

Директор ФГБНУ ИУО РАО,
доктор экономических наук _____ С.С. Неустроев

Руководитель ОФЭРНиО, почетный
работник науки и техники России _____ А.И. Галкина