Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Тамбовский государственный технический университет»

А. Д. ОБУХОВ, М. Н. КРАСНЯНСКИЙ

СТРУКТУРНО-ПАРАМЕТРИЧЕСКИЙ СИНТЕЗ АДАПТИВНЫХ ИНФОРМАЦИОННЫХ СИСТЕМ НА ОСНОВЕ НЕЙРОСЕТЕВЫХ МЕТОДОВ И АРХИТЕКТУРЫ

Рекомендовано Научно-техническим советом
федерального государственного бюджетного
образовательного учреждения высшего образования
«Тамбовский государственный технический университет»
в качестве монографии

Научное издание

Т·Г·Т·У

Тамбов
Издательский центр ФГБОУ ВО «ТГТУ»
2021

УДК 004
ББК 32.972.53
О-26

Рецензенты:

Доктор технических наук, профессор,
проректор по научно-инновационной деятельности
ФГБОУ ВО «ТГТУ»
Д. Ю. Муромцев

Доктор технических наук, профессор,
заведующий кафедрой информационных систем
Института информационных систем и технологий
МГТУ «СТАНКИН»
Б. М. Позднеев

Обухов, А. Д.

О-26 Структурно-параметрический синтез адаптивных информационных систем на основе нейросетевых методов и архитектуры : монография / А. Д. Обухов, М. Н. Краснянский. – Тамбов : Издательский центр ФГБОУ ВО «ТГТУ», 2021. – 240 с. – 400 экз.

ISBN 978-5-8265-2353-7

Рассмотрен вопрос структурно-параметрического синтеза адаптивных информационных систем, изложены подходы к анализу, обработке и генерации данных в них, реализации алгоритмов управления с применением нейросетевых методов и архитектуры. Предложена конкретная реализация нейросетевых методов, сформулированы основные принципы методологии синтеза адаптивных информационных систем, проведена их апробация в области систем электронного документооборота.

Предназначена для специалистов в области разработки адаптивных систем для различных предметных областей, автоматизации процессов анализа, обработки и генерации данных, реализации систем управления, а также магистрантов и студентов старших курсов, специализирующихся в области системного анализа и разработки информационных систем.

УДК 004
ББК 32.972.53

ISBN 978-5-8265-2353-7

© Федеральное государственное бюджетное образовательное учреждение высшего образования «Тамбовский государственный технический университет» (ФГБОУ ВО «ТГТУ»), 2021

ВВЕДЕНИЕ

Одним из направлений развития современных информационных систем является их персонализация, адаптация под индивидуальные особенности каждого пользователя. Данный подход позволяет повысить гибкость информационных систем, удобство их использования для конечного потребителя, совместимость с различным программно-аппаратным обеспечением. Однако данный процесс в настоящее время все еще остается трудоемкой задачей, требующей привлечения значительных человеческих ресурсов. Особенно это актуально для небольших коллективов разработчиков, занимающихся проектами малого и среднего масштаба, однако имеющими высокую ценность для социальной, научной, образовательной, промышленной сфер человеческой деятельности.

Другим важным трендом развития информационных систем является автоматизация процессов анализа, обработки и передачи данных, автоматизация разработки компонентов информационных систем, что позволит снизить влияние человеческого фактора, сократить время и трудоемкость разработки.

Наконец, в настоящее время активно развиваются технологии искусственного интеллекта, машинного обучения для решения задач классификации, генерации, обработки данных. Актуальным направлением является исследование и формализация процесса интеграции данных технологий в процесс проектирования и функционирования информационных систем для решения конкретных прикладных задач, а также выявление, изучение возможностей и ограничений, разработка теоретического обоснования по применению методов машинного обучения в процессах анализа, обработки, генерации информации.

Таким образом, при разработке информационных систем перед разработчиками встает ряд нетривиальных задач: формализация предметной области, реализация архитектуры информационной системы, выбор методов проектирования, персонализация навигации и интерфейса под особенности каждого пользователя, стабильная работа на различных платформах и оборудовании, интеграция в существующую информационную среду, модернизация системы в соответствии с внешними или внутренними воздействиями.

Поэтому актуальным направлением научных исследований является разработка теории, моделей и методов автоматизированного структурно-параметрического синтеза адаптивных информационных систем (АИС), направленных на снижение стоимости и сложности

их разработки, требований к квалификации коллектива разработчиков, что позволит увеличить объем рынка и конкуренцию за счет привлечения новых участников, уменьшить сроки внедрения информационных систем, повысить степень автоматизации при их проектировании. Достижение перечисленных целей ускорит процессы цифровизации общества за счет использования и развития технологий искусственного интеллекта.

В рамках данной монографии рассматриваются современное состояние вопроса разработки АИС, подходы к анализу, обработке и генерации данных в них, реализации алгоритмов управления. На основе проведенного системного анализа предлагается новая методология структурно-параметрического синтеза АИС на основе нейросетевой архитектуры и методов.

Далее рассматривается апробация разработанных нейросетевых методов управления, анализа, обработки и генерации информации. Проведенные экспериментальные исследования доказывают применимость и эффективность предлагаемых методов.

Завершает монографию пример апробации методологии для реализации адаптивной системы электронного документооборота. Полученный в ходе практических исследований положительный эффект от перехода от классической архитектуры на нейросетевую, замены алгоритмических блоков на нейросетевые методы также подтверждает эффективность изложенных подходов.

Предлагаемый научный труд может быть полезен специалистам в области разработки адаптивных систем электронного документооборота, так как разработанная методология и методы применимы для различных предметных областей, в которых требуется автоматизировать процесс структурно-параметрического синтеза АИС, анализа, обработки и генерации данных, реализации систем управления. Изложенные методы универсальны и могут использоваться для решения прикладных задач в разных сферах информатизации общества и промышленности.

Глава 1
СОСТОЯНИЕ ВОПРОСА РАЗРАБОТКИ АДАПТИВНЫХ ИНФОРМАЦИОННЫХ СИСТЕМ

Адаптивными информационными системами (АИС) являются системы, в которых заложены возможности модификации алгоритмов их функционирования в ответ на действия пользователей или изменения характеристик внешней среды. В рамках данной главы рассматриваются основные подходы к разработке информационных систем и организации их архитектуры, реализация функций адаптивности и автоматизация разработки АИС.

1.1. АНАЛИЗ СУЩЕСТВУЮЩИХ МЕТОДОЛОГИЙ РАЗРАБОТКИ ИНФОРМАЦИОННЫХ СИСТЕМ

Разработка информационных систем в настоящее время подчиняется строгим правилам, условиям и принципам, собранным в рамках различных методологий реализации АИС. Это позволяет структурировать и упорядочить процесс разработки, избежать организационных ошибок, упростить планирование предстоящих работ, прогнозировать результат с некоторой вероятностью, что невозможно в условиях «свободной» разработки без какого-либо плана. Рассмотрим некоторые категории методологий разработки информационных систем.

Классическая «водопадная» модель

Концепция «водопадной» модели заключается в последовательном выполнении этапов анализа предметной области, описания задачи и требований, принятия спецификации, формирования технического задания, реализации проектного решения, тестирования [1].

«Водопадная» модель обладает следующими недостатками:

– значительное увеличение затрат и сбои в графике работы при итерационных возвратах для исправления недостатков, допущенных на предыдущих этапах;

– увеличение стоимости устранения ошибок при интеграции компонентов в конце разработки либо модификации отдельных модулей;

– отсутствие возможности внесения необходимой функциональности в систему на поздних этапах разработки;

– необходимость уже после сдачи проекта доработки системы ввиду новых требований, изменений в условиях эксплуатации, факторов внешней среды.

Так как для многих государственных организаций разработка информационных систем осуществляется в соответствии с требованиями

ГОСТа и по «водопадной» модели, полученные системы к моменту запуска могут быть устаревшими и не соответствовать текущим требованиям предметной области, а их окончательная доработка требует больших материальных и временных затрат.

Методологии на основе CASE-технологий

К данному типу относятся следующие типы методологий.

SADT (structured analysis and design technique) – методология функционального моделирования работ, которая основана на структурном анализе и графическом представлении организации как системы функций. Процесс разработки включает следующие этапы: экспертный анализ, формализация задачи в виде диаграмм и моделей, документирование, оценка адекватности моделей и их дальнейшее использование в процессе разработки. Также известна как нотация IDEF0 [2, 3].

DFD (data flow diagrams) – методология графического структурного анализа, описывающая внешние источники и адресаты данных, логические функции, потоки данных и хранилища данных, к которым осуществляется доступ. В качестве инструмента формализации используется стандарт UML и различные типы диаграмм. Модель DFD построена на иерархическом принципе и может быть декомпозирована на отдельные подмодели, при достижении требуемой степени декомпозиции используется текстовая спецификация [4].

ERD (entity-relationship diagrams) – модель данных, позволяющая описывать концептуальные схемы предметной области, используемая при концептуальном проектировании баз данных. Используется в разработке информационных систем для формализации модели данных (выделения объектов-сущностей и связей между ними) и ее последующем преобразовании в базу данных [5].

Достоинством CASE-технологий является высокая наглядность представления процесса разработки системы как в целом, так и отдельных ее частей. К недостаткам можно отнести: необходимость подготовки специалистов по CASE-технологиям, высокая стоимость программных инструментов для внедрения CASE, для больших проектов модели имеют высокую сложность описания, что приводит к проблемам восприятия общей задачи разработки.

Гибкие методологии разработки

Гибкая методология разработки (Agile software development) включает набор подходов к разработке программного обеспечения, ориентированных на использование итеративной разработки, динамическое формирование требований и обеспечение их реализации в результате постоянного взаимодействия внутри самоорганизующихся рабочих групп, состоящих из специалистов различного профиля.

Можно выделить следующие направления (методики) гибких методологий: экстремальное программирование, DSDM, Scrum, FDD, BDD [6].

Реализация гибких методологий заключается в выполнении следующих основных принципов: получение работающего продукта в максимально короткие сроки, возможность изменения проекта на любом этапе разработки, ориентация на людей и их взаимодействие. При использовании этой методологии на первый план выходят мотивация разработчиков, тесный контакт с заказчиком, работоспособность проекта, простота реализации, постоянная адаптация и готовность к изучению новых инструментов решения задачи.

Недостатками данного подхода являются отсутствие четкого плана разработки продукта, особенно в долгосрочной перспективе, возможность допущения и игнорирования архитектурных ошибок, которые повлекут значительные временные и материальные затраты на последующих этапах разработки.

Методология RUP

В методологии RUP реализуются итерационный и наращиваемый (инкрементный) подходы [7]. Построение системы происходит на базе некоторой архитектуры, а планирование и проектное управление – на базе функциональных требований. Разработка осуществляется итерационно, как набор отдельных небольших проектов.

Предлагаемый подход обладает следующими преимуществами:

– декомпозиция системы на модули ускоряет итерационную разработку общей функциональности системы;

– итерационный подход облегчает поэтапное управление проектом;

– модульный принцип позволяет вносить изменения в продукт на различных этапах разработки с меньшими временными потерями;

– анализ каждой итерации позволяет лучше оценить квалификацию разработчиков и затраты на весь процесс разработки проекта.

Однако, одной из особенностей методологии RUP является исключение этапа анализа и описания предметной области. Это оправдано для небольших информационных систем, но если речь идет о социально значимых системах, то данный подход не только является одним из первых, но и проводится в течение всего процесса разработки.

Концепция быстрой разработки приложений (RAD)

Концепция быстрой разработки приложений (Rapid Application Development, RAD) заключается в итеративном и инкрементном подходе к разработке, который придает особое значение продолжительному участию в процессе пользователя/потребителя. В рамках данной методологии основная цель – сдать готовый проект в установленные

сроки с заданным бюджетом, учитывая возникающие требования к проекту в процессе проектирования и реализации [8].

Принципы RAD-технологии направлены на обеспечение трех основных ее преимуществ – высокой скорости разработки, низкой стоимости и высокого качества. Достоинством методологии RAD является быстрая разработка, клиентоориентированность, легкая адаптируемость проектов, простота развития функциональности системы.

RAD-технология не является универсальной и имеет следующие ограничения: сжатые сроки на реализацию проекта; постоянно меняющиеся требования заказчика и его включение в процесс разработки на ранних этапах; необходимость в декомпозиции большого проекта на отдельные функциональные компоненты; необходимость формирования графического интерфейса на ранних этапах разработки; применение готовых решений и библиотек, а не разработка оригинальных алгоритмов при решении сложных вычислительных задач.

Таким образом, имеем следующие недостатки методологии RAD: для больших проектов необходим большой коллектив разработчиков; система должна декомпозироваться на отдельные модули; требует привлечения пользователя (заказчика) на всех этапах разработки; высокие требования к квалификации разработчиков; жесткие временные ограничения; высокие риски невыполнения проекта.

Проведенный анализ позволяет сделать следующий вывод. Несмотря на широкое распространение и эффективность данных методологий, задача автоматизации процесса разработки информационных систем в них не решается. Их применение позволяет снизить затраты материальных или временных ресурсов, оптимизировать процессы реализации систем, но универсальными и однозначно применимыми подходами они не являются, так как каждая задача и предметная область накладывают свои ограничения.

1.2. ОБЗОР СУЩЕСТВУЮЩИХ АРХИТЕКТУР ИНФОРМАЦИОННЫХ СИСТЕМ

После выбора соответствующей методологии разработки необходимо определиться с архитектурой информационной системы. В ряде случаев методология регламентирует или ограничивает этот выбор. Рассмотрим основные типы архитектур и их применимость при решении задачи разработки АИС.

Архитектурные шаблоны

Шаблоны (паттерны) не являются в полном смысле архитектурой приложения, но содержат в себе конструкции, позволяющие решить некоторые задачи разработки АИС. Таким образом, объединение не-

скольких шаблонов позволяет решать более широкие наборы задач. Рассмотрим несколько примеров архитектурных шаблонов.

«Модель–Представление–Контроллер» (MVC). Отделяет интерфейс от модели данных, позволяя изменять его, не нарушая процессов передачи, сохранения и чтения информации. Таким образом, все данные хранятся в Модели, их визуальное представление и элементы взаимодействия с системой – в Представлении, а для связи между ними реализуется Контроллер. Достоинством шаблона является возможность создания различных моделей и представлений и их автоматический выбор в зависимости от сценариев использования. Однако применение данного шаблона приводит к усложнению структуры приложения.

Посредник. При наличии множества связей между модулями их поддержка и взаимодействие усложняются, поэтому в систему добавляется посредник, через которого все модули (объекты) обмениваются информацией. Недостатком является снижение производительности и зависимость работы всей системы от одного компонента (посредника). С другой стороны, такая реализация позволяет упростить расширение и модернизацию системы.

Абстрактная фабрика. Шаблон предоставляет интерфейс для создания семейств взаимосвязанных объектов, не формализуя особенности отдельных классов. Шаблон реализуется посредством некоторого интерфейса или абстрактного класса, используемого для генерации компонентов системы. Для каждого компонента реализуется свой класс. Таким образом, разработав семейство классов, реализующих компоненты, можно предоставить лишь их интерфейс, а не полную реализацию. Объекты внутри семейства изолированы и выполнены по единому шаблону, что обеспечивает их совместимость. Система работает независимо от тех объектов, что создает абстрактная фабрика. Однако появление нового типа объектов приводит к необходимости созданий новой фабрики.

Шина (канал) событий. В этом шаблоне используется концепция прослушивания событий. Источники размещают некоторые события на шине по отдельным каналам, после чего слушатели, подписанные на эти каналы, могут получить уведомления о их наступлении. Достоинством данного подхода является легкость реализации и высокая пригодность для распределенных приложений. Однако шина событий является узким местом системы, что может привести к проблемам с производительностью при масштабируемости.

Ведущий–ведомый. Основан на взаимодействии двух участников: ведущий распределяет задачи между ведомыми и анализирует полу-

ченные от них результаты. При этом можно достигнуть оптимального распределения задач, однако между ведомыми отсутствуют какие-либо связи.

Клиент-серверный шаблон. При необходимости распределения ограниченного объема ресурсов между компонентами системы можно выделить один или несколько модулей как сервер, а остальным присвоить статус клиентов, запрашивающих данные у сервера при необходимости. Такой подход повышает масштабируемость и доступность системы, все данные и вычисления можно сосредоточить на сервере, обладающем большей мощностью и защитой. Минусом является необходимость отделения запросов между клиентами и серверами в отдельные потоки, а также неработоспособность клиентов в случае недоступности сервера.

Отметим, что не были рассмотрены многие разновидности архитектурных шаблонов, несомненно, представляющих интерес. Это такие распространенные подходы, как Одиночка (Singleton), Прототип (Prototype), Строитель (Builder), Адаптер (Adapter), Декоратор (Decorator), Компоновщик (Composite), Фасад (Facade), Итератор (Iterator), Наблюдатель (Observer), Шаблонный метод (Template method) и многие другие.

Таким образом, архитектурные шаблоны могут использоваться в рамках общей архитектуры АИС для решения конкретных задач широкого профиля. Их объединение и модернизация позволяют гибко формировать архитектуру приложения, используя заложенные в шаблонах стандарты, как каркас системы.

Монолитная архитектура

При монолитной архитектуре системы все компоненты объединены в единое целое на базе одной платформы. Это значительно повышает связанность и зависимость компонентов между собой. Классическим подходов к реализации монолитного приложения является разделение его на базу данных, серверную и клиентские компоненты. Данная архитектура является традиционной при разработке программного обеспечения.

Достоинствами монолитной архитектуры являются:

– удобство при работе небольшими командами разработчиков;

– все компоненты расположены на единой платформе, что упрощает их интеграцию и обмен данных;

– все компоненты системы могут быть обновлены одновременно;

– упрощение связей между модулями благодаря единой программной базе, сокращение проблем с совместимостью разных элементов системы;

– высокая производительность за счет общей программной базы, памяти и ресурсов.

Однако монолитная архитектура имеет ряд существенных недостатков:

– общая программная база расширяется и становится громоздкой, а структура – слишком тяжелой для понимания, что приводит к сложности поддержки приложения;

– сложность модернизации и внедрения новых технологий, так как часть компонентов приложения может быть несовместима с ними, что приведет к разработке нового приложения;

– низкая гибкость, каждое обновление приводит к перезапуску всей системы и недоступности всех функций и компонентов системы.

Сервис-ориентированная архитектура

Основой сервис-ориентированной архитектуры является декомпозиция системы на множество модулей (сервисов), слабо связанных между собой стандартизированными протоколами, легко заменяемых, решающих отдельные конкретные задачи. Реализация каждого модуля может быть выполнена на различных платформах, языках программирования, фреймворках, так как она неизвестна для остальных компонентов. Это позволяет распределить реализацию системы между множеством команд разработчиков с разными стилями программирования, облегчить масштабируемость системы, обеспечить постоянное развитие и модернизацию компонентов. При реализации сервис-ориентированных систем используются такие технологии, как REST, RPC, DCOM, CORBA, веб-сервисы и др.

Достоинства сервис-ориентированной архитектуры заключаются в следующем:

– масштабирование приложений;

– распределение задач;

– безопасность;

– распределение ресурсов на каждый сервис.

Недостатки сервис-ориентированной архитектуры:

– сложность нахождения ошибок и тестирования;

– согласованность данных при постоянном обмене информацией между сервисами;

– рост количества трафика за счет обмена данными между микросервисами.

Событийно-ориентированная архитектура

Данная архитектура ориентирована на системы, состоящие из слабосвязанных компонентов, управляемых на основе возникающих

событий. Таким образом, архитектура включает источники события и потребителей, реагирующих на них. Реакция может включать анализ, обработку или дальнейшую передачу данных. При таком подходе реализуемая система обладает большой интерактивностью и гибкостью.

Достоинства архитектуры:

– асинхронность и независимость модулей;
– легкая масштабируемость;
– применимость для организации связи между различными системами.

Однако асинхронность архитектуры приводит к значительному усложнению отладки приложений, так как события могут запускать последовательности операций сразу у нескольких модулей, что усложняет протоколирование и отслеживание ошибок.

Многослойная архитектура

В основе многослойной (многоуровневой) архитектуры лежит клиент-серверный подход с разделением функций по слоям (уровням). Зачастую используется архитектура из трех уровней (представление, обработка и хранение данных), однако общее их число неограниченно. Декомпозиция задач на уровни позволяет вносить изменения только в один из них, не затрагивая остальные уровни. Такая декомпозиция упрощает разработку, однако усложняет структуру и снижает производительность.

Достоинством многослойной архитектуры является четкое разделение функций по уровням, возможность независимой разработки и модернизации уровней без ущерба другим слоям.

К недостаткам относится низкая скорость работы из-за необходимости передавать большие объемы данных от слоя к слою (часть передач не несут полезной нагрузки). Наличие несколько слоев абстракций усложняет структуру программы и ее поддержку. Отладка многослойных приложений также связана с некоторыми трудностями: если информация проходит через несколько уровней, то ошибка может возникнуть на каждом из них.

Проведенный анализ архитектур информационных систем позволил выявить некоторые ключевые особенности, которые могут быть полезны при автоматизации их разработки.

Во-первых, необходимо отметить сервис-ориентированную архитектуру, содержащую эффективные решения по декомпозиции задач обработки данных и реализации модулей системы. Часть сервисов может быть в дальнейшем заменена на автоматизированные компоненты с развитием методов и технологий обработки данных, что позволит

использовать в качестве основы уже работающие, функционирующие системы.

Во-вторых, рассмотренные архитектурные шаблоны могут активно использоваться для решения отдельных задач при реализации некоторых модулей, интегрироваться в общую архитектуру системы. Это значительно повысит ее гибкость и эффективность за счет применения оптимальных подходов.

Наконец, проведенный анализ показал, что не существует идеальной архитектуры, которую можно применить для реализации большинства информационных систем. Ряд архитектур имеет серьезные проблемы с масштабируемостью и гибкостью, что для АИС является существенным недостатком.

Поэтому перспективным решением выглядит синтез новой архитектуры АИС, учитывающей сильные стороны рассмотренных подходов и ориентированной на автоматизацию процессов анализа, обработки и передачи данных.

1.3. АНАЛИЗ ПОДХОДОВ К РЕАЛИЗАЦИИ ФУНКЦИЙ АДАПТИВНОСТИ В ИНФОРМАЦИОННЫХ СИСТЕМАХ

Проблема разработки удобных, адаптированных интерфейсов для пользователей существует достаточно давно, так как в данном вопросе имеются противоречия между взглядом разработчиков на универсальный, эффективный интерфейс и субъективными пожеланиями конечных пользователей [9]. Кроме того, бизнес-процессы организации находятся в постоянном развитии и изменении, что требует соответствующей модификации информационной системы, в том числе и интерфейса. Автоматизация данного процесса позволила бы значительно сократить временные и материальные затраты в процессе разработки и модернизации информационной системы. Изложенный в статье [9] сравнительный анализ подходов показал, что единственного и универсального решения для построения интерфейсов не представлено, часть методов эффективна только на этапе проектирования, часть позволяет подробно формализовать требования к интерфейсу, но не позволяет построить его.

Алгоритм адаптации интерфейса рассматривается в статье [10]. При выборе оптимальных параметров авторы рекомендуют оценивать когнитивные особенности пользователей, а также использовать конструкторы интерфейсов. Выбранные пользователями параметры интерфейса сохраняются в базу данных для последующего анализа и повторного использования в других системах или модулях.

В работе [11] рассмотрен набор эргономических критериев для оценки интерфейсов. К ним относятся 18 метрик, объединенных в 8 категорий (табл. 1.1).

1.1. Эргономические критерии

Категория	Метрики
Доступность руководства пользователя	1. Подсказки и справочные инструменты 2. Понятность группировки элементов интерфейса по местоположению 3. Понятность группировки элементов интерфейса по формату и стилю 4. Качество обратной связи 5. Удобочитаемость (разборчивость) интерфейса
Загруженность интерфейса	6. Лаконичность интерфейса 7. Минимизация действий для решения задачи 8. Плотность информации
Контроль пользователя над выполняемыми операциями	9. Явное выполнение системой действий, заданных пользователем 10. Пользовательский контроль за выполнением операций ввода, обработки и вывода информации
Адаптивность	11. Гибкость интерфейса для различных групп пользователей 12. Адаптация системы под различный пользовательский опыт
Управление ошибками	13. Защита от ошибочных действий пользователя 14. Качество сообщений о допущенных ошибках 15. Качество исправления ошибок
Согласованность	16. Согласованность дизайна, расположения элементов, реакций системы на действия в различных модулях
Значение идентификаторов (кодовых имен)	17. Смысловая значимость идентификаторов (кодовых имен) для пользователя, облегчающая выполнение операций
Совместимость	18. Совместимость между индивидуальными характеристиками пользователя и способами организации диалога и операций ввода-вывода

Использование данных критериев при экспертной оценке позволяло повысить ее объективность и точность, а также выявить ряд проблем в анализируемом интерфейсе программного продукта, не обнаруженных ранее.

Для автоматизации процесса адаптации интерфейса необходима формализация характеристик пользователя в виде некоторой модели [12]. В данной работе представлено несколько существующих подходов по сбору данных о пользователе и их дальнейшем анализе, причем помимо информации непосредственно о пользователях рекомендуется оценивать характеристики оборудования и используемое программное обеспечение.

Алгоритмы и модели адаптации интерфейса в области сенсорного взаимодействия пользователя с информационными системами представлены в работе [13]. Однако авторы столкнулись при решении задачи с несколькими проблемами: необходимость автоматической настройки интерфейса после смены пользователя и необходимость сбора большого объема статических данных, без которых затруднительно осуществить качественный прогноз предпочтений пользователей.

В работе [14] рассмотрена модель навигационного интерфейса. Авторами выделены две основные составляющие – оформление (дизайн) и навигационная структура. Оценка дизайна может быть субъективной и зависеть от личных предпочтений и жизненного опыта пользователей, с другой стороны, навигационная структура может быть оценена с позиции соответствия ментальной модели пользователя (когнитивности).

Большой интерес представляет математическая модель стратегии построения адаптивного интерфейса, изложенная в работе [15]. Авторами также представлен алгоритм адаптивного человеко-машинного интерфейса, основанный на сборе информации о пользователе и последующей адаптации системы, и критерии оценки интерфейса:

- время от момента обращения к системе до получения доступа к ней;
- функциональность;
- гибкость;
- защищенность интерфейса от ошибок;
- доступность;
- комфортность.

Совокупность критериев влияет на два важных для интерфейса показателя: способность адаптации пользователя к возможностям системы и способность адаптации системы к индивидуальным особенностям пользователя. Далее авторами предлагается методика расчета

данных показателей, проводится исследование закономерностей влияния критериев на показатели адаптации, после чего формируется методика выбора структуры адаптивного интерфейса с использованием множества критериев оценки. Таким образом, можно отметить высокую научную и практическую ценность изложенных в работе положений.

Одним из вариантов адаптации интерфейса является использование сложной системы правил [16]. Авторами данной работы предлагается частично автоматизированный метод генерации адаптированных интерфейсов, основанный на анализе предпочтений пользователя и их последующей обработке множеством заданных правил. В работе [17] также используется система правил для создания интеллектуальных пользовательских интерфейсов в реальном времени. Однако недостатком данных работ является статичность системы правил – ее расширение не происходит автоматически в процессе работы.

Перспективным направлением в области адаптации интерфейса под индивидуальные особенности пользователя является использование методов машинного обучения, например нейронных сетей. Например, в работе [18] авторами успешно решена задача выбора компонентов интерфейса с применением системы рекомендаций на основе методов машинного обучения. В качестве предметной области использовалось приложение для контроля за состоянием окружающей среды. Осуществив обучение алгоритмов машинного обучения на исходных данных о пользователях (возраст, должность, оборудование), времени года и части дня, в которые он взаимодействовал с системой, авторы реализовали систему рекомендаций компонентов для новых пользователей на основе предпочтений уже зарегистрированных пользователей.

Задача динамической адаптации интерфейса в зависимости от местоположения человека рассмотрена в работе [19] на примере информационной системы для медперсонала. Местоположение персонала определялось путем сканирования NFC-меток в каждом помещении, полученная информация обрабатывалась на классификаторе методом дерева решений. Реализованный классификатор смог с точностью выше 80% предсказывать текущее местоположение медперсонала (уже без использования в процессе работы сканирования меток). На основе этого местоположения на устройстве сотрудника формируется нужный ему в данный момент интерфейс. Данное исследование интересно тем, что в его рамках рассмотрены аспекты построения архитектуры такой информационной системы, интеграция различных средств программирования и технологий машинного обучения.

Подводя итог анализу подходов к адаптивности в информационных системах, можно выделить несколько основных направлений ра-

боты. Во-первых, необходима формализация модели пользователя в виде совокупности ключевых его характеристик. Во-вторых, формализованное представление критериев адаптивности систем требует дальнейшего развития и доработки в целях повышения их объективности. Далее, требуется более глубокое исследование возможности применения нейронных сетей для адаптации интерфейсов информационных систем, организация сбора и анализа больших объемов данных. И, наконец, необходимо обеспечить тесную интеграцию разработанных методов, алгоритмов и моделей в АИС.

1.4. АНАЛИЗ ПОДХОДОВ К АВТОМАТИЗАЦИИ РАЗРАБОТКИ АДАПТИВНЫХ ИНФОРМАЦИОННЫХ СИСТЕМ

При разработке информационных систем перед разработчиками встает ряд нетривиальных задач: анализ, формализация и декомпозиция протекающих в предметной области процессов [20], формирование структурных и функциональных моделей системы [21], реализация процессов управления и обработки информации, персонализация навигации и интерфейса под особенности каждого пользователя [22], стабильная работа на различных платформах и оборудовании, интеграция в существующую информационную среду, модернизация системы в соответствии с внешними или внутренними воздействиями.

Однако реализация гибких, персонализированных, адаптивных информационных систем связана с длительным и дорогостоящим процессом разработки, что снижает коммерческий интерес к подобным системам [23]. Особенно это актуально для небольших коллективов разработчиков, занимающихся проектами малого и среднего масштаба [24]. Для корпораций данный вопрос стоит менее остро, так как за счет размера и количества персонала положительный эффект от разработки и внедрения адаптивных информационных систем перекрывает экономические затраты [25, 26].

Вопросам автоматизации процессов обработки и управления информации посвящены многие исследования [27]. Их решение необходимо для реализации качественных экспертных систем, систем поддержки принятия решений, электронных архивов, систем электронного документооборота, разнообразных модулей по обработке информации в информационных системах широкого профиля.

Несмотря на возможность формализации процессов, протекающих в информационных системах, с использованием различных математических моделей [28], необходимо отметить их слабые стороны в области решения задачи автоматизации разработки информационных систем. Исследованию этой проблемы, например, в рамках структур-

но-параметрического синтеза информационных систем не уделяется достаточного внимания. С одной стороны, данная задача является NP-сложной, не существует алгоритмов, позволяющих найти оптимальное решение за приемлемое время, более того, сама формализация данного процесса затруднительна [28]. До сих пор задача формирования структуры информационных систем решается аналитически. С одной стороны, без влияния человеческого фактора получить оптимальную архитектуру системы затруднительно, с другой, субъективные оценки могут привести к ошибочному решению. Именно поэтому задача снижения влияния экспертной оценки на конечную структуру информационных систем является весьма актуальной.

Тем не менее постоянное развитие информационных технологий и перспективы использования квантовых компьютеров для решения сверхсложных задач позволят со временем решать и подобные задачи. Поэтому актуальной задачей является теоретическая проработка вопроса автоматизации реализации информационных систем, подготовка алгоритмического и математического обеспечения, на основе которого в дальнейшем будет возможно проведение экспериментальных исследований и получение конкретных оптимальных решений.

Анализ научных источников показал достаточно слабую проработку рассматриваемой проблемы в различных предметных областях. Например, в области систем электронного документооборота можно отметить работу [29], в которой авторами рассматривается методология эволюционного структурно-параметрического синтеза имитационных моделей. Предложенный подход предусматривает возможность синтеза программно-аппаратного обеспечения, нахождение оптимальной конфигурации оборудования путем использования аппарата сетей Петри для моделирования процессов взаимодействия компонентов [30, 31]. Несмотря на высокую перспективность данных исследований и их применимость для решения задач структурно-параметрического синтеза информационных систем, данный подход имеет свои недостатки: большие временные затраты на поиск оптимального решения (особенно в задачах структурной оптимизации); плохая масштабируемость генетических алгоритмов под размер области поиска решений (это потребует декомпозиции информационной системы на подсистемы, что приведет к проблеме совместимости и защиты высокоэффективных решений от низкоэффективных); генетические алгоритмы зачастую находят лишь локальный минимум, а не глобальный.

Для решения задачи автоматизации структурно-параметрического синтеза во многих предметных областях зачастую рекомендуется метод морфологического ящика или подобные ему [32]. Несмотря на большое количество научных работ, посвященных данному методу,

реальные практические исследования проводились редко. Данный подход успешно позволил решить задачу нахождения оптимальной конфигурации оборудования в рамках многокритериальной задачи оптимизации, однако на данном этапе рассматриваемый подход не применим для оценки структуры программного обеспечения, требует переосмысления и адаптации. Напротив, в статье [32] при применении морфологического подхода большое внимание уделяется непосредственно этапу поиска рациональной структуры на основе технического задания, а уже после – проведению параметрического синтеза для выбранной структуры. Суть подхода заключается в построении морфологической таблицы, заполнении ее возможными альтернативными вариантами и в выборе оптимального варианта.

Если рассматривать общие подходы к автоматизации структурно-параметрического синтеза, то можно отметить работу [33], где затронут важный вопрос структурно-параметрического синтеза, но не объектов, а их математических моделей. Исследовалась возможность автоматического подбора нужной функциональной зависимости по совокупности экспериментальных данных. Для обеспечения автоматического выбора структуры предлагается определить все возможные математические модели с использованием необходимых функциональных преобразований, причем моделей с одинаковыми типами функциональных преобразований в этом наборе быть не должно.

Проведенный анализ подходов к автоматизации разработки информационных систем показал, что универсального и полного решения этой задачи не существует, в каждой предметной области предлагаются свои методы, позволяющие частично автоматизировать процесс синтеза сложных систем. Однако анализ источников позволил сформулировать некоторые рекомендации для решения задачи автоматизации разработки информационных систем. Для осуществления автоматизированной разработки необходимо реализовать следующие компоненты:

– универсальная архитектура разрабатываемых информационных систем: позволяет формализовать структуру модулей и параметры множества информационных систем с указанием тех задач, на решение которых они направлены;

– автоматизация процесса взаимодействия и передачи данных между модулями;

– автоматический компилятор модулей информационной системы;

– система целевых функций, позволяющая оценивать характеристики системы (как структуру, так и параметры);

– блок выбора структуры и параметров системы по целевым функциям.

Таким образом, можно выделить ряд ключевых задач: необходимость разработки общей, универсальной модели архитектуры системы, позволяющей описать целый класс подобных информационных систем; определение целевых функций; формирование блока структурно-параметрического синтеза с использованием современных технологий, в том числе экспертных систем, систем поддержки принятия решений, технологий искусственного интеллекта.

Для повышения уровня автоматизации при решении задачи разработки АИС требуется снизить влияние человеческого фактора и необходимость принятия аналитических решений. Предполагаемое решение должно включать в себя возможность автоматизировать подбор структур элементов системы и их параметров, однако в данный момент из-за NP-сложности данной задачи решить ее численными методами невозможно. Однако ее можно частично автоматизировать за счет решения следующих подзадач:

– реализация инструмента формализации и хранения данных об информационных потоках организации на различных уровнях детализации;

– разработка новой оригинальной концепции архитектуры, позволяющей представить структуру информационной системы таким образом, чтобы ее элементы можно было изолировать друг от друга, в том числе отделить программное обеспечение (управляющие элементы и представление информации) от самих данных и набора требуемых функций, после чего решить задачу автоматизации каждого компонента по отдельности;

– разработка набора автоматизированных методов анализа, обработки и передачи информации, что позволит снизить общую нагрузку на разработчиков в ходе реализации АИС.

Отметим, что ни одна из ранее разработанных математических или информационных моделей не подходит для решения перечисленных задач, так как требуемая концепция должна объединить воедино программные составляющие информационной системы, особенности предметной области, структуру информационных потоков, набор необходимых функций, различные варианты представления данных, а также учитывать влияние пользователя и характеристики тех устройств, с помощью которых он взаимодействует с системой.

Таким образом, анализ современного состояния исследований в данной предметной области показал, что с использованием существующих математических моделей и подходов решить задачу автоматизации разработки адаптивных информационных систем невозможно, поэтому необходимы переход к качественно иной архитектуре и реализация новой методологии разработки АИС.

1.5. АНАЛИЗ ПОДХОДОВ К ПРИМЕНЕНИЮ МАШИННОГО ОБУЧЕНИЯ В АДАПТИВНЫХ ИНФОРМАЦИОННЫХ СИСТЕМАХ

В рамках исследования вопроса автоматизации процесса разработки АИС рассмотрим основные технологии машинного обучения, применяемые для автоматизации анализа, обработки, генерации информации, организации процессов управления в информационных системах.

Технологии машинного обучения активно развиваются последние десятилетия, их возрастающая эффективность обусловлена появлением новых практических подходов и программных методов, а также значительно выросшей производительностью компьютеров. Таким образом, постоянно расширяется и ряд задач, решение которых возможно с применением машинного обучения. Рассмотрим основные направления в этой области [99 – 101]:

1. Регрессия – анализ или прогноз на основе выборки объектов с различными признаками. На основе входных данных строится предположение о выходном значении, выраженном вещественным числом.
2. Классификация – определение класса (категории) объекта по значениям его признаков. Обычно выражается в вещественном значении от 0 (не является представителем класса) до 1 (совершенно точно является). Используется для распознавания цифр, объектов, определения наличия или отсутствия объектов на изображениях, определения тональности или тематики текстов и т.д.
3. Кластеризация – разделение данных на группы по их признакам без заранее определенного перечня этих групп. При таком подходе методы машинного обучения могут выявить закономерности группировки объектов по ранее незамеченным признакам.
4. Уменьшение размерности – переход от большого числа признаков к меньшему для удобства их последующей визуализации и упрощения анализа данных.
5. Выявление аномалий – отделение аномалий от стандартных случаев. В отличие от задачи классификации направлено на выявление редких, аномальных случаев в анализируемых данных, количество которых в обучающей выборке может быть мало либо просто отсутствовать.

1.5.1. Обзор технологий машинного обучения

Алгоритмы машинного обучения с учителем

В основе алгоритмов машинного обучения с учителем лежит принцип «стимул–реакция». То есть для каждого входного набора

(«стимула») задается соответствующее выходное значение («реакция»). Зависимость между входом и выходом неизвестна, однако задана обучающая выборка, содержащая конечный набор пар «стимул–реакция». На основе анализа обучающей выборки алгоритмы машинного обучения с учителем восстанавливают зависимость между входными и выходными данными. Примерами таких алгоритмов являются [102]:

- Метод Байеса [103].
- Метод k ближайших соседей [104].
- Метод опорных векторов [105].
- Метод деревьев решений [106].

Так же алгоритмы обучения с учителем включают большой класс методов, основанных на нейронных сетях, но данное направление будет рассмотрено ниже в рамках отдельного раздела.

Алгоритмы машинного обучения без учителя

В обучении без учителя (unsupervised learning) у модели есть набор данных, и нет явных указаний, что с ним делать. Алгоритмы машинного обучения извлекают полезные признаки и анализируют их, не учитывая остальные признаки [102,107]. К данному направлению относятся задачи кластеризации и обнаружения аномалий, а также следующие:

Ассоциации. На основе нескольких признаков объекта выбираются другие объекты, с которыми у него может быть связь. Так работают рекомендательные алгоритмы интернет-магазинов.

Автоэнкодеры. На основе входных данных формируется некоторое кодированное представление, из которого затем воссоздаются исходные данные. Таким образом можно генерировать картины, убирать шум с видео и изображений.

Большой проблемой данного направления является сложность определения точности полученной модели, правильности результата, так как нет набора размеченных данных с известными выходными значениями. С другой стороны, данные алгоритмы позволяют получать интересные, полезные результаты в тех областях, где алгоритмы обучения с учителем бессильны.

Нейронные сети

Нейронные сети активно используются в связи с появлением больших объемов данных и больших вычислительных возможностей. Сети с прямой связью являются универсальным средством аппроксимации функций, что позволяет их использовать при решении задач обработки информации.

В последние годы направление машинного обучения – нейронные сети – активно развивается. Нейронные сети успешно применяются

для решения следующих задач: классификация, предсказание, распознавание, генерация новых данных [108]. Постоянно появляются новые архитектуры и подходы к построению нейронных сетей. Можно выделить следующие основные типы нейронных сетей [109]:

- Нейронные сети прямого распространения и перцептроны.
- Сети радиально-базисных функций.
- Нейронная сеть Хопфилда.
- Цепи Маркова.
- Машина Больцмана.
- Автокодировщик.
- Разреженные и вариационные автокодировщики.
- Сверточные нейронные сети.
- Развертывающие нейронные сети.
- Капсульная нейронная сеть.
- Глубинные сверточные обратные графические сети.
- Генеративные состязательные сети.
- Рекуррентные нейронные сети.
- Сети с долгой краткосрочной памятью.
- Управляемые рекуррентные нейроны.
- Нейронные сети Кохонена.

Таким образом, количество разнообразных типов нейронных сетей и методов машинного обучения позволяет эффективно применять их при решении различных задач по анализу и обработке данных. Рассмотрим конкретные примеры этого применения и их возможную адаптацию под предметную область АИС.

1.5.2. Применение машинного обучения для анализа информации

Рассмотрим основные задачи анализа информации в АИС, решение которых возможно с применением машинного обучения.

Распознавание категории информации (файла). Классическая задача классификации, основанная на анализе текста, численных значений, атрибутов и содержимого файлов [110]. Численные данные нормируются, текстовые – проходят через токенизацию и лемматизацию, из файлов извлекаются метаданные и информация. Для аудио- и видеоданных осуществляется преобразование информации в спектрограммы и последовательности кадров. Возможно выделение приоритетных блоков информации, так как это позволяет снизить вычислительную нагрузку и повысить точность классификации.

Распознавание ошибочных данных. В данной задаче также осуществляется классификация входной информации, однако ее результа-

том является не только определение конкретного класса объекта, но и проверка, насколько вероятность принадлежности определенной категории совпадает с реальным классом объекта [111].

Распознавание вредоносной информации, файлов и программ. В данном классе задач методы машинного обучения используются для классификации информации по критерию безопасности. На основе анализа структуры и содержимого информации, действий программ или пользователей осуществляется принятие решения о потенциальной опасности для АИС [112].

Распознавание дубликатов файлов. В случае электронного архива возможны ситуации, когда разным метаданным файлов соответствуют одинаковые или сходные файлы. Тогда встает задача поиска дубликатов или файлов, имеющих ряд сходных признаков, которая может быть решена при использовании нейронных сетей для классификации файлов и выделения их ключевых признаков с помощью анализа значений промежуточных слоев [113].

Маршрутизация и распределение данных. Нейронные сети можно использовать для анализа нагрузки на узлы сети, информационной системы или пользователей, что позволяет оптимизировать движение информационных потоков и повысить эффективность их работы [114]. Так же в данную категорию можно отнести автоматизацию процесса выбора протоколов передачи данных.

Поиск скрытых связей и закономерностей. Данная задача включает ряд серьезных вопросов, связанных с анализом неразмеченных данных, кластеризацией, изучением признаков информационных объектов и нахождением связей между входными и выходными переменными. Нейронные сети и другие методы машинного обучения позволяют выявить эти закономерности в ходе анализа большой выборки исходных данных без необходимости построения формализованных аналитических решений и формул.

1.5.3. Применение машинного обучения для обработки информации

Следующим важным направлением использования технологий машинного обучения в АИС является обработка и преобразование данных. В области обработки информации можно выделить следующий набор задач.

Обработка численных данных. Нейронные сети и технологии машинного обучения могут использоваться для решения регрессионных задач, так как по теореме Цыбенко [115] двухслойная нейронная сеть может аппроксимировать любую непрерывную функцию многих пе-

ременных. Кроме того, вторая теорема Хехт-Нильсена [116] и следствие из теоремы Колмогорова–Арнольда [117] доказывают, что для любого множества пар векторов произвольной размерности существует трехслойная нейронная сеть с конечным числом нейронов, формирующая для каждого входного вектора соответствующий ему выходной вектор. Данные свойства нейронных сетей позволяют не искать аналитические решения (что может быть затруднено наличием множества скрытых факторов и закономерностей), а использовать модель «черного ящика» в виде нейронной сети, обеспечивающую достаточную точность преобразования данных. Так же, если некоторая зависимость переменных аппроксимируется с помощью сплайна или полинома некоторой степени, то она может быть заменена обученной нейронной сетью. Перечисленные возможности позволяют упростить процесс обработки данных, так как в ряде задач позволят заменить привлечение экспертов для нахождения сложных аналитических решений на более простую реализацию алгоритмов машинного обучения.

Обработка текстовых данных. Технологии машинного обучения позволяют реализовать различные преобразования над текстовыми данными. Рассмотрим основные подзадачи в этой области. Нейронные сети эффективно применяются для осуществления машинного перевода с одного языка на другой с возможностью передачи естественной формы предложений. В поисковых системах данные технологии используются для выдачи релевантных результатов на основе обработки цифрового профиля пользователя, автоматической коррекции ошибок в запросах и выдачи связанной информации. При обработке документов нейронные сети позволяют осуществить стилевое оформление текста, его переформатирование, обработку текста с заменой необходимых фрагментов, в том числе автоматически заполнить шаблоны документов [49].

Обработка графических данных. В данной области нейронные сети активно применяются при цифровой обработке фотографий в мобильных камерах, позволяя повышать качество, резкость и детализацию изображений в сложных условиях. Так же подобные алгоритмы успешно используются для увеличения разрешения изображений без потери качества и эффекта размытия. Использование сверточных нейронных сетей позволяет выделить основные стилевые и структурные признаки изображений и перенести их на другие графические объекты, таким образом, осуществив перенос стиля [118]. Большое прикладное значение имеет применение технологий машинного обучения при распознавании рукописного текста [119].

Обработка аудиоданных. Распространение технологий машинного обучения значительно расширило возможности по обработке аудиоданных. Это привело к развитию голосовых помощников, общающихся с пользователем с естественными интонациями, улучшению качества распознавания речи, возможности озвучивания текста, а также продвинутой обработке звука (устранение шумов, искажений, повышение качества, удаление речи, замена стилевого оформления и т.д.). Нейронные сети выделяют ключевые признаки из исходных аудиоданных, причем для решения этих задач могут использоваться сверточные сети, если в качестве входных данных будет поступать графическое представление в виде спектрограмм [120].

Обработка видеоданных. Комбинация подходов по обработке графических и аудиоданных позволила реализовать новые приемы и технологии в сфере видеообработки. К ним относится повышение качества и разрешения видеосигнала, более эффективные методы кодирования, стилизация, а также новые реалистичные методы замены объектов (например, лиц) на видео. Разработанные методы машинного обучения успешно позволяют распознавать, классифицировать и удалять объекты в кадре, превратить изображение в видео с возможностью имитации различных действий [121].

Таким образом, рассмотренные задачи по обработке данных в различных информационных системах позволяют сделать следующий вывод: нейронные сети и алгоритмы машинного обучения нашли свое применение при преобразовании данных самых различных типов (численных, текстовых, графических, аудио- и видеофайлов) и доказали свою эффективность.

1.5.4. Применение машинного обучения для генерации информации

В рамках предметной области разработки АИС под термином «задача генерации информации» будем понимать следующий процесс: формирование нового информационного объекта в АИС с применением различных технологий обработки данных. К основным направлениям задач генерации информации отнесем следующие четыре направления: подготовка тестовых данных, восстановление поврежденной информации, прогнозирование и анализ состояния объектов.

Ключевым отличием задач генерации будем считать необходимость получения комплекса значений или параметров, в совокупности образующих конкретную сущность в информационной среде АИС, а не нахождение отдельных значений параметров или функций. Например, к задачам генерации в АИС отнесем формирование нового

информационного объекта – изображения определенного класса с набором требуемых свойств, используемого при решении определенных прикладных задач в АИС. Преобразование и изменение графических параметров (цвет, резкость, контраст) относится к классу задач обработки информации и не считается генерацией нового объекта. Аналогично, решение задачи восстановления поврежденной информации должно приводить к получению полностью восстановленного информационного объекта со всеми его свойствами и параметрами, а не только нахождению отдельного отсутствующего значения.

Рассмотрим основные задачи (*К1 – К4*) генерации данных в АИС.

Задача *К1*. Генерация сжатого представления о состоянии объекта. АИС, как и любая информационная система, обрабатывает и хранит огромное количество информации. Однако ее особенностью является возможность адаптации к изменяющимся факторам окружения, что позволяет обеспечить ее работоспособность. Если рассматривать в качестве фактора окружения структуру информационных объектов, с которыми осуществляется работа в АИС, то одной из актуальных задач является анализ этих объектов, их классификация, формирование некоторой обобщенной оценки состояния объектов. Такая оценка приводит к получению сжатой информации о состоянии объекта, что позволяет ускорить сравнительный анализ объектов, определить новые связи между ними, а также осуществить их кодирование и последующее декодирование на основе имеющегося сжатого представления [122].

Задача *К2*. Генерация подобных (псевдореальных) объектов. При разработке и тестировании АИС требуется проверить ее работоспособность на больших объемах различных данных, чтобы убедиться в надежности и стабильности ее работы на всем спектре различных значений из области определения входных параметров. Однако получить весь набор входных значений для информационных объектов не всегда представляется возможным. Поэтому одним из перспективных направлений в генерации данных является формирование подобных (псевдореальных) объектов, под которыми будем понимать объекты, соответствующие по области значений, структуре и свойствам классу исходного набора объектов. Применение таких объектов позволяет осуществить более полное тестирование АИС, выявить скрытые ошибки и неточности в ее функционировании на всей области определения переменных [123]. Данная задача может быть расширена до генерации новых подобных объектов с требуемыми свойствами. Свойства могут отражать тип объекта, его характеристики. На основе некоторой совокупности входных параметров (свойств) генерируется объ-

ект, обладающий этими свойствами [124]. Для проверки соответствия заданному классу или свойству можно использовать классификаторы или функции наград [125, 126].

Задача *К3*. Генерация отсутствующих данных в задачах обработки информации. АИС для обеспечения стабильной работы должны адаптироваться к ситуациям, когда поток поступающих данных может стать нестабильным, а набор информации – неполным. Таким образом, необходимо решать задачу восстановления поврежденных данных, значения которых не только принадлежат области определения, но и соответствуют всем значениям в текущей выборке. Возможность решения данной задачи с высокой эффективностью с помощью нейронных сетей подтверждается многочисленными исследованиями [127, 128].

Задача *К4*. Генерация предстоящих состояний объектов в задачах управления и обработки информации. В ряде задач управления и обработки информации в АИС требуется работать не только с текущими данными, но и анализировать предстоящие состояния объектов. Для этого на основе уже имеющихся данных об объектах или процессах требуется прогнозировать данные в некоторые ближайшие отрезки времени [129].

Проведенный анализ задач генерации информации в АИС приводит к необходимости разработки универсальных и автоматизированных методов генерации данных, что позволит повысить гибкость и надежность системы в нестандартных режимах работы, восстанавливать поврежденные данные, осуществлять анализ процессов и объектов, в том числе с возможностью прогнозирования.

1.5.5. Применение машинного обучения для управления и поддержки принятия решений

В области управления и поддержки принятия решений на основе методов машинного обучения можно выделить несколько основных направлений (задач).

Частичное нейросетевое управление. Под данной задачей будем понимать применение нейронных сетей или иных алгоритмов машинного обучения для выбора режима функционирования АИС на основе анализа данных о системе и объектах [135]. Нейронная сеть (регрессионная, рекуррентная или обычная многослойная) в данном случае выступает как классификатор, обученный на входном наборе состояний системы или объекта и заданных выходных значениях необходимых действий или операций. Данный подход достаточно легко реализуется,

однако плохо показывает себя на неучтенных в процессе обучения данных.

Полное нейросетевое управление. В отличие от предыдущей задачи, нейронная сеть анализирует состояние АИС или объектов и самостоятельно принимает решения о необходимых действиях или операциях. Реализуется сеть на основе машинного обучения с подкреплением [135]. Задается определенная функция награды, оценивающая качество работы системы в зависимости от ее текущего состояния и выбранных действий. Обученная нейронная сеть (чаще всего рекуррентная) стремится к выбору максимально «полезных» действий для системы. Такой метод управления позволяет адаптироваться к любым изменениям в системе, но подготовка соответствующей нейронной сети трудоемка и отнимает много вычислительных и временных ресурсов.

Прогнозирующее управление. Является частью вышерассмотренных задач и опирается на гипотезу, что процесс управления может быть реализован более качественно и эффективно, если в качестве исходных данных для управляющей нейронной сети использовать не текущие данные, а прогноз с некоторым временным шагом, что позволяет устранить возможные аппаратные и программные задержки в АИС, а также действовать с опережением на любые поступающие воздействия. Реализуется посредством регрессионной нейронной сети, обученной прогнозировать состояния объектов или всей системы.

Нейросетевая поддержка принятия решений. Данная задача включает использование нейронных сетей в составе систем поддержки принятия решений. В таком случае в качестве управляющего блока выступают не экспертные системы или логические модели, а методы машинного обучения, которые на анализе выполненных ранее действий осуществляют автоматическую (или автоматизированную) поддержку принятия решений.

Данное направление применения нейронных сетей активно развивается, так как решаемые ими задачи могут найти применение в самых разных сферах человеческой деятельности: промышленности, транспорте, организации бизнес-процессов, военной отрасли и др. Таким образом, задача управления может быть успешно решена с применением нейронных сетей, хотя и требует индивидуального подхода к каждой предметной области, тщательного анализа объектов и их окружения. Разработка новых алгоритмов и методов интеграции существующих АИС с нейросетевыми подходами к управлению позволит повысить адаптивность, надежность и гибкость подобного рода систем.

1.5.6. Подходы к оценке эффективности машинного обучения

Использование нейронных сетей или иных методов машинного обучения приводит к необходимости оценки полученного решения. Всегда возможна комплексная оценка программного модуля, в состав которого включена нейронная сеть, однако, если требуется сравнить точность или эффективность нескольких альтернативных вариантов нейронной сети, то необходимо воспользоваться специализированными подходами.

Прежде всего в ходе обучения и сравнения архитектур нейронных сетей используются широко известные функции оценки. Рассмотрим некоторые из них [136].

Доля правильных ответов (*Accuracy*) – относительное количество правильных прогнозов к общему объему контрольной выборки, выраженное в процентном соотношении. При использовании данной метрики необходимо соблюдать равномерность распределения различных классов данных.

Точность (*precision*) – процент правильно распознанных объектов заданного класса А к общему количеству объектов, который классификатор оценил как А. Данная метрика, в отличие от *Accuracy*, не зависит от соотношения классов.

Полнота (*recall*) – отношение правильно распознанных объектов класса А к общему объему объектов данного класса.

Оценка на основе функции потерь (ошибок). Данная функция рассчитывается как разница между фактическим и ожидаемым значением выходных нейронов. Существует несколько видов функций потерь.

Средняя абсолютная ошибка (*MAE*) – средняя разница между исходными и полученными нейронной сетью значениями, отражающая отклонение прогноза от нужного результата.

Средняя квадратичная ошибка (*MSE*) – усредненный квадрат разницы между полученными нейронной сетью значениями и истинными.

Кросс-энтропия (или логарифмическая функция потерь) – усредненное расхождение между двумя вероятностными распределениями.

Бинарная классификация определяет расхождение для двух классов. В данном случае каждая предсказанная вероятность сравнивается с фактическим значением класса (0 или 1), и вычисляется оценка, которая штрафует вероятность на основе расстояния от ожидаемого значения.

F1-мера – гармоническое среднее между точностью и полнотой, имеющее диапазон от 0 до 1. F1-мера стремится к нулю, если точность

или полнота стремится к нулю. Данная оценка отражает как точность, так и устойчивость сети.

Важным свойством нейронных сетей является сходимость – стремление сети в ходе обучения к минимизации ошибки. Если с ростом числа итераций ошибка находится на постоянном высоком уровне или последовательно многократно увеличивается и уменьшается (имеет поведение синусоиды), то нейронная сеть не сходится.

Следующее свойство, определяющее качество нейронной сети, – переобучение. Данное явление наблюдается при перенасыщении сети одинаковыми данными, на которых она показывает удовлетворительные результаты, но на любых других данных вне заданного набора – допускает ошибки.

Отдельно стоит разобрать вопрос оценки генеративно-состязательных нейронных сетей (GAN). Так как результат работы такой сети – сгенерированный набор данных, то его оценка первоначально проводилась экспертом. С другой стороны, такой подход занимает много времени, не может использоваться в процессе обучения автоматически, отличается субъективностью и полностью зависит от квалификации эксперта. Поэтому постоянно появляются все новые методы оценки качества GAN.

В работе [137] проведен сравнительный анализ 24 количественных и 5 качественных метрик для оценки GAN. Несмотря на такое количество различных подходов, на данный момент нет единого мнения относительно того, какая метрика лучше всего отражает качество моделей GAN и должна использоваться для объективного сравнения моделей. Однако среди всех этих оценок две получили наибольшее распространение – Inception Score и Fréchet Inception Distance.

Начальная оценка (Inception Score, IS), предложенная в работе [138], является одним из способов объективной оценки качества сгенерированных изображений. Следовательно, данная метрика также применима и для объективной и автоматической оценки качества GAN. Метрика IS, как показали многочисленные эксперименты, хорошо коррелирует с субъективными оценками экспертов.

IS основывается на использовании предварительно обученной модели нейронной сети глубокого обучения Inception v3 [139]. Данная модель используется для классификации сгенерированных изображений на основе случайного вектора z, после чего оценивается вероятность принадлежности изображений каждому из N классов (в случае Inception v3 $N = 1000$). Прогнозы суммируются и образуют итоговую оценку IS. Начальная оценка принимает значения от 1 до N. Высококачественная GAN должна генерировать изображения x, относящиеся

с высокой условной вероятностью $p(y|x)$ к определенному классу y. При соблюдении этого требования суммарная вероятность всех изображений будет иметь низкую энтропию. Также GAN должна обеспечивать разнообразие генерируемых изображений, а не только соответствие их заданному классу. Получаем два ключевых условия, обеспечивающих высокое качество GAN. Для их объединения разработчики метрики IS предлагают вычисление расстояния Кульбака–Лейблера [140].

Помимо IS большую популярность имеет метрика Fréchet Inception Distance (FID) [141], являющаяся дальнейшим развитием идеи объективной оценки GAN. Суть метрики FID заключается в сравнении признаков сгенерированных и реальных изображений. В FID также используется обученная сеть Inception v3 с исключением последнего слоя, что позволяет использовать для оценки не метки классов, а специфические признаки изображений, полученные из значений функций активации предпоследнего слоя модели. Для реальных и сгенерированных изображений рассчитывается многомерное нормальное распределение на основе среднего значения и ковариации активаций предпоследнего слоя. Расстояние между двумя распределениями определяется как FID. Низкое значение FID соответствует изображениям высокого качества и наоборот.

Необходимо отметить, что рассмотренные метрики оценки GAN способны оценивать качество только цветных изображений, так как именно на них обучена сеть Inception V3, используемая при расчете IS и FID. Кроме того, ряд исследователей, например в работе [142], отмечают, что IS недостаточно совершенна и обладает рядом существенных недостатков, что подтверждает актуальность задачи дальнейшего исследования и улучшения метрик, повышения их объективности, нахождения новых подходов к их использованию для решения задач оценки качества нейронных сетей.

Рассмотренные оценки качества нейронных сетей позволяют оценить эффективность их применения при решении задач анализа, обработки и генерации данных в АИС.

1.6. ВЫВОДЫ

В первой главе проведен анализ существующих методологий разработки информационных систем. Наличие большого количества используемых подходов, методов и шаблонов, однако, не позволяет полноценно решить задачу автоматизации разработки АИС. Каждая методология имеет свои ограничения и область применения.

Рассмотрены различные варианты архитектур информационных систем, что позволило выявить ключевые моменты и подходы, использование которых направлено на снижение сложности и автоматизацию процесса разработки АИС. Однако здесь также не существует универсальной архитектуры, что приводит к выводу о необходимости разработки новой архитектуры АИС, учитывающей сильные стороны рассмотренных подходов и ориентированной на автоматизацию процессов анализа, обработки и передачи данных.

Далее проведен анализ подходов к адаптивности в информационных системах, что позволило выявить достигнутые в этой области результаты и направления дальнейшей работы. Во-первых, необходимо осуществить формализацию модели пользователя в виде совокупности ключевых его характеристик. Во-вторых, требуется формализовать критерии адаптивности информационной системы, использовать их при оценке оптимальности АИС. Наконец, необходимо более тесно интегрировать оценку адаптивности системы в информационную систему с помощью систем поддержки принятия решений или интеллектуальных технологий.

Автоматизация процесса разработки АИС также содержит множество нерешенных вопросов, вызванных большой сложностью данного процесса. Структурно-параметрический синтез АИС является NP-сложной задачей, поэтому полностью автоматизировать ее решение с помощью существующих моделей и методологий в настоящий момент не представляется возможным. Разработка новой методологии, содержащей в своем составе автоматизированные методы анализа, обработки и передачи информации, позволит продвинуться в решении данной задачи.

В рамках исследования вопроса автоматизации процесса разработки АИС рассмотрены основные технологии машинного обучения, применяемые для автоматизации анализа, обработки, генерации информации, организации процессов управления в информационных системах. Анализ полученных результатов в этой области показал востребованность и эффективность данных технологий в различных прикладных задачах.

Однако применение нейронных сетей не является универсальным решением для всех задач. Во-первых, в ходе анализа, проведенного в данной главе, выявлены некоторые ограничения полученных программных решений. Существуют особые требования к исходным данным, без выполнения которых обеспечить сходимость нейронной сети невозможно. Нетривиальным вопросом является выбор архитектуры

нейронной сети и автоматических, универсальных и эффективных во всех случаях методов в настоящий момент не существует. Несмотря на существование ряда оценок качества алгоритмов машинного обучения, для некоторых задач достаточно сложно выбрать объективную метрику.

На основе проведенного анализа методологий разработки программного обеспечения информационных систем, архитектур и парадигм программирования, а также существующих подходов к решению задач адаптации компонентов АИС сформулирована научная проблема, имеющая важное хозяйственное значение: разработка и развитие теоретических основ и инструментальных программных средств на основе технологий машинного обучения и нейронных сетей для повышения эффективности процесса разработки адаптивных информационных систем: сокращения экономических затрат, уменьшения сложности программной реализации, повышения адаптивности, качества и производительности работы системы.

Глава 2
СИСТЕМНЫЙ АНАЛИЗ ПРОЦЕССОВ ОБРАБОТКИ ИНФОРМАЦИИ, МАТЕМАТИЧЕСКОГО МОДЕЛИРОВАНИЯ И ОПТИМИЗАЦИИ В АДАПТИВНЫХ ИНФОРМАЦИОННЫХ СИСТЕМАХ

В данной главе рассматриваются основные подходы к формализованному представлению информации в АИС на основе существующих методов, моделей и теорий.

2.1. АНАЛИЗ СУЩЕСТВУЮЩИХ ПОДХОДОВ К ОРГАНИЗАЦИИ И ФОРМАЛИЗАЦИИ ПРОЦЕССОВ РАБОТЫ С ИНФОРМАЦИЕЙ

При разработке информационных систем на первом этапе осуществляется анализ предметной области и исследование основных объектов, субъектов, протекающих процессов. Вербальная форма представления такой важной информации может привести к ошибкам и противоречиям между пользователями и разработчиками системы. Поэтому важным вопросом является применение формализованных подходов к анализу предметной области, описание и организация процессов работы с данными.

2.1.1. Процесс анализа информации

Анализ информации является важнейшим этапом при разработке любой информационной системы, решении задач по обработке данных, прогнозированию, реализации экспертных систем и организации поддержки принятия решений. В настоящее время существует огромное количество подходов и методик, используемых для анализа данных. Рассмотрим некоторые из них.

Методы анализа информации

Собранная информация, независимо от ее формата и способа организации, анализируется (это могут быть сами разработчики либо квалифицированные эксперты) для поиска неизвестных ранее закономерностей, обнаружения аномалий, несогласованности в данных и т.д. Действительно, при сборе информации частым случаем является наличие как поврежденных данных, так и «шума», не несущего смысловой нагрузки, а также образцов информации, не вписывающихся в заданные рамки области определения переменных. Аналитический подход на основе проверки данных квалифицированным специалистом позво-

ляет выявить и исключить такие некорректные данные из общей выборки.

Однако для повышения эффективности работы следует использовать существующие средства и методы, позволяющие упростить процесс анализа информации. Во-первых, это прежде всего методы системного анализа [45]:

– абстрагирование;
– классификация и кластеризация;
– декомпозиция;
– структурный анализ [46];
– функциональный анализ [47] и др.

В ходе анализа собранной информации исследователь может также воспользоваться различными методами статистического наблюдения – систематического сбора данных по интересующим характеристикам [48]:

– статистическое наблюдение;
– сводка;
– группировка;
– вычисление статистических величин;
– вариация;
– выборочный метод;
– корреляционный и регрессионный анализ;
– диаграмма распределения данных;
– ковариация и ковариационная матрица.

Способы формализованного представления данных

Для четкого понимания, какие объекты и субъекты входят в предметную область, каким образом они взаимодействуют друг с другом, могут использоваться различные способы организации информации [34]:

– файлы в различных форматах, как неструктурированных, так и структурированных (JSON, CSV и др.);
– базы данных и представление в табличной форме с возможностью отслеживания связей между сущностями;
– базы знаний;
– семантические сети;
– фреймы;
– продукционные и логические модели;
– абстрактные модели, построенные на основе теорий множеств, автоматов, графов.

Таким образом, собранная на основе наблюдения реального мира информация может быть представлена множеством способов в зависимости от специфики реализации информационной системы и тех процессов обработки данных, что в ней протекают.

2.1.2. Процесс обработки информации

Собранные данные после осуществления анализа позволяют выявить новые закономерности и условия протекания процессов, характеристики и свойства объектов, взаимосвязи между ними. Однако АИС направлены на решение более широкого спектра задач. И одним из них является процесс обработки информации – преобразование данных из одной формы в другую, выполненное в соответствии со строгими, формализованными правилами. Данный процесс условно можно разделить на несколько направлений [35]:

– получение новых данных на основе существующей информации с помощью математических вычислений или логических умозаключений;

– модификация формы представления информации с сохранением ее содержания или смысла;

– упорядочение, группировка, сортировка информации;

– поиск информации в массиве данных;

– изменение фрагментов информации, их замена в соответствии с заданными правилами.

Большинство процессов обработки данных в АИС являются производными от перечисленных либо объединяют их совместно в рамках конкретных операций, методов, функций. Таким образом, к процессам обработки данных относится создание, редактирование, перемещение и удаление документов и файлов всех возможных форматов, ввод и отображение информации, добавление, извлечение и изменение информации из баз данных, генерация информации, изменение параметров модулей и АИС в целом, и т.д.

Методы формализации процесса обработки информации

Определив перечень возможных операций по обработке данных, необходимо далее выбрать аппарат для их формализации. Это позволит сформировать четкое, однозначное представление для процесса обработки данных.

Так как в предметной области АИС процессы обработки информации связаны с использованием программного обеспечения, то основой реализации данных процессов является алгоритмическое и математическое обеспечение, реализованное с помощью заданных языков программирования. Таким образом, процесс обработки данных формализуется программным кодом, созданным разработчиком АИС в соответствии с поставленной задачей.

Однако реализовать программный код без формализованного представления операции невозможно с достаточной точностью. Поэтому этапу практической реализации предшествует формализация

процесса обработки данных с помощью существующих математических или алгоритмических подходов. Словесную (или вербальную) форму представления алгоритмов обработки данных мы рассматривать не будем, так как она не обеспечивает однозначности и воспроизводимости результатов.

Первым подходом является формализация процесса обработки информации в виде блок-схемы. Этот подход является простейшим и понятным как разработчику, так и заказчику, так как позволяет визуализировать и структурировать движение информации, ее преобразование, начальные и конечные данные.

Другим способом представления может быть граф-схема алгоритма обработки данных. В таком случае вершины соответствуют операциям, а дуги (ребра) – задают их порядок.

Следующим подходом на основе графического представления процесса обработки данных является UML-диаграммы. UML может быть использован для определения, визуализации, проектирования и документирования различных информационных систем, кроме того, существуют средства по генерации программного кода из UML. Альтернативой UML является визуальный алгоритмический язык программирования и моделирования ДРАКОН.

Формализация процессов обработки данных и бизнес-процессов организации в общем возможна с применением IDEF. К семейству стандартов IDEF относится 14 разновидностей диаграмм различной направленности. В рамках задачи формализации обработки данных особый интерес представляет IDEF0, позволяющая графически представить процессы с указанием входных и выходных данных, управляющих воздействий и условий.

Доступным и понятным инструментом для иллюстрации процессов и структуризации концепций являются диаграммы связей (Mind map). Несмотря на доступность представления информации, данный инструмент имеет ограниченную масштабируемость.

От графических способов формализации процесса обработки информации перейдем к символьным. Одним из способов упрощенного представления алгоритмов является псевдокод, позволяющий как разработчику, так и неподготовленному человеку без знания языков программирования понять последовательность операций над данными.

При наличии в команде разработчиков квалифицированных системных аналитиков возможно использование аппарата теории множеств для формализации объектов предметной области и процессов их взаимодействия. Тогда операции обработки данных будут представлены функциями или отображениями, различными действиями над мно-

жествами. Это позволит проанализировать процесс обработки еще до его практической реализации, изучить его свойства. Например, для отображений можно выявить его характер (сюръекция, инъекция, биекция и т.д.), что может использоваться в качестве ограничений при разработке программных методов и модулей.

Методы автоматизированной обработки информации

В современных информационных системах, в том числе АИС, вопрос автоматизации процессов обработки данных является одним из приоритетных, так как позволяет снизить влияние человеческого фактора на результат преобразования информации. Решается данная задача путем реализации новых методов и технологий, направленных на большую автоматизацию процессов обработки данных, в том числе с использованием интеллектуальных подходов [178].

Рассмотрим некоторые существующие подходы к автоматизированной обработке информации [36] в области:

1) текстовой и численной информации: классификация и кластеризация данных; прогнозирование; восстановление поврежденной или потерянной информации; генерация;

2) файлов и документов: автоматизированное создание документов; классификация файлов и документов по различным признакам; преобразование форматов файлов; извлечение информации;

3) графической информации: распознавание и классификация объектов; точечная обработка изображений; создание и генерация изображений по заданным правилам;

4) бизнес-процессов организаций: проведение расчетов и прогнозов планов развития организации, показателей эффективности работы; обработка информационных потоков, контроль выполнения распоряжений, заданий и операций; преобразование данных о состояниях, процессах и явлениях, деятельности предприятия.

Применение перечисленных подходов при разработке АИС позволит повысить производительность работы организации за счет автоматизации обработки данных.

Предварительная обработка данных в целях повышения их качества

В процессе сбора и анализа информации полученный набор данных может быть получен из разных мест, иметь различные форматы и структуру, более того, часть информации может иметь ошибки и неточности, быть искажена. Наиболее распространенными проблемами для данных является их неполнота (часть значений или атрибутов отсутствует), зашумленность (ошибочность некоторых значений), несогласованность (конфликты и противоречия между данными). Поэтому

качество данных является обязательным условием для реализации АИС. Для исключения искажения данных, их зашумленности и некорректности необходимо осуществлять процедуру предварительной обработки информации, которая включает следующие операции:

1. Очистка данных путем исключения некорректных данных.
2. Преобразование данных: нормализация, округление до целого.
3. Лемматизация и токенизация текстовых данных для перехода от символьных конструкций к численным обозначениям лексем.
4. Уплотнение данных: сокращение информации за счет выбора определенного подмножества.
5. Дискретизация данных: переход от непрерывных значений к категориальным, преобразование значений к разреженным матрицам.
6. Очистка текста: удаление лишних символов, пробелов либо добавление необходимых разделителей, упрощающих анализ текста.

Процесс анализа и обработки данных зачастую включает этап нормализации данных, т.е. их масштабирования в заданном диапазоне. Для нормализации могут использоваться различные подходы, например минимакс, Z-показатель, десятичное масштабирование и др.

Если объем анализируемых данных слишком велик, алгоритмы обработки данных могут работать медленно или некорректно. В таком случае требуется уменьшение размерности либо путем сокращения количества данных, либо с помощью отбрасывания менее полезных и важных атрибутов.

2.1.3. Процесс передачи и распределения информации

Под передачей информации понимается перемещение данных из одного места (физического носителя, логического раздела, модуля информационной системы) в другое в соответствии с выбранным протоколом. Под распределением информации понимается размещение данных и файлов на заданных позициях: по категориям, носителям, каталогам и т.д. Таким образом, эти два процесса неразрывно связаны.

Процессы передачи информации представляются как совокупность маршрутов движения данных – информационных потоков. Существует несколько подходов к его формализации и организации.

Первый подход основан на использовании аппарата теории системы массового обслуживания для формализации процессов движения информационных потоков [37]. Движение информации представляется в виде матриц, а переход от одного исполнителя к другому осуществляется в соответствии с заданным матрицей маршрутом. Об-

работка документов или данных может осуществляться в порядке очереди либо в соответствии со срочностью или приоритетом операций.

Вторым вариантом является применение алгоритма Клейнрока [38] и организация переадресации информационных потоков по наиболее безопасным маршрутам. Полученное данным методом решение не будет оптимальным по времени выполнения, но будет отвечать ограничениям на минимальное быстродействие и надежность доставки.

Процесс передачи данных напрямую зависит от нагрузки на информационную систему и сеть [39, 40]. Разработанные алгоритмы по оценке загруженности узлов информационных систем можно использовать для решения сопутствующих задач по оценке состояния пользователей и количества закрепленных за ним задач или операций, нагрузки на модули и компоненты системы. На основе этих данных можно принимать решение о перенаправлении информационных потоков.

Другим важным аспектом процесса передачи информации яляется его эффективность [41]. Под этим можно понимать время нахождения компонента (узла) системы в состояниях, когда он обрабатывает либо ожидает информацию, т.е. время его загруженности. Сокращение этого времени для каждого компонента повышает эффективность работы всей системы.

В настоящее время процесс передачи регламентируется написанием протоколов передачи данных – набором соглашений, которые определяют обмен данными между различными программами. Разработчики при реализации информационных систем выбирают подходящий для решения конкретной задачи протокол. Так, например, широкое распространение получили протоколы семейства TCP/IP: HTTP, TCP, UDP, FTP, POP3 и др.

Организация процесса передачи информации между модулями системы также зависит от архитектуры приложения. Сейчас можно выделить два основных направления: монолитные и сервис-ориентированные (к которым относятся микросервисные) архитектуры [42].

Монолитная структура проще в реализации для небольших коллективов разработчиков, связи между ее элементами более простые и тесные. Однако ее сложнее модифицировать и масштабировать после ввода в эксплуатацию, а если приложение аварийно завершает работу, то все компоненты становятся недоступны для пользователей.

Микросервисная архитектура требует большего количества разработчиков, имеет более сложную структуру связей между отдельными модулями (сервисами), однако каждый из них является обособленным, независимым приложением, разработка которого проще, чем

всей системы в целом. Такая декомпозиция и разделение функциональности повышает надежность функционирования системы, позволяет легко модернизировать отдельные ее элементы, не нарушая работу остальных. Каждый сервис может разрабатываться независимо друг от друга. Процесс передачи данных в микросервисной архитектуре осуществляется посредством таких протоколов, как HTTPs (HTTP) или AMQP.

На базе микросервисной архитектуры реализуются такие стили передачи информации, как REST для организации межмодульного взаимодействия по стандартам HTTP, URL, JSON и XML. Тем не менее даже приложения, построенные на основе REST, могут не иметь единой формы передачи данных, так как нет жестких правил по организации структуры данных или выбору типа данных. Этот вопрос решается применением протокола SOAP, структура передаваемых данных в котором задается спецификацией. К сожалению, использование SOAP для передачи сообщений увеличивает их объем и снижает скорость обработки.

Вопрос автоматизации процесса передачи информации остается нерешенным и требует пристального внимания разработчика в области вопросов согласования передаваемых типов данных и их структуры, протоколов связи, оценки загруженности. Аналогично ситуация обстоит и с распределением данных, где многие операции требуют непосредственного участия эксперта (модератора). Поэтому актуальной задачей является разработка новых методов и подходов, объединяющих ключевые свойства процесса переадресации (эффективность, загруженность, оценка компетенций исполнителя) и направленных на автоматизацию данного процесса.

2.2. АНАЛИЗ ПОДХОДОВ ПО ФОРМАЛИЗАЦИИ И МОДЕЛИРОВАНИЮ ИНФОРМАЦИОННЫХ СИСТЕМ

После анализа и формализации процессов работы с информацией в предметной области необходимо осуществить формализацию структуры информационной системы. Наиболее простым способом анализа структуры системы является ее графическое изображение в виде структурной схемы. Такие схемы показывают вложенность компонентов, направление связей между ними и позволяет в дальнейшем осуществить декомпозицию процесса разработки. Однако, структурные схемы несут описательный характер и не дают возможности осуществить моделирование или оптимизацию системы. Поэтому далее мы рассмотрим некоторые подходы по математическому моделированию структуры информационных систем.

2.2.1. Теоретико-множественные модели

Модели данного типа формализуют объекты как элементы множеств, а их взаимодействие – как операции (отображения) над множествами или отдельными элементами. Аппарат теории множеств развит, имеет глубокую теоретическую базу, на основе которой возможно формализовать объекты, процессы и явления как реального мира, так и движение информационных потоков.

Рассмотрим обобщенный подход к математическому моделированию АИС с применением теории множеств. Пусть множество X задает информационные объекты, множество Y-субъекты (пользователей), множество Z-компоненты (модули) системы. Тогда можно формализовать процессы обработки данных через множество функций F, таких, что:

$$\begin{gathered} F : X,\ Y \to X, \\ Z \to F. \end{gathered} \tag{2.1}$$

Таким образом, функции АИС распределены между модулями, а каждая функция приводит к изменению информационных объектов с помощью пользователя.

Структуру S информационных модулей возможно задавать кортежем

$$S = (m_1, m_2, ..., m_n). \tag{2.2}$$

Оценка параметров P АИС формализуется в виде некоторых функций R (критериев), которые в ходе оптимизации АИС стремятся к минимуму или максимуму:

$$\begin{gathered} R : P \to (r_1,\ r_2,\ r_3,\ ...), \\ r_1 \to \max,\ r_2 \to \min,\ r_3 \to \min,\ ... . \end{gathered} \tag{2.3}$$

Применение теоретико-множественного аппарата для формализации АИС позволяет также до разработки самой системы сформулировать структуры базы данных (перечень атрибутов a_k), каталогов и подгрупп (X_t), электронного хранилища за счет множеств и подмножеств:

$$\begin{gathered} x_{ij} \in X_t,\ x_{ij} \to (a_1,\ a_2,\ a_3,\ ...,\ a_k), \\ X_t \subseteq X. \end{gathered} \tag{2.4}$$

Применение подходов и методов из теории множеств доказало свою эффективность при решении задач из различных классов и сфер деятельности.

2.2.2. Теоретико-графовые модели

Следующий тип моделей включает как формализацию объектов и их структуры с помощью теории множеств, так и графическую визуализацию протекающих в предметной области процессов в виде графов. Такое сочетание позволяет достаточно подробно представить структуру данных, их преобразования через множества и отображения, отразить жизненный цикл и характер выполнения операций через графы различных структур.

Рассмотрим общий подход к формализации структуры информационных систем с помощью теоретико-графовых моделей.

Если граф $G(V, E)$ используется для описания жизненных циклов информационных объектов, то вершинам V будут соответствовать состояния объекта, а ребрам E – действия (операции), приводящие к переходу из одного состояния в другое.

Если граф $G(V, E)$ используется для формализации структуры информационной системы, то вершинам V будут соответствовать компоненты системы, а ребрам E – взаимосвязи между ними, по которым осуществляется обмен информацией.

Причем связи между вершинами могут быть ориентированы для отображения направления перехода из одного состояния в другое или отражения движения информации между компонентами структуры системы. Примеры графов представлены на рис. 2.1.

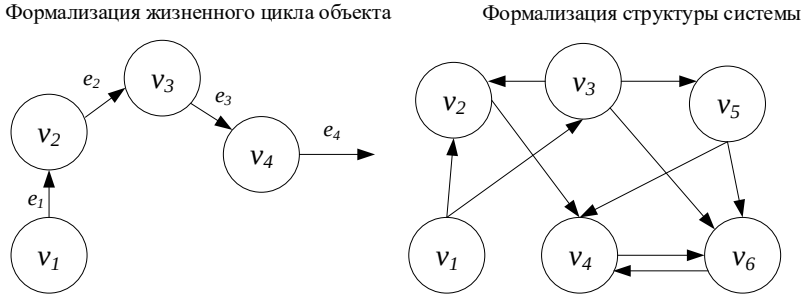


Рис. 2.1. Примеры графовых моделей

Помимо наглядной визуализации в теоретико-графовых моделях присутствует формализация объектов с применением теории множеств. Это означает, что некоторая вершина $v_i \in V$ не только расположена на графе и имеет связи $e_k \in E$ с другими вершинами $v_j \in V$, но и может быть описана как элемент некоторого множества со своей сложной структурой:

$$v_i = (a,\ b,\ c,\ ...), \tag{2.5}$$

а ребро может быть представлено как отображение:

$$e_k : v_i \to v_j. \tag{2.6}$$

Теоретико-графовые модели, несомненно, представляют большой интерес при решении задач формализации структуры информационных систем, процессов обработки данных, исследовании жизненных циклов объектов и систем.

2.2.3. Автоматные модели

Автоматные модели широко применяются для описания процессов функционирования различных процессов. Автоматы разделяют на абстрактные и структурные.

Абстрактные автоматы принято описывать как некоторое устройство, которое распознает определенные последовательности из A^*, где A – конечный алфавит, а A^* – все возможные комбинации из символов алфавита. Различные автоматы распознают различные последовательности из A^*. Подмножество символов из множества A, принимаемое автоматом M, называется языком над алфавитом A. Для заданного конечного алфавита автомат состоит из множества состояний S и множества функций $F : A \times S \to S$, называемых функциями переходов. Множество S содержит начальное состояние s_0 и одно или несколько завершающих (решающих) состояний T. Таким образом, абстрактный автомат однозначно задается следующей пятеркой:

$$(A,\ S,\ s_0,\ T,\ F). \tag{2.7}$$

Структурные модели автоматов выступают в роли управляющих устройств в системах управления и рассматриваются не как некоторый распознаватель языка, а как устройство управления. Для структурных автоматов важными компонентами являются множества входных воздействий X, множество состояний Y и множество выходных воздействий Z.

Рассмотрим пример использования автоматной модели в предметной области обработки документов [51]. Запишем ее в следующем виде:

$$AM = (Q, \sum, \delta, q_0, F), \tag{2.8}$$

где Q – конечное множество состояний документов; $\sum$ – конечное множество входных символов, образующих входной алфавит и представляющих собой входные данные для системы документооборота; δ – функция переходов, аргументами которой являются текущее состояние и входной символ, а значением – новое состояние; q_0 – начальное состояние (или множество начальных состояний документов) из множества Q; F – множество завершающих или допускающих состояний из множества Q. Множество состояний автоматов Q получается из множества состояний документов. Состояния задаются последовательно, завершающим является то, что имеет входящие связи, но не одной исходящей.

Представление информационной системы в виде автоматной модели позволяет отобразить сложные процессы в виде отдельных автоматов, каждый из которых формализует порядок смены состояний информационных объектов. Применение графов позволяет использовать развитый аппарат теории графов при описании связности автоматов. Достоинством данного типа моделей является ее полнота, универсальность, возможность использования большого набора инструментов теорий графов и автоматов.

2.2.4. Модели на основе интеллектуальных агентов

Данный тип моделей основан на применении при разработке информационных или иных систем интеллектуальных агентов – сущностей, получающих информацию о состоянии процессов через систему сенсоров и осуществляющих контроль над ними через систему управляющих механизмов.

Основными свойствами агентов являются: автономность (способность выполнять действия без участия других модулей или пользователя), коммуникация (взаимодействие с другими агентами и пользователем); независимость (самостоятельное принятие решений).

При построении модели информационных систем агенты обычно объединяются в некоторое множество (мультиагентные системы), где каждый элемент отвечает за решение определенной задачи. Это позво-

ляет декомпозировать процесс проектирования, изолировать агенты друг от друга для обеспечения их совместной независимой работы.

Формализация информационных систем с помощью интеллектуальных агентов связана с их математическим и графическим представлением. Для первого возможно использование различных математических аппаратов, например множеств и автоматов для описания структуры и функционирования агента. Графические агенты могут быть представлены графами, блок-схемами, UML или иными способами.

Рассмотрим пример формализации модели на основе агента [52]. На первом этапе необходимо описать структуру системы, включающую основные объекты предметной области C_i, организационную структуру агентов O_A, внутреннюю структуру агентов S_A:

$$A = \{C_1, C_2, ..., C_2, O_A, S_A\}. \quad (2.9)$$

Организационная структура включает следующие компоненты, которые могут быть формализованы множествами и их элементами:

$$O_A = \{T_A, R_A, AB_A, ACT_A, ST_A, L_A, ST_A, Y_A, FT_A\}, \quad (2.10)$$

где T_A – множество целей агента, достижение которых нужно для решения поставленной перед ним задачи; R_A – множество ролей агента (определяет набор его функций); AB_A – множество способностей агента; ACT_A – множество доступных агенту действий; ST_A – множество стратегий агента для достижения целей (последовательности действий); L_A – множество языков (на котором агенты формируют запросы и ответы, взаимодействуют между собой); ST_A – множество состояний агента; Y_A – множество правил перехода между состояниями; $FT_A = ST_A \times ACT_A \times SL_A$ – функции перехода между состояниями при выполнении действий с соблюдением условий.

Применение агентов в ряде задач показывает свою эффективность, однако и сопряжено с рядом трудностей, так как даже на этапе формализации можно заметить, что агенты являются дополнительным слоем абстракции над известными математическими и алгоритмическими подходами. Поэтому их использование может дополнительно усложнить формализацию и реализацию информационной системы. Тем не менее те полезные свойства агентов, что были рассмотрены, могут быть учтены при реализации как архитектуры АИС, так и отдельных ее компонентов.

2.3. АНАЛИЗ ПОДХОДОВ К ОПТИМИЗАЦИИ И ОЦЕНКЕ АДАПТИВНЫХ ИНФОРМАЦИОННЫХ СИСТЕМ

При реализации адаптивных информационных систем недостаточно формализовать их структуру и описать протекающие в предметной области процессы обработки и движения информации. Основополагающим вопросом при проектировании любых информационных систем является оценка их эффективности, определение, насколько полученное решение является оптимальным с точки зрения выбранных критериев.

Так как к вопросу оптимизации адаптивных информационных систем нельзя подойти однозначно, рассмотрим основные направления и критерии, по которым возможно оценить разрабатываемые системы.

2.3.1. Оценка экономической эффективности адаптивных информационных систем

Одним из основных направлений оптимизации любых систем является снижение экономических затрат на их проектирование и эксплуатацию. Можно выделить следующие составляющие [53, 54]:

– стоимость разработки программного обеспечения;
– стоимость аппаратного обеспечения;
– затраты на модернизацию и замену клиентского оборудования пользователей АИС;
– стоимость проведения НИР для разработки АИС;
– лицензионные отчисления на программное обеспечение;
– затраты на эксплуатацию АИС и амортизационные отчисления.

Важной составляющей экономической эффективности АИС является положительный или отрицательный эффект от использования АИС, который достигается после ее внедрения и в ходе эксплуатации [55]. Таким образом, даже максимально эффективная с точки зрения первоначальных затрат АИС может привести к значительным затратам в случае отрицательного эффекта от использования, например ввиду ошибок повреждения информации или иных рисков и потерь.

При расчете экономической эффективности АИС необходимо учитывать ряд субъективных факторов, оказывающих значительное влияние на итоговую стоимость разработки и внедрения [56]. К ним относятся: неправильный выбор бюджета разработки АИС; слишком жесткие рамки бюджета; выбор самого дешевого решения.

В работе рассматривается экономическая эффективность внедрения информационной системы с точки зрения совокупной стоимости

владения (TCO) [57]. В основе модели расчета TCO лежит разделение расходов на прямые и косвенные расходы. К прямым расходам относится стоимость сервера, обслуживания, сетевого оборудования, фиксированные эксплуатационные расходы. Косвенные затраты включают затраты на конвертацию, интеграцию, снижение риска, эксплуатационные расходы. При вычислении TCO рассматриваются затраты за весь период эксплуатации системы.

К показателям экономической эффективности автор работы [58] относит следующие показатели: чистая текущая стоимость, индекс рентабельности, внутренняя норма возврата. Мониторинг данных показателей позволяет оценивать как положительные эффекты от внедрения АИС, так и выявить проблемы, возникающие при ее эксплуатации.

Проведенный анализ подходов к определению экономической эффективности позволяет сделать следующие выводы. Данный критерий достаточно тяжело поддается точному вычислению на этапе планирования и разработки системы, существует ряд субъективных факторов, которые могут изменить общие затраты на реализацию информационной системы. С другой стороны, существует ряд подходов, которые можно использовать как для оценки существующей системы и ее эффективности в процессе эксплуатации, так и для расчета затрат на процессы обработки, хранения и движения информации. Их применение при решении задачи структурно-параметрического синтеза АИС позволит выявить оптимальные с экономической точки решения и не учитывать изначально нерентабельные варианты.

2.3.2. Оценка качества адаптивных информационных систем

Под качеством работы АИС можно понимать результат оценки разработанной системы экспертной группой по ряду показателей. В качестве перечня таких показателей можно использовать различные стандарты: ГОСТ 28195–99, ГОСТ Р ИСО/МЭК 25010–2015, ISO / IEC 25040:2011, ISO / IEC 33001 (и другие стандарты семейства ISO / IEC 330xx) и др. При использовании некоторой системы экспертной оценки программного обеспечения качество АИС может определяться следующим образом:

$$Q = \sum_{i=1}^{N} (v_i q_i), \qquad (2.11)$$

где v_i – весовой коэффициент i-й метрики; q_i – оценка экспертов по i-й метрике.

На оценку качества АИС оказывает влияние множество факторов: ее стабильность и быстродействие, надежность, визуальные параметры, удобство интерфейса и т.д. Не всегда самое дорогое программное обеспечение является наилучшим и удобным для конечного пользователя. Поэтому при разработке АИС важно учитывать предпочтения сотрудников, выбирать средства разработки, формы интерфейса и инструменты взаимодействия с АИС, обеспечивающие наибольшее удобство при работе, интуитивность интерфейса, меньший срок адаптации и переобучения сотрудников. Таким образом, качество зависит и от адаптивности системы. Определение адаптивности информационных систем было рассмотрено в первой главе на примере критериев адаптивности [15] и эргономических критериев [11].

Если информационная система используется в образовательном процессе (например, электронное обучение в школе, университете, системы тестирования, тренажерные комплексы и т.д.), то показателем ее качества может быть уровень сформированности компетенций обучаемого и скорость освоения знаний. Удобство, доступность системы в таком случае оказывают непосредственное влияние на процесс освоения знаний [43].

Рассмотрим еще несколько подходов к определению качества АИС. Авторы работы [59] выделяют следующие направления, по которым можно оценить информационную систему: целостная оценка всей системы; оценка отдельных компонентов; оценка программного обеспечения, процессорный подход. Предпочтение отдается последнему методу, так как остальные, по мнению авторов, не дают целостного и подробного описания влияния компонентов системы друг на друга, не рассматривают их жизненные циклы.

Выбор критериев для оценки системы может зависеть от множества параметров: требований заказчика, ожидаемого набора функций, специфики системы и ее фрагментов и т.д. Для каждого критерия необходимо определить его меру, шкалу, способ изменения, эталонные и граничные значения. Критерии оценки качества информационных систем можно разделить на качественные и количественные [60, 61].

К качественным критериям относятся субъективные оценки, которые невозможно четко сформулировать в виде закономерности или формулы, однако они позволяют получить некоторые сведения о качествах и свойствах системы, хотя и не всегда точные и достоверные.

Количественные критерии представляются как непрерывные или дискретные функции, заданные на некоторых интервалах и позволяющие объективно сравнить два объекта между собой, проверить их при-

надлежность некоторой области. Так как количественные критерии выражаются численными характеристиками, они более четко и понятно отражают достоинства и недостатки информационных систем по некоторой шкале. С другой стороны, формулы, используемые для определения этих критериев, могут быть недостаточно точны и правильны, не учитывать все возможные факторы, что может негативно повлиять на общую оценку [62]. Среди критериев качества информационных систем авторами выделяются три основных, которые разделяются на подкритерии и отдельные метрики.

Надежность. Объединяет несколько критериев, характеризующих стабильность работы информационной системы.

Достоверность. Характеризует систему с точки зрения возникновения в ней ошибок.

Эффективность. Определяет общие качественные характеристики информационной системы.

Данный перечень критериев и метрик позволяет всесторонне оценить качество функционирования информационных систем. Подходы к расчету рассмотренных критериев, их подробное описание и применение представлены в работе В. В. Липаева [61, 62].

Перечень возможных показателей качества информационных систем представлен также в работе [63]. Авторы относят к ним следующие критерии: целостность, сложность, структурированность, адаптивность, лабильность, интегрируемость, делимость, валидность. К критериям качества также могут быть отнесены показатели ее экономической эффективности: затраты на разработку и эксплуатацию системы, социально-экономический эффект от эксплуатации системы.

В статье [64] рассматривается задача оценки качества информационных систем с распределенной архитектурой. Предлагаемый авторами метод прогнозирования оценки качества системы отличается использованием квалиметрических измерений, где качество рассматривается как иерархическая совокупность характеристик. Оценка системы формируется на основе экспертной оценки каждой характеристики системы.

Качество системы может быть определено по ее эффективности в соответствии с четырьмя критериями: наличие основных элементов, достаточность, доступность и востребованность информационного обеспечения [65].

Многокритериальная оценка качества информационных систем рассматривается в работе [66]. Предлагается модель оценки характеристик системы в виде двухуровневой модели принятия решений в условиях неопределенности. Модель включает множество решений (выбо-

ров ответственного лица касательно архитектуры информационной системы), множество состояний среды (положительно или отрицательно влияющих на значения функции полезности) и множество характеристик (описывается функциями полезности, отражающими полезность, эффективность, успех выполнения операции). Ответственное лицо решает задачу оптимизации на двух уровнях: на верхнем ищется максимум функции качества (зависящий от множества критериев качества) среди всех вариантов системы, на нижнем – для каждого решения определяются значения критериев качества.

Таким образом, проведенный анализ оценки качества АИС позволяет сделать вывод о многокритериальности данного показателя и большей доле субъективности при его определении (за счет привлечения экспертов или опроса пользователей). Разработка систем поддержки принятия решений на основе интеллектуальных технологий и применение количественных оценок на основе точных расчетов позволят повысить объективность данной оценки.

2.3.3. Оценка сложности разработки адаптивных информационных систем

Помимо оценки экономической эффективности и качества АИС, важным моментом при разработке информационных систем является анализ сложности предстоящей разработки. Это обусловлено тем, что сложность разрабатываемого программного обеспечения напрямую влияет на стоимость и время его реализации, а также на вероятность возникновения ошибок. Более сложную систему также затруднительно поддерживать и модифицировать.

Существует несколько общепринятых подходов к оценке сложности разработки программного обеспечения. Одним из простейших методов анализа сложности программного кода является оценка его объема. Рассмотрим основные методы данной категории.

Метод LOC/KLOC измеряет сложность программы в количестве строк кода (в KLOC – в тысячах строк) [67]. Объем программы может влиять на сложность ее реализации и дальнейшей поддержки, однако современные языки программирования и мощные библиотеки значительно повысили значимость отдельной строки программного кода.

ABC-метрика анализирует три основных компонента кода: количество присваиваний значений переменным (Assignment), вызовов функций (Branch) и логических проверок (Condition). Для расчета значения метрики используется квадратный корень из суммы квадратов трех полученных значений. Недостатком данного подхода является

возможность получения нулевого значения для непустых программных модулей [67, 68].

Метрика Чепина заключается в оценке информационной прочности программного модуля с помощью анализа характера использования переменных из списка ввода-вывода [67, 68].

Цикломатическая сложность CC является метрикой, основанной на расчете количества переходов, циклов, генераторов, обработчиков исключений и логических операторов [69]. Цикломатическая сложность может быть определена через граф потока программы, где узлам соответствуют неделимые блоки команд, а ребрам – переходы между этими блоками, если они могут быть выполнены друг за другом. Таким образом, чем сильнее разветвлен граф, тем сложнее программа.

Сложность программного кода Джилба J определяется отношением количества условных операторов c к общему количеству операторов.

Таким образом, чем больше в программе ветвлений, тем более она сложна по метрике Джилба [70]. И хотя в ряде случаев эта оценка не является объективной, большое количество условных операторов вынуждает программиста проверять правильность всех используемых в них условий и границы блоков, что оказывает влияние на сложность программы.

Перечисленные методы обладают рядом недостатков, так как опираются на подсчет блоков или операторов программного кода. Однако привлечение квалифицированных программистов и внедрение сторонних библиотек позволяет значительно снизить объем кода, что не влияет на общую сложность решаемой задачи. Кроме того, количество кода сильно зависит от тех языков программирования, что используются при разработке. Часть кода в процессе разработки может быть отброшена, закомментирована, использоваться только в тестовых или промежуточных версиях, что усложняет процесс оценки.

Однако данные метрики достаточно легко рассчитываются, позволяют оценить масштаб предстоящей работы, сравнить два проекта (при их реализации сходными командами разработчиков). Различные варианты решений и функций также могут быть оценены с точки зрения этих методов.

Далее рассмотрим метод функциональных точек. Он используется для оценки времени разработки, производительности труда и объема работ на ранних этапах проекта [71]. Точность метрики зависит от детализации требований к разрабатываемому программному обеспечению. Достоинством данного метода является его независимость от языка разработки и возможность использования на ранних этапах про-

ектирования. К недостаткам же относятся сложность использования и тот факт, что метод основан на экспертных оценках сложности и зависит от квалификации экспертов в данной предметной области.

Инженерный метод оценки трудоемкости проекта PERT опирается на три оценки: M – наиболее вероятная оценка трудозатрат; O – минимально возможные трудозатраты на реализацию пакета работ (без учета возможных рисков); P – пессимистическая оценка трудозатрат (с учетом всех возможных рисков). Для расчета средней трудоемкости реализации пакета работ используется следующая формула:

$$E = (P + 4M + O)/6. \tag{2.12}$$

Метод PERT может быть расширен и доработан, например, в работе [72] задается ковариационная матрица продолжительностей этапов разработки, для каждого из которых вводится система штрафных выплат для регулирования сроков выполнения.

Одной из самых известных моделей оценки стоимости и времени разработки проекта является модель COCOMO [73, 74], которая предполагает использование различных метрик для прогнозирования стоимости и времени разработки и позволяет оценить трудоемкость создания программного обеспечения в человеко-месяцах, общую длительность программного проекта, а также численность коллектива, необходимую для успешного завершения проекта в заданные сроки.

Базовый уровень COCOMO рассчитывает трудоемкость и стоимость разработки как функцию от размера программы. Размер выражается в оценочных тысячах строк кода (KLOC – kilo lines of code). Основные метрики рассчитываются по следующим формулам:

$$\text{Трудоемкость}\left(\text{человеко-месяцев}\right) = a_b (KLOC)^{b_b}, \tag{2.13}$$

$$\text{Срок разработки}\left(\text{месяцев}\right) = c_b (\text{Трудоемкость})^{d_b}, \tag{2.14}$$

$$\text{Число разработчиков}\left(\text{человек}\right) = \frac{\text{Трудоемкость}}{\text{Срок разработки}}. \tag{2.15}$$

Вспомогательные коэффициенты представлены в справочных таблицах [75]. Недостатком базового уровня COCOMO является то, что он не принимает во внимание опыт персонала, а также те программные и аппаратные средства, с помощью которых осуществляется разработка.

Средний уровень учитывает множество вспомогательных факторов, включающих субъективные оценки характеристик продукта, про-

екта, персонала и аппаратного обеспечения, которые используются как коэффициенты при расчете формулы трудоемкости:

$$\text{Трудоемкость}\left(\text{человеко-месяцев}\right) = a_i\,(KLOC)^{b_b} \cdot \prod \text{ФТ}, \quad (2.16)$$

где ФТ – факторы трудоемкости, значения которых принимаются в диапазоне от 0,9 до 1,4 и заданы в справочных таблицах [75].

Детальный уровень определяет трудоемкость на каждом этапе разработки, что является излишним, если оценка производится только один раз для определения тех программных средств, что планируется использовать при разработке АИС. Метод COCOMO отличается простотой, субъективностью, достаточной детализацией, учитывает квалификацию разработчиков и особенности тех средств, которые они используют, поэтому очень часто применяется на практике при оценке сложности проектов.

В качестве критерия оценки сложности разработки АИС может использоваться совокупная вычислительная сложность *O*, вычисляемая путем оценки сложности используемых алгоритмов, процедур и функций по О-нотации [76]. Сложность может зависеть от объема предполагаемых работ, например количества входных данных либо от объема занимаемой памяти.

Еще одним способом оценки сложности программного кода являются метрики Холстеда. Благодаря данным метрикам возможно определение норм первоначальных ошибок, количественная оценка языков программирования и эффекта модульности, обоснование различий между программами, написанными специалистами разного уровня [77].

Метрики Холстеда анализируют два основных компонента программ: операторы и операнды (переменные и константы). На первом этапе определяется количество различных операций *n1* и операндов *n2*, общее количество операций *N1* и операндов *N2*. Метрики Холстеда включают определение словаря ($n = n1 + n2$), длины ($N = N1 + N2$), объема ($V = N\log2(n)$), а также сложности программы, трудозатрат разработчика, времени разработки и количества возможных ошибок.

С метриками Холстеда связан также такой подход, как индекс поддерживаемости кода (Maintainability Index), отражающий, насколько сложно будет поддерживать или редактировать фрагмент программы [78]. Данная метрика позволяет оценить как качество модуля, так и всей программы на основе комбинации значений оценок критерия Холстеда (*HAL*), цикломатической сложности (*CC*) и количества строк в программе (*LOC*). В работе [79] предложена следующая формула для расчета индекса:

$$MI = \max\left(0, \frac{171 - 5.2\ln(HAL) - 0.23CC - 16.2\ln(LOC)}{1.71}\right). \quad (2.17)$$

Рассмотренные метрики активно используются при оценке сложности программного обеспечения различных информационных систем. Помимо перечисленных выше подходов, существует большое количество различных методов, например Мартина, Кафура, Чидамбера и Кемерера, Лоренца и Кидда, Абреу, Пивоварского, Харрисона, Мейджела и др. [80]. Кроме того, постоянно появляются все более точные и сложные метрики.

Например, в работе [81] автор приходит к выводу о необходимости комбинированной оценки программного кода сразу по нескольким метрикам и использует комплексный критерий, объединяющий метрики Холстеда, меры Чепина и Шнадевида, цикломатическую сложность и ряд других метрик, отражающих количество внутренних связей. На их основе формируется набор метрик, отражающих сложность кода, его связность, структурированность, а также объемные характеристики.

О необходимости множественной оценки программного обеспечения упоминается и в работе [82], где рассмотрена задача оценки трудоемкости разработки мобильных систем. В качестве основных метрик автор использует LOC, COCOMOII и метод функциональных точек. Такой подход позволяет получить более точную итоговую оценку сравниваемых систем. Хотя автор подчеркивает, что между рассмотренными метриками присутствует некоторая зависимость и, зная значение одной из них, можно спрогнозировать значения остальных.

Таким образом, рассмотрев оценку сложности программного обеспечения, получены следующие выводы: существующие метрики позволяют с некоторой точностью получить оценку программного кода, однако для большей объективности результата необходимо использовать их комбинацию или комплексный критерий, объединяющий в себе несколько метрик.

2.3.4. Оценка производительности адаптивных информационных систем

Производительность АИС является важной составляющей ее работы и оказывает влияние как на экономическую эффективность, так и на оценку качества. Высокая производительность системы всегда сопряжена либо с мощным оборудованием, на котором она функционирует, либо с высочайшей степенью оптимизации программного обеспечения, что ведет к большим сроку и стоимости разработки, необходимости привлечения высококвалифицированных сотрудников.

Однако низкая производительность способна нивелировать весь положительный эффект от внедрения АИС, обеспечить негативный пользовательских опыт, затормозить процессы обработки и передачи данных, что понизит общую эффективность работы всей организации и приведет только к большим экономическим потерям.

Рассматривать оценку производительности АИС можно по двум направлениям: аппаратному и программному.

В первом случае анализируется вычислительная мощность аппаратных средств, используемых для работы АИС. Необходимо четко понимать, чем один набор компонентов сервера, терминала или иного оборудования отличается от другого. В качестве метрик рекомендуется использовать достаточно объективные тесты производительности, позволяющие сравнивать характеристики оборудования по различным критериям [83].

При оценке производительности и надежности оборудования важно помнить, что избыточная мощность оборудования приведет лишь к повышению экономических затрат, и выбор компонентов должен быть обоснован, с одной стороны, требованиями к необходимой производительности, а с другой – желанием максимально снизить затраты на оборудование [53]. Оцениваться должен каждый ключевой компонент аппаратного обеспечения, так как использование даже одного некачественного, малопроизводительного комплектующего может привести к значительным потерям в общем быстродействии [84].

Процесс выбора аппаратного обеспечения серверов можно автоматизировать, используя качественные оценки компонентов, их текущую стоимость и сочетаемость [85]. Как упоминалось выше, для оценки следует использовать независимые результаты тестов производительности. Рассмотрим некоторые из них.

Для тестирования быстродействия центрального процессора используется множество тестов, которые условно можно разделить на две категории, первая из которых ориентирована на оценку производительности при отображении графики, а вторая – на вычислительную мощность. К популярным тестам относятся: 7-Zip CPU Benchmark, BAPCo SYSmark, Cinebench Rendering Benchmark, FutureMark 3DMark, Intel Linpack x64 и др. [86 – 89]. Часть тестов ориентирована на получение конкретного значения вычислительной мощности, выраженного в количестве операций в минуту (flops).

Так как оценивать вычислительную мощность процессора в относительных единицах какого-либо теста не имеет смысла, то имеет смысл обратиться к тестам, оценивающим быстродействие процессора непосредственно при выполнении конкретных задач. Примером

такого теста является PCMARK Web-JunglePin [88] от компании FutureMark, который имитирует активность пользователя в некоторой абстрактной социальной сети, что по набору выполняемых действий сходно с работой в информационной системе. Результат представляется в виде скорости загрузки страницы в секундах. Примером обработки файла может быть его архивирование в тестах 7-Zip CPU Benchmark [87].

Для оценки оперативной памяти существует также набор тестов (PassMark memory test, AIDA64, WinRAR и др.), однако приоритетным параметром является объем доступной памяти для АИС. Для определения минимально необходимого объема существуют различные расчетные формулы и рекомендации, например в статье [90].

Следующим важным аппаратным компонентом, влияющим на производительность АИС, является используемый для хранения накопитель – жесткий диск или SSD. Характеристики накопителя определяют емкость электронного хранилища и скорость обработки, передачи и получения данных [91]. Поэтому в ходе оценки качества аппаратного обеспечения определяющими параметрами являются объем накопителя, скорость получения и записи данных. Для получения этих данных достаточно спецификации на оборудование и результатов тестов быстродействия жестких дисков (ATTO Disk Benchmark, CrystalDiskMark, HDD Tune Pro, IOmeter и др.) [89].

Остальное оборудование – материнские платы, сетевые карты, блоки питания, устройства ввода-вывода, периферия и т.д. оцениваются по спецификациям, указанным производителем, так как это в полной мере описывает их характеристики.

Второе направление оценки производительности – это анализ разработанного программного обеспечения. Данная оценка может быть реализована самостоятельно в ходе тестирования АИС – тогда улучшение/ухудшение данной метрики отражает повышение или снижение производительности системы. Рассмотрим также существующие подходы и методы оценки производительности АИС.

Различные подходы к оценке производительности информационных систем рассмотрены в работе [92]. Приводятся следующие метрики: Apdex, Opdex, Mpdex. Авторы отдают предпочтение последней:

$$Mpdex = \frac{n_1 + \frac{n_2}{2} + \frac{k}{mn_3}\sum_{i=1}^{n_3}(t_i - Z)}{n_1 + n_2 + n_3}, \qquad (2.18)$$

где n_1, n_2, n_3 – количество замеров производительности, которые попали соответственно в диапазоны от 0 до мягкого порога T, от T до жест-

кого порога Z, больше Z; k – эмпирический коэффициент; t_i – замеры производительности для времени больше T; m – коэффициент, равный Z/T.

Таким образом, вопрос производительности также оказывает существенное влияние на разработку и функционирование АИС. Допущенные ошибки при оценке производительности и выборе необходимых параметров аппаратного обеспечения могут привести к нестабильной работе и сбоям системы под высокой нагрузкой.

2.4. РАЗРАБОТКА КОМПЛЕКСНОГО КРИТЕРИЯ ОПТИМИЗАЦИИ АДАПТИВНЫХ ИНФОРМАЦИОННЫХ СИСТЕМ

Проведенный анализ существующих подходов к оценке АИС по различным направлениям показал необходимость формулирования комплексного критерия, с помощью которого возможно осуществить задачу оптимизацию. Ввиду необходимости всесторонней оценки системы, в состав критерия будут входить несколько различных метрик, т.е. предполагается решение многокритериальной задачи оптимизации.

На первом этапе сформулируем основные компоненты комплексного критерия, на втором – определим подходы к решению многокритериальной задачи.

На основе проведенного анализа существующих оценок АИС были выбраны следующие компоненты комплексного критерия, которые могут использоваться при оптимизации АИС:

- экономические затраты на разработку;
- сложность программного обеспечения;
- оценка адаптивности;
- оценка качества;
- производительность.

Перед тем как осуществить формализацию критериев, примем некоторые условные обозначения [50, 53]:

$U = \{u_i \mid i = 1, \ ..., \ I\}$ – множество информационных объектов;

$P = \{p_q \mid q = 1, \ ..., \ Q\}$ – множество пользователей АИС;

$O = \{o_l \mid l = 1, \ ..., \ L\}$ – множество операций.

Рассмотрим подходы к расчету данных критериев [53].

Экономические затраты на разработку АИС складываются из стоимости необходимого аппаратного обеспечения серверного и клиентского оборудования, оплаты работы разработчиков, тестировщиков, экспертов и аналитиков, а также дополнительных расходов (лицензи-

рование, НИР, испытания и т.д.). Формализуем компонент экономических затрат в следующем виде:

$$R_V = V_{hw} + V_{sw} + V_{pers} + V_{data} + V_{rd}, \quad (2.19)$$

где V_{hw} – затраты на серверное оборудование и терминалы пользователей (включая эксплуатационные и прочие затраты); V_{sw} – затраты на программное обеспечение (в том числе лицензирование); V_{pers} – заработная плата персонала, участвующего в разработке АИС; V_{data} – стоимость работы с информацией (хранение, обработка и передача); V_{rd} – стоимость НИР, необходимых для реализации АИС.

Следующий критерий – сложность программного обеспечения, входящего в состав АИС. Данный критерий оказывает влияние на трудоемкость процесса разработки (и, следовательно, на общие экономические затраты), а также на вероятность возникновения ошибок в процессе функционирования и модернизации системы.

Для оценки сложности программного обеспечения используем следующую формулу:

$$R_D = \sum_{m=1}^{M} DF_m, \quad (2.20)$$

где DF_m – значение комплексной метрики сложности m-го программного метода, определяемой как сумма нормированных метрик:

$$\begin{aligned} DF_m = {} & \lambda_1 d_m^{COCOMO} + \lambda_2 d_m^{J} + \lambda_3 d_m^{HAL} + \\ & + \lambda_4 d_m^{CC} + \lambda_5 d_m^{CH} + \lambda_6 d_m^{ABC} + \lambda_7 d_m^{MI}, \end{aligned} \quad (2.21)$$

где d_m^{COCOMO} – оценка трудоемкости проекта по методике COCOMO [73, 74]; d_m^{J} – оценка сложности программного кода по методике Джилба [70]; d_m^{HAL} – оценка сложности программного кода по методике Холстеда [77]; d_m^{CC} – цикломатическая сложность программного кода [69]; d_m^{CH} – метрика Чепина [67]; d_m^{ABC} – метрика ABC [67]; d_m^{MI} – индекс поддерживаемости кода [78]; $\lambda_1 - \lambda_7$ – нормирующие весовые коэффициенты, приводящие значения DF_m к диапазону [0; 1].

Комплексный подход к расчету сложности программного обеспечения каждого модуля позволит на этапе разработки АИС оценить

сложность различных вариантов программного кода и выбрать среди них с наименьшей совокупной сложностью, что положительно скажется на процессе как разработки, так и поддержки АИС. Количество метрик M может варьироваться в зависимости от условий задачи.

Оценка адаптивности АИС R_A представляет собой линейную свертку комплекса критериев адаптивности [15] и эргономических критериев [11]:

$$R_A = \sum_{i=1}^{6} \lambda_i ra_i + \sum_{j=1}^{18} \lambda_j re_j, \qquad (2.22)$$

где λ_i – весовой коэффициент i-го критерия адаптивности; $ra_1 = (t_{\max} - t)/t_{\max}$ – критерий оценки времени получения доступа пользователя к АИС (время авторизации), определяемый как отношение времени t при текущей конфигурации программного и аппаратного обеспечения пользователя к наибольшему $t_{\max}$ для всех конфигураций; $ra_2 = (n/n_{\max})$ – критерий функциональности, определяемый отношением количества реализованных функциональных блоков (модулей) n к их максимальному числу $n_{\max}$ в данной предметной области; $ra_3 = \sum qg_i / nU, i = 1, ..., nU$ – критерий гибкости, основанный на оценке qg_i пользователей (количеством nU), заданный в диапазоне от 0 до 1 и отражающий способность интерфейса адаптироваться под потребности пользователей с разной конфигурацией оборудования; $ra_4 = (nTest - nError)/nTest$ – критерий стабильности, определяемый количеством ошибок $nError$, совершенных пользователей в ходе $nTest$-тестов; $ra_5 = \sum qd_i / nU, i = 1, ..., nU$ – критерий доступности, основанный на оценке qg_i пользователей (количеством nU), заданный в диапазоне от 0 до 1 и отражающий качество навигации, простоту установки АИС, удобство интерфейса с позиции четкости текста, его размера и оформления; $ra_6 = \sum qk_i / nU, i = 1, ..., nU$ – критерий качества поддержки, основанный на оценке qk_i пользователей (количеством nU), заданный в диапазоне от 0 до 1 и отражающий наличие необходимых вспомогательных средств поддержки пользователя (поисковых или справочных подсистем), а также оценку пользователей по скорости освоения системы; λ_j – весовой коэффициент j-го эргономиче-

ского критерия; re_j – экспертная оценка по 18 эргономическим критериям [11].

Качество работы АИС R_Q определяется экспертной оценкой по двум категориям метрик – количественным (QN) и качественным (QLT):

$$R_Q = \alpha \sum \omega_i qn_i + \beta \sum \omega_j qlt_j, \tag{2.23}$$

где α, β – весовые коэффициенты, определяющие приоритет количественных метрик над качественными; ω_i, ω_j – весовые коэффициенты количественных и качественных метрик соответственно; $qn_i \in QN$ – расчетное значение количественных метрик надежности, безотказности, долговечности, ремонтопригодности, достоверности, эффективности [61, 62]; $qlt_j \in QTL$ – экспертная оценка качественных метрик: целостность, сложность, структурированность, адаптивность, лабильность, интегрируемость, делимость, валидность [63].

Производительность АИС будет складываться из оценок производительности программного P_{SW} и аппаратного P_{HW} обеспечения системы:

$$R_P = \frac{P_{SW} + P_{HW}}{2}. \tag{2.24}$$

Для оценки производительности программного обеспечения АИС используем метрику на основе *Mpdex*:

$$P_{SW} = \frac{1}{M} \sum_{m=1}^{M} \frac{n_m^+ + \frac{n_m^0}{2} + \frac{k^-}{n_m^-} \sum_{i=1}^{n_m^-} (t_{m,i} - t_{\max})}{n_m^+ + n_m^0 + n_m^-}, \tag{2.25}$$

где M – количество тестируемых модулей (программных методов) АИС; $t_{m,i}$ – время решения задачи m-м модулем; n_m^+ – количество замеров производительности m-го модуля, в которых время решения задачи меньше заданного порога $t_{\max}$ (положительный результат); n_m^0 – количество замеров производительности m-го модуля, в которых время решения задачи находится в диапазоне $t_{\max} \le t_{m,i} \le 2t_{\max}$ (нейтральный результат); n_m^- – количество замеров производительно-

сти m-го модуля, в которых время решения задачи превышает порог $2t_{\max}$ (отрицательный результат); k^- – весовой коэффициент, подбираемый эмпирически.

Для оценки производительности аппаратного обеспечения АИС используем линейную свертку метрик:

$$P_{HW} = \frac{\sum_{k=1}^{K} p(hw_k^{CPU}) + p(qhw_k^{RAM}) + p(chw_k^{net}) + p(chw_k^{ROM})}{4K}, \quad (2.26)$$

где $p(hw_k^{CPU})$ – нормированное значение функции оценки времени обработки запроса для получения информации при центральном процессоре hw_k^{CPU} k-го хранилища; $p(qhw_k^{RAM})$ – нормированное значение функции оценки объема оперативной памяти k-го хранилища; $p(chw_k^{net})$ – нормированное значение функции оценки пропускной способности сети от k-го хранилища до клиента; $p(chw_k^{ROM})$ – нормированное значение функции оценки пропускной способности накопителя k-го хранилища.

Рассмотренные критерии R_V, R_D, R_A, R_Q, R_P позволяют получить комплексную оценку АИС экономических затрат, сложности реализации, адаптивности, качества и производительности.

В ходе формирования комплексного критерия оптимизации АИС получено некоторое множество метрик, для каждого из которых необходимо определить экстремум. Однако, как и большинство многокритериальных задач оптимизации, найти единственное, наилучшее решение в данной ситуации невозможно. Так, например, метрики качества и производительности всегда противопоставлены экономической эффективности, следовательно, улучшение одного из них приведет к ухудшению значений остальных.

Пусть задано множество возможных решений X в виде векторов значений варьируемых параметров, а также множество критериев Y. Критерии имеют различную размерность и важность (которая может меняться в зависимости от решаемой задачи предметной области). При решении задачи оптимизации необходимо разрешить вопрос приоритетов одних критериев над другими [66]. Перечислим основные методы решения многокритериальных задач:

- Оптимальность по Парето.
- Принцип идеальной точки.
- Принцип равенства.
- Принцип максимина.

- Метод уступки.
- Принцип главного критерия.
- Лексикографический принцип.

Среди рассмотренных методов решения многокритериальных задач оптимизации не существует однозначно универсального и правильного, однако проведенный анализ позволяет сформулировать некоторые рекомендации к решению задачи оптимизации АИС.

На первом этапе возможно определение парето-оптимального множества допустимых решений, на втором – выбор главного критерия, обладающего наибольшей значимостью, и перевод остальных критериев в разряд ограничений. Варьируя пороговые значения ограничений, возможно сузить область допустимых решений, обеспечив нахождение близкого к оптимальному варианта. Возможно использование также лексикографического метода или метода уступок с учетом ранжирования критериев. В качестве наиболее важного критерия зачастую используется экономическая эффективность [93 – 97]. Данный выбор достаточно обоснован в трудах многих исследователей и на практике лучше всего поддается объективной оценке, особенно при наличии множества строго сформулированных ограничений, не позволяющих получить неоптимальное по остальным критериям решение.

Полноценное использование метода Парето затруднено следующим фактом: метод Парето используют, когда критерии равнозначны и выделить наиболее важный из них невозможно. Однако рассмотренные выше компоненты комплексного критерия не являются таковыми и можно говорить о превосходстве ряда метрик над другими, а также наличии между ними зависимостей. Кроме того, в редких случаях парето-оптимальное множество сводится к единственному решению, что требует привлечения экспертов и специалистов для окончательного решения.

Однако рассмотренные выше рекомендации несут лишь общий характер. Любой из рассмотренных методов решения задачи оптимизации может использоваться в зависимости от сложности решаемой задачи и специфики предметной области.

На основе проведенного в первой главе анализа, а также необходимости нахождения не только отдельных параметров АИС, но и выбора структурных элементов, решено остановиться на методе структурно-параметрического синтеза, так как большинство информационных систем содержат компоненты параметрической оптимизации, а структура при этом задается на этапе проектирования и в дальнейшем ее изменение связано с большими затратами как времени, так и денежных средств. Использование метода структурно-параметрического синтеза направлено на устранение этой фундаментальной проблемы проектирования информационных систем [98].

2.5. ФОРМАЛИЗОВАННАЯ ПОСТАНОВКА НАУЧНОЙ ЗАДАЧИ

В формализованном виде задача структурно-параметрического синтеза АИС имеет следующий вид.

Необходимо определить такое множество элементов структуры NNA, связей между элементами D, методов анализа, обработки, генерации и передачи информации MTD, а также множество параметров PRM для каждого элемента структуры АИС, при которых целевая функция эффективности R АИС достигает максимума:

$$\{NNA^*,\ PRM^*\} = \arg\max_{NNA,\ PRM}(R), \tag{2.27}$$

$$NNA^* = (E^*,\ C^*,\ M^*,\ V^*,\ NN^*,\ D^*,\ MTD^*) \tag{2.28}$$

при выполнении:

– множеств ограничений SO на связи между элементами структуры NNA:

$$SO = \{so_i\},\ so_i \to \gamma_i(D) \subseteq D^*; \tag{2.29}$$

– множеств ограничений PO на область значений параметров:

$$PO = \{po_i\},\ po_i \to \begin{cases} \varphi_i(PRM^*) = 0, \\ \varphi_i(PRM^*) < 0, \\ \varphi_i(PRM^*) \in PRM_i; \end{cases} \tag{2.30}$$

– множеств ограничений RO на максимальные и минимальные границы значений оценок качества АИС:

$$RO = \{ro_i\},\ ro_i \to \alpha_i^{\min} \le \phi_i(R) \le \alpha_i^{\max}, \tag{2.31}$$

где $\gamma_i(D)$ – функция, формирующая подмножество связей в соответствии с условием so_i, которые должны входить в сформированное множество D^*; $\varphi_i(PRM^*)$ – функция, преобразующая параметры АИС к виду, в котором их можно использовать в равенствах, неравенствах или для проверки вхождения в некоторые подмножества PRM_i; $\phi_i(R)$ – функция, преобразующая целевую функцию эффективности или отдельные ее компоненты для проверки соответствия минимальным $\alpha_i^{\min}$ и максимальным $\alpha_i^{\max}$ допустимым границам.

Под функцией эффективности R АИС в общем виде будем понимать комплексную оценку системы следующего вида:

$$R = FR(R_V,\ R_D,\ R_A,\ R_Q,\ R_P), \qquad (2.32)$$

где FR – метод поиска оптимального решения в многокритериальной задаче оптимизации; R_V – оценка экономических затрат на разработку АИС, определяемых формулой (2.19); R_D – оценка сложности программного обеспечения АИС (2.20); R_A – оценка адаптивности АИС (2.22); R_Q – оценка качества АИС (2.23); R_P – оценка производительности АИС (2.24).

В качестве метода решения многокритериальной задачи возможно использование метода главного критерия с переводом остальных критериев в ограничения. Пусть таким критерием выступает оценка экономических затрат R_V, тогда постановка задачи (2.27) примет вид

$$\{NNA^*,\ PRM^*\} = \underset{NNA,\ PRM}{\arg\min}\,(R_V), \qquad (2.33)$$

с учетом добавления новых ограничений:

$$R_D \le R_D^{\max},\ \ R_A \ge R_A^{\min},\ \ R_Q \ge R_Q^{\min},\ \ R_P \ge R_P^{\min}, \qquad (2.34)$$

где $R_D^{\max}$ – оценка сложности программного обеспечения, реализованного на основе классического подхода без использования нейросетевых методов; $R_A^{\min}$ – минимальная оценка адаптивности либо оценка программного обеспечения, реализованного на основе классического подхода без использования нейросетевых методов; $R_Q^{\min}$ – минимально допустимая оценка качества программного обеспечения, реализованного на основе классического подхода без использования нейросетевых методов; $R_P^{\min}$ – минимально возможная производительность системы.

2.6. ВЫВОДЫ

В главе 2 рассмотрены основные подходы к анализу, обработке и передачи информации. Выявлены основные методы представления информации, необходимые процедуры предварительной обработки информации, позволяющие в дальнейшем повысить эффективность процессов ее анализа и преобразования. Рассмотрены существующие способы организации межмодульной связи, передачи и распределения данных.

Для решения задачи формализации структуры АИС и протекающих в ней процессов проанализированы основные типы моделей:

теоретико-множественные, теоретико-графовые, автоматные, мультиагентные. Рассмотрены примеры представления структуры АИС в нотации данных моделей, а также опыт их применения в работах других исследователей. Проведенный анализ различных типов моделей позволяет сделать следующий вывод о их применимости для формализации структуры АИС.

Теоретико-множественные модели отличаются простотой и позволяют описывать протекающие в предметной области процессы с достаточной полнотой, но в них отсутствует графическое представление.

Теоретико-графовые модели содержат формализованное структурное представление системы и процессов в нотации теории множеств, графическое представление в виде графов, что, несомненно, делает этот тип моделей перспективным и применимым в большинстве ситуаций.

Автоматные модели используют тот же принцип, но опираются не на теорию множеств, а на теорию автоматов, могут быть дополнены теорией графов. С помощью автоматной модели можно наглядно представить процессы смены состояний информационных объектов, но при большем количестве связей между объектами и наличием множества функций преобразования информации модель становится слишком громоздкой и сложной для понимания.

Модели на основе интеллектуальных агентов (мультиагентные) в ряде задач показывают свою эффективность, так как позволяют декомпозировать структуру системы на отдельные независимые фрагменты. Однако агенты являются дополнительным слоем абстракции над известными математическими и алгоритмическими подходами, поэтому их использование может дополнительно усложнить формализацию и реализацию информационной системы.

Далее рассмотрены основные подходы к оценке АИС. Выделены четыре направления: экономическая эффективность, качество, сложность разработки и производительность. Для каждого направления рассмотрены основные метрики и методики их расчета.

Проведенный анализ позволил выявить основные компоненты комплексного критерия оптимизации АИС, в состав которого вошли оценки экономических затрат, сложности реализации, адаптивности, качества и производительности АИС. Рассмотрены подходы к решению многокритериальной задачи оптимизации, позволяющие учесть все компоненты комплексного критерия. Выявлены основные компоненты и методы, которые будут использоваться при построении математических моделей АИС различных предметных областей, постановке и решении задач их оптимизации. Сформулирована и формализована задача структурно-параметрического синтеза АИС.

Глава 3
МЕТОДОЛОГИЯ СТРУКТУРНО-ПАРАМЕТРИЧЕСКОГО СИНТЕЗА АДАПТИВНЫХ ИНФОРМАЦИОННЫХ СИСТЕМ НА ОСНОВЕ НЕЙРОСЕТЕВОЙ АРХИТЕКТУРЫ И МЕТОДОВ

На основе проведенного анализа методологий разработки программного обеспечения информационных систем определена необходимость в разработке новой методологии структурно-параметрического синтеза АИС, объединяющей наиболее эффективные подходы в области декомпозиции архитектур информационных систем и применения автоматизированных методов работы с информацией.

Исследование методов анализа, обработки, генерации и передачи информации в различных информационных системах показало перспективность применения технологий машинного обучения для автоматизации данных процессов.

Однако в настоящее время не существует методологии и методов, направленных на автоматизацию процесса структурно-параметрического синтеза АИС. Для решения данного вопроса в целях повышения эффективности разработки АИС предлагается комплексный подход, включающий разработку новой методологии структурно-параметрического синтеза АИС, реализованной на основе нейросетевых методов и нейросетевой архитектуры системы.

3.1. ТЕРМИНЫ И ПОНЯТИЯ МЕТОДОЛОГИИ

На первом этапе реализации методологии структурно-параметрического синтеза АИС сформулируем основные термины и понятия.

Адаптивная информационная система – система, автоматически изменяющая данные алгоритма своего функционирования и структуру в целях сохранения или достижения оптимального состояния при изменении внешних условий.

Адаптация информационной системы – процесс накопления и использования информации в системе, направленный на достижение оптимального состояния системы при изменяющихся внешних условиях.

Адаптивность – способность системы изменяться для сохранения своих эксплуатационных показателей в заданных пределах при изменениях внешней среды.

Нейросетевая архитектура (Neural Network Architecture, NNA) – структура информационных систем, включающая следующие ключе-

вые сущности: Окружение, Нейронные сети, Модель, Представление и Управление, обеспечивающая возможность их изолированной и независимой разработки.

Окружение (Environment, E) – комплексная структура, включающая внешние факторы воздействия на информационную систему, сведения о пользователях системы, программно-аппаратные характеристики их терминалов, права доступа к информации и принимающая участие в процессах управления и воздействия на другие сущности.

Нейронные сети (Neural Networks, NN) – некоторое множество алгоритмов машинного обучения на основе искусственных нейронных сетей, их математическое, алгоритмическое и программное обеспечение, используемое для решения задач генерации, обработки и передачи информации, автоматизированного синтеза структуры и параметров компонентов информационной системы и реализации их межмодульного взаимодействия.

Модель (Model, M) – математическое, даталогическое и алгоритмическое обеспечение информационной системы, объединенное в соответствии с направлением решаемых задач и изолированное от других сущностей архитектуры.

Представление (View, V) – текстовое, графическое или иное отображение информации, формируемое в соответствии с запросами пользователей либо с состоянием системы.

Управление (Control, C) – совокупность управляющих блоков, включающих алгоритмическое, математическое и программное обеспечение, по заданным соотношениям, осуществляющих управление информационной системой и взаимодействие между остальными компонентами архитектуры под влиянием Окружения.

Модуль – элемент сущности (или некоторое их множество), направленный на решение конкретной задачи хранения, обработки или передачи данных.

Нейросетевой метод – метод, базирующийся на применении различных типов нейронных сетей.

Генерация информации в АИС – формирование нового информационного объекта в АИС с применением различных технологий обработки данных.

3.2. ОСНОВНЫЕ ПРИНЦИПЫ МЕТОДОЛОГИИ

Анализ существующих методологий разработки информационных систем позволил сформулировать перечень требований и ограничений для разрабатываемой методологии. Их выполнение позволит

решить поставленные задачи, связанные с автоматизацией процесса реализации АИС.

Возможность автоматизации процесса структурно-параметрического синтеза АИС основывается на выполнении предлагаемого **принципа нейросетевой организации структуры АИС**, который определяет, что структура АИС представлена в виде множества изолированных и независимых сущностей, сгруппированных по пяти категориям: Окружение, Нейронные сети, Модель, Представление и Управление. Каждый элемент АИС для взаимодействия с другими компонентами и передачи информации использует нейросетевые каналы данных. Структура АИС предполагает возможность динамического изменения параметров элементов АИС в зависимости от внешних факторов. Принцип нейросетевой организации структуры АИС основан на выполнении следующих требований к архитектуре АИС:

– декомпозиция информационной системы в виде совокупности ключевых сущностей, связанных между собой унифицированными каналами передачи и обработки информации, функционирующими на основе технологий искусственного интеллекта;

– изолированность и независимость сущностей информационной системы, что позволяет осуществить декомпозицию задачи программной реализации АИС, параллельную автоматизированную разработку модулей системы, модернизацию одних компонентов без влияния на работоспособность других;

– отделение математического и алгоритмического обеспечения функционирования информационной системы от математической модели данных и информационных потоков, протекающих в предметной области, т.е. разграничение информации и процессов ее обработки;

– необходимость анализа и учета влияния пользователя и внешней среды на процессы движения информационных потоков, элементы управления и визуального представления информационной системы для последующей формализации процессов маршрутизации данных, адаптации интерфейса под индивидуальные особенности пользователя и программно-аппаратные характеристики его оборудования, обеспечение саморегулирования и устойчивости системы.

Осуществив декомпозицию АИС в соответствии с изложенным принципом, необходимо решить задачу автоматизации разработки каждого отдельного компонента структуры системы. В процессе реализации модулей АИС можно выделить четыре основные операции: анализ, обработка, генерация и передача информации.

Принцип подготовки и соответствия информации определяет, что для реализации автоматизированных методов анализа, обработки,

генерации и передачи, функционирующих на основе нейронных сетей, в АИС необходимо обеспечить предварительную обработку, корректность, достаточность и соответствие исходных данных для последующего обучения нейронных сетей и алгоритмов искусственного интеллекта.

Принцип подготовки и соответствия информации основан на выполнении следующих положений и условий:

– Для обеспечения работы автоматизированных методов и алгоритмов работы с данными необходимо обеспечить корректность структуры данных. Исходные данные могут быть получены из разных мест, иметь различные форматы и структуру, более того, часть из них может иметь ошибки и неточности, быть искажена. Наиболее распространенными проблемами является неполнота (часть значений или атрибутов отсутствует), зашумленность (ошибочность некоторых значений) и несогласованность (конфликты и противоречия между данными).

– Для повышения точности и снижения вычислительной сложности автоматизированных методов и алгоритмов необходимо осуществить предварительную обработку данных, которая включает следующие операции: преобразование данных и фильтрация данных.

– Для обеспечения работы методов на основе машинного обучения и искусственного интеллекта необходимо обеспечить достаточность и соответствие структуры исходных данных, что выражается выполнением следующих условий: набор данных для обучения нейронных сетей задан на области определения возможных значений входных и выходных данных; каждому входному вектору данных должен соответствовать выходной вектор, причем единственный; содержимое входных и выходных данных можно привести к некоторому численному значению.

Принцип применимости нейросетевой архитектуры заключается в возможности использования методов и моделей, основанных на подходах теории искусственного интеллекта, и формирует перечень требований к достаточности исходных данных, формализации процессов обработки информации, возможности применения аппроксимирующих функций и необходимости реализации функций адаптивности в выбранной предметной области. Принцип определяет следующие условия, не позволяющие осуществить применение методологии:

– отсутствие необходимого объема исходных данных, несоответствие их структуры и качества для использования в нейронных сетях для решения задач анализа и обработки информации;

– отсутствие формализованных закономерностей в процессах обработки информации в предметной области, т.е. для каждой проце-

дуры обработки данных четко сформулированы в алгоритмическом, математической или ином виде определенные зависимости;

– высокие требования к надежности функционирования и точности обрабатываемых данных, не допускающие использования приближенных или аппроксимированных значений, возможных при использовании методов искусственного интеллекта;

– отсутствие потребности в функциях адаптивности и персонализации, оптимизации интерфейса и параметров АИС для пользователей, либо полное отсутствие взаимодействия с пользователем в случае автоматических систем управления.

Выполнение одного или нескольких условий ограничивает возможность применения методологии для осуществления синтеза АИС.

Следование определенным принципам позволяет сформулировать общую методологию автоматизированного структурно-параметрического синтеза АИС на основе нейросетевой архитектуры. Данная методология охватывает широкий класс систем различной сложности. Достигается это за счет разработки отдельных универсальных автоматизированных методов и алгоритмов анализа, обработки, генерации и передачи информации.

При разработке методологии учитывались наиболее актуальные и универсальные подходы к разработке программного обеспечения, термины и понятия, принятые в сфере программирования. Однако использование нейронных сетей как центрального элемента процесса структурно-параметрического синтеза АИС, комплексный подход к автоматизации процессов анализа, обработки, генерации и передачи информации с использованием нейронных сетей предлагается впервые.

3.3. НЕЙРОСЕТЕВАЯ АРХИТЕКТУРА АИС

На основе перечисленных выше основных принципов формализуем нейросетевую архитектуру информационных систем [143] в виде структурной схемы (рис. 3.1)

Окружение как независимая сущность объединяет сведения о множествах пользователей (их правах доступа, личных данных, спецификации программного и аппаратного обеспечения) и внешних факторов воздействия. С одной стороны, Окружение влияет на систему извне, адаптируя ее под особенности каждого пользователя и его личные предпочтения, с другой стороны – сами пользователи являются участниками процессов создания и обработки информации, исполнителями или наблюдателями, что выражается в Модели как ссылка на конкретного пользователя из Окружения. Напрямую при этом Модель

и Окружение не взаимодействуют. Пользователи, входящие в Окружение, в процессе работы с информационной системой активно контактируют с сущностью Управление. При этом в зависимости от уровня доступа пользователя в Управлении ему доступны только необходимые управляющие блоки и функции.

Управление является связующим звеном между остальными сущностями. В нем происходит обработка команд от пользователей, отслеживание внешних воздействий, прием и передача данных в другие компоненты информационной системы, управление возникающими в системе событиями, регулирование параметров в соответствии с требованиями. Управление активно взаимодействует с Моделью: передает ей новые данные, получает необходимую информацию из хранилища, осуществляет контроль за обработкой и дальнейшей передачей информации из Модели в Представление. В Управлении также реализована некоторая часть интерфейса, ответственная за генерацию событий, получение и отправку данных.

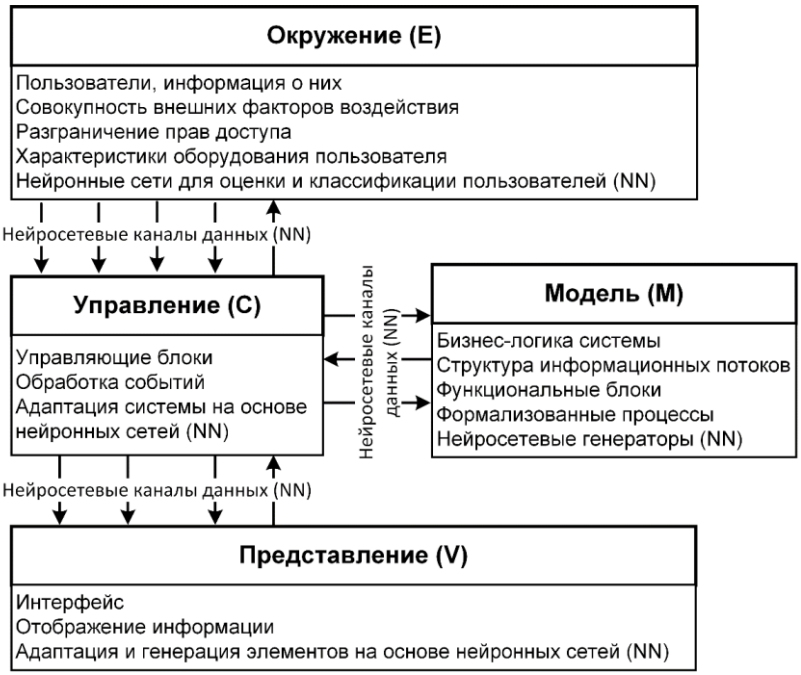


Рис. 3.1. Структурная схема нейросетевой архитектуры

Модель является достаточно сложной структурой в рассматриваемой архитектуре, так как включает формализованное представление бизнес-логики системы, структуры информационных потоков, хранение информации, математическое и алгоритмическое обеспечение требуемых операций по обработке и преобразованию данных. Отдельные компоненты Модели могут взаимодействовать между собой, но некоторая изолированность между ними остается. Модель не является стационарной, она постоянно динамически развивается за счет появления новых данных, расширения структуры информационных потоков и добавления дополнительной функциональности в информационную систему.

Сущность Представление отвечает за визуализацию интерфейса для Окружения. Параметры интерфейса могут гибко меняться в зависимости от воздействия Управления, подстраиваясь под требования конкретных пользователей, оборудование или иные внешние воздействия. Через Управление передается необходимый объем данных, отображаемый в Представлении, а также управляющие блоки, адаптированные под текущие настройки интерфейса.

В состав каждой сущности входят компоненты, основанные на технологиях машинного обучения и относящиеся к сущности Нейронные сети. Также все сущности взаимодействуют между собой через Управление, что отражается на структурной схеме в связях между блоками. Подобные связи обозначим понятием «нейросетевые каналы данных» (NNDC) и также отнесем к сущности Нейронные сети.

Опишем концепцию функционирования нейросетевого канала данных. Канал устанавливает связь между двумя модулями: с модуля i поступают входные данные X заданной структуры STR_{SX}, на выходе из канала модулем j ожидаются данные Y структуры STR_{SY}. Для обеспечения этого соответствия нейронная сеть классификации типов данных анализирует структуры SX и SY входных и выходных данных и выбирает необходимую функцию преобразования типов. Далее осуществляются обработка и преобразование данных нейронной сетью, если это требуется в рамках установленной связи. Преобразованные данные передаются модулю j по протоколу передачи данных, определяемому третьей нейронной сетью.

Сформулировав общую структурную схему нейросетевой архитектуры, осуществим декомпозицию каждой сущности и, используя аппарат нейросетевых каналов данных, конкретизируем их взаимодействие.

Декомпозиция сущности Окружение представлена на рис. 3.2.

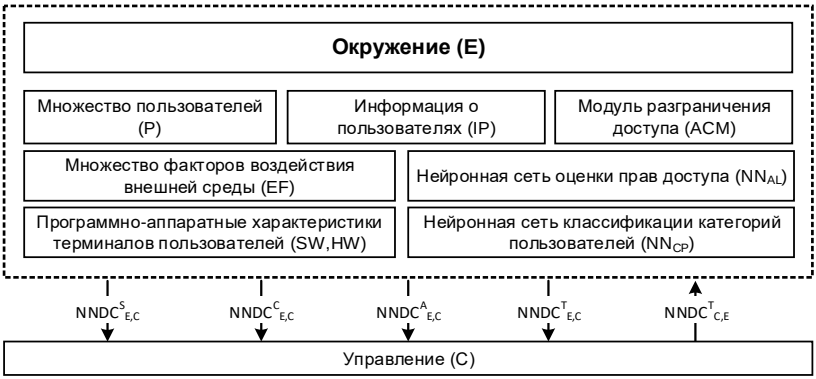


Рис. 3.2. Декомпозиция сущности Окружение

Множество факторов воздействия внешней среды объединяют в одном модуле математическое описание влияния внешнего воздействия на структуру и параметры информационной системы. К этим факторам относятся: действия пользователей, текущее состояние оборудования, на котором функционирует информационная система, модернизация системы, приводящая к необходимости изменения структуры и параметров отдельных модулей.

Модуль разграничения доступа (ACM) формализует в некотором (дискретном, ролевом или атрибутном) представлении множество уровней доступа AL к информации. В состав модуля входит нейронная сеть оценки прав доступа NN_{AL}, которая по формализованному представлению уровня доступа пользователя p_i позволяет получить некоторую общую оценку al_{ij}, используемую при решении задачи x_j разграничения доступа к конкретному документу, операции, компоненту системы и т.д.

Нейронная сеть NN_{CP} позволяет классифицировать пользователя u_i на основе данных из модуля «Информация о пользователях» IP (содержит сведения о личных данных, возрасте, образовании, должности, достижениях и т.д.) по заданным категориям cp_i, кластеризовать по общим признакам, обнаруженным в процессе анализа.

Следующий модуль содержит сведения о программных SW и аппаратных HW характеристиках терминалов пользователей и включает все необходимые сведения об оборудовании пользователя, его производительности, размере и разрешении экрана устройства, операционной системе, браузере (если информационная система

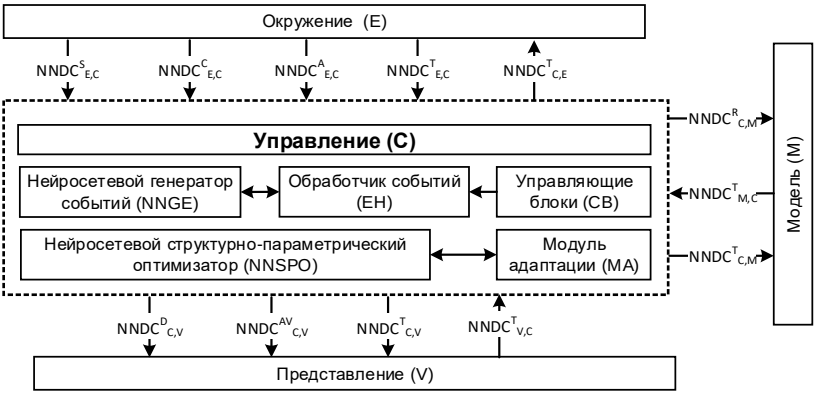


Рис. 3.3. Декомпозиция сущности Управление

реализована как Web-приложение), наборе дополнительного программного обеспечения.

Окружение через нейросетевые каналы данных может передавать всю необходимую информацию и команды пользователей Управлению, а также получать от него требуемые данные.

Сущность Управление (рис. 3.3) состоит из трех основных компонентов.

Управляющие блоки (*CB*) – совокупность элементов интерфейса и программных модулей, реализующая управляющие функции в *NNA*. Пользователи по нейросетевому каналу управления осуществляют передачу команд, что порождает необходимые события, работа с которыми осуществляется в обработчике событий.

Обработчик событий (*EH*) отвечает за реагирование компонентов системы на поступающие от пользователей команды по нейросетевому каналу управления. Обработчик работает совместно с нейросетевым генератором событий $NNGE(event_i) = \{(pd_{ij}, pl_{ij})\}$, программным модулем на основе нейронных сетей, формирующим программный код pd и логику pl обработки множества событий.

Модуль адаптации (*MA*) используется для настройки параметров информационной системы (например, интерфейса) под программно-аппаратные характеристики оборудования пользователей, а также для оптимизации системы в процессе ее функционирования и модернизации по выбранному набору критериев. Основой модуля адаптации является нейросетевой структурно-параметрический оптимизатор $NNSPO(SM, PM, R) = (SM^*, PM^*)$, который на основе текущих

структуры SM, параметров PM модулей системы и заданного множества критериев R подбирает оптимальные SM^* и PM^*. К функциям модуля адаптации относится также корректировка структуры информационных потоков вследствие структурных изменений в информационной системе.

Занимая центральное положение в нейросетевой архитектуре, Управление взаимодействует с остальными сущностями через множество нейросетевых каналов данных, с некоторыми эта связь двусторонняя.

Декомпозиция сущности Модель представлена на рис. 3.4. Каждый внутренний компонент Модели изолируется от остальных и связывается только через нейросетевые каналы данных, что позволяет при изменениях одного компонента не нарушить структуру и данные другого. Данное правило обеспечивает целостность данных и возможность независимого формирования каждой модели и замены их одну на другую. Анализ решаемых Моделью задач показал необходимость ее декомпозиции на четыре ключевых компонента: Модель информационных потоков (содержит данные о структуре, процессах движения и взаимодействия информационных объектов, их жизненных циклах), Модель функциональных блоков (содержит математические соотношения, реализующие определенные операции по обработке и преобразованию данных, направленные на решение конкретных задач в рамках программных модулей), Модель процессов (используется для анализа предметной области и автоматизированной генерации данных и модулей для других компонентов Модели) и Модель посредника (используется для упрощения механизма взаимодействия между Моделью и Управлением и координации операций приема-передачи данных с другими сущностями в одном блоке).

Модель информационных потоков MI будет отвечать только за данные о процессах движения информации, формализуемых с помощью многоуровневой модели графов $NNIF(MG) = DB$ [50]. Данная модель MG обрабатывается в нейросетевом генераторе информационных потоков $NNIF$, образуя необходимую структуру базы данных DB, которая далее в автоматическом формате заполняется данными о информационных объектах, их жизненных циклах, пользователях системы, служебной и отчетной информации и т.д.

Распределение информации в автоматическом режиме возможно за счет работы нейросетевого классификатора информации $NN_{CI}(\{a_{ij}\}) = cd_k$, $cd_k \to \{dbt_m\}$, по множеству атрибутов $\{a_{ij}\}$ поступивших данных, определяющего его категорию cd_k, к которой в свою очередь соотнесены таблицы базы данных $\{dbt_m\}$.

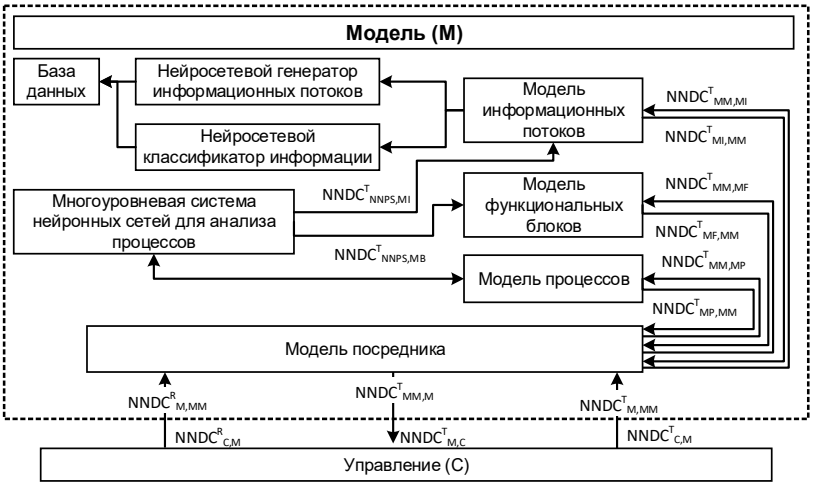


Рис. 3.4. Декомпозиция сущности Модель

Модель функциональных блоков MF включает набор готовых программных модулей, реализуя таким образом функции (интерфейсы) по обработке данных. Важным условием при реализации каждого модуля является его независимость и стандартизация входных $Data_{in}$ и выходных $Data_{out}$ данных, что позволит применять его для решения конкретной задачи в рамках любой предметной области и обеспечивать работоспособность на любых данных из области допустимых значений. Работу каждого функционального блока MF_i можно описать следующим соотношением: $MF_i(Data_{in}) = Data_{out}$.

Модель процессов MP формализует специфические процессы движения информации и на основе их анализа генерирует необходимые функциональные блоки или информационные потоки для остальных компонентов Модели, что позволяет системе самостоятельно адаптироваться под изменения бизнес-процессов организации. Для автоматизации такой сложной задачи планируется использование многоуровневой системы $NNPS$ нейронных сетей $\{NN_i\}$, анализирующих процессы предметной области PSA и генерирующих на их основе графы информационных потоков MG^* и структуры данных SMG_i функциональных блоков для соответствующих компонентов Модели, которые передаются по каналам $NNDC^T_{NNPS,\,MI}$ и $NNDC^T_{NNPS,\,MB}$.

Модель посредника MM используется для упрощения взаимодействия между сущностями таким образом, чтобы все потоки данных (как входящие, так и исходящие) проходили через нее, что позволяет сократить количество нейросетевых каналов данных между Управлением и Моделью. Таким образом, принимая от Управления запросы, Модель посредника определяет, к каким компонентам Модели относится этот запрос, после чего через множество каналов $NNC^{T}_{MM,X}$ (где X – некоторый элемент Модели из набора MI, MF, MP) получает необходимые данные $Data$ и передает их обратно Управлению в два этапа (сначала от посредника к Модели, потом от Модели к Управлению). Внешне со стороны Управления данный механизм не прослеживается, что позволяет использовать нейросетевой канал запроса данных с разными Моделями, так как всю архитектурную разницу примет на себя Модель посредника. Входящие извне данные также передаются не напрямую в Модель, а через посредника и далее в необходимые компоненты.

Рассмотрим взаимодействие Модели с другими сущностями, а также между ее компонентами. Внешняя связь ограничивается тремя нейросетевыми каналами данных с Управлением. Все основные компоненты связаны с Моделью посредника нейросетевыми каналами передачи данных, чтобы обеспечить передачу информации в двух направлениях между компонентами и Управлением. Два канала передачи данных осуществляют взаимодействие $NNPS$ с MG и MI. Также имеются три внутренних канала, принимающие и передающие данные на границе между Управлением и Моделью. Таким образом, все необходимые операции по передачи информации можно осуществлять через Модель посредника.

Подводя итог, отметим, что сущность Модели имеет самую сложную структуру движения информации, которую удалось упорядочить и сосредоточить в одном месте за счет использования Модели посредника.

Последняя сущность нейросетевой архитектуры – Представление. Ее декомпозиция показана на рис. 3.5. Основным компонентом Представления является Интерфейс – совокупность набора визуальных элементов v_i, отображаемых на устройстве пользователя в соответствии с требованиями, которые сформированы Управлением.

Формирование графической, текстовой, звуковой или иной информации на основе полученных от Управления данных для визуальных элементов возможно в автоматическом режиме на основе работы нейросетевого генератора интерфейса $NNGI(V_{type}, V_{param}) = \{v_i\}$.

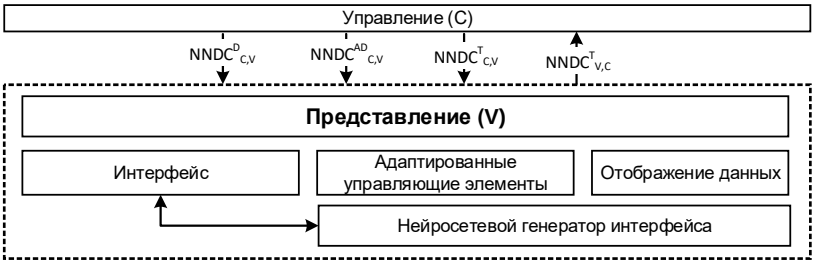


Рис. 3.5. Декомпозиция сущности Представление

Генератор интерфейса анализирует полученные по каналу управления отображением $NNDC^{D}_{C,\,V}$ тип и параметры интерфейса и выбирает из некоторой библиотеки готовые визуальные элементы v_i.

Второй задачей является автоматическая адаптация Представления под характеристики оборудования и используемое программное обеспечение пользователя, которая осуществляется последовательной передачей нужной информации сначала через канал адаптации $NNDC^{A}_{E,\,C}$, а потом по каналу управления отображением $NNDC^{D}_{C,\,V}$. При этом учитывается мощность оборудования, которое используется при выборе тех средств визуализации интерфейса, что обеспечат наилучшее пользовательское взаимодействие с системой, а также такие параметры, как размеры и разрешение экрана для компоновки элементов представления. Помимо визуальных элементов Представление включает также отображение полученных из Управления данных, выводимых в элементы интерфейса или непосредственно на экран, и адаптированные под визуальные требования управляющие элементы de_i. В нейросетевой архитектуре Представление взаимодействует лишь с Управлением по четырем нейросетевым каналам данных, рассмотренным выше.

На этом декомпозиция нейросетевой архитектуры завершается, дальнейший анализ будет связан уже с описанием структуры и закономерностей функционирования каждого из рассмотренных модулей, а также структур нейронных сетей, используемых для анализа, обработки и передачи информации.

Представленная нейросетевая архитектура обобщает рассмотренные подходы к формализации структуры информационных систем, объединяет существующие технологии (нейронные сети, микросервисы) и шаблоны проектирования (MVC), что позволяет использовать сильные стороны существующих решений. С другой стороны,

нейросетевая архитектура реализует новый тип организации связей между модулями, объединяющий анализ, обработку и передачи информации – нейросетевые каналы данных.

Научная новизна нейросетевой архитектуры заключается в следующем:

– впервые организация связей между модулями АИС реализована на основе программных нейросетевых компонентов, автоматизирующих процессы анализа, обработки и передачи данных;

– в отличие от существующих архитектур информационных систем, в частности шаблона MVC, нейросетевая архитектура дополнена сущностями Окружение и Нейронные сети, что позволило учесть влияние внешних факторов и пользователя на структуру и параметры системы, а также отделить автоматизированные процессы анализа, обработки и передачи данных от сущности Управление, что снизило связность сущностей;

– нейросетевая архитектура от отличие от существующих структур обладает совокупностью свойств: изолированность и независимость ключевых сущностей информационных систем; изолированность математического обеспечения; разграничение моделей информационных процессов и функциональных элементов от управляющих систем и систем представления информации; учет влияния пользователя как субъекта информационной системы на процессы движения информационных потоков, элементы управления и представления системы; возможность адаптации структурных блоков информационных систем под особенности предметной области, параметры оборудования пользователя без необходимости внесения существенных изменений в архитектуру.

Предлагаемая архитектура позволяет осуществить декомпозицию структуры информационной системы, параллельную и автоматизированную реализацию модулей, модернизацию отдельных компонентов информационных систем без влияния на работоспособность других подсистем, а также интегрировать искусственные нейронные сети для решения задач анализа, обработки и передачи информации, организации взаимодействия компонентов.

3.4. СТРУКТУРА МЕТОДОЛОГИИ

Рассмотрев существующие подходы к разработке информационных систем, их преимущества и недостатки применительно к процессу реализации АИС, перейдем к разработке концепции новой методологии. Основой является модернизация методологии RAD, так как ее основные принципы полностью отвечают поставленной цели: сниже-

ние стоимости и времени разработки информационных систем, ориентирование на небольшие коллективы.

Реализация методологии невозможна на основе лишь существующих методов системного анализа и разработки программного обеспечения. Для автоматизации таких задач, как анализ, обработка, генерация и передача информации в АИС, необходима адаптация, разработка и переосмысление существующих подходов по работе с данными и интеграция в них технологий машинного обучения. Это позволит снизить нагрузку на разработчика программного обеспечения при принятии решений, анализе информации, реализации типовых операций. Тогда, используя в качестве основы апробированную методологию RAD, расширим и дополним ее необходимыми этапами и методами, направленными на решение задачи автоматизации структурно-параметрического синтеза АИС.

Методология включает все основные этапы жизненного цикла АИС, начиная с планирования структуры системы и формализации предметной области, заканчивая программной реализацией, оптимизацией и модернизацией АИС (рис. 3.6).

В рамках методологии реализуются следующие **новые методы**:

– **метод формализации информационных потоков на основе моделей многоуровневых графов** для представления процессов обработки информации, исследования закономерностей движения и жизненного цикла информационных объектов, анализа и синтеза структуры информационных потоков системы;

– **нейросетевой метод обработки и передачи информации** для автоматизированной организации связи между компонентами нейросетевой архитектуры на основе реализации нейросетевых каналов данных, осуществляющих анализ, обработку и передачу данных с применением методов искусственного интеллекта;

– **нейросетевой метод автоматической генерации данных** на основе применения нейронных сетей для формирования информационных объектов с заданной структурой и свойствами с автоматическим поиском оптимальных параметров нейронных сетей;

– **нейросетевой метод автоматической переадресации информации** между пользователями АИС, отличающийся применением нейронных сетей для определения оптимального исполнителя операции на основе прогноза времени и вероятности ее успешного выполнения, позволяющий осуществить маршрутизацию информации;

– **нейросетевой метод классификации и распределения данных** для организации их автоматизированного хранения, поиска, проверки и сравнения;

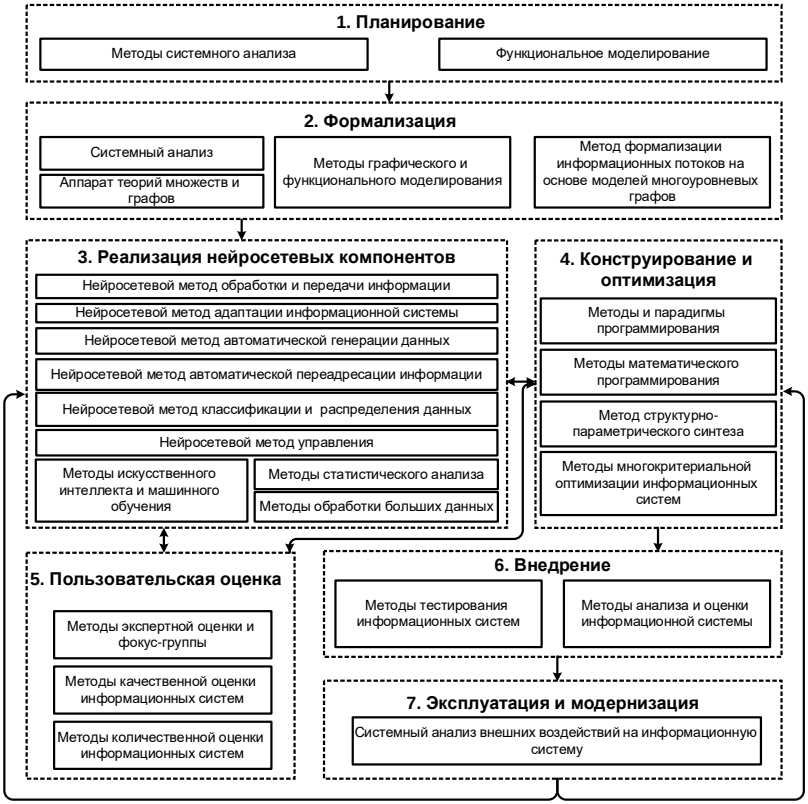


Рис. 3.6. Структура методологии структурно-параметрического синтеза АИС на основе нейросетевых методов

– **нейросетевой метод адаптации информационной системы** под индивидуальные особенности пользователя, основанный на технологиях искусственного интеллекта, анализе больших данных и машинном обучении;

– **нейросетевой метод управления** на основе применения нейронных сетей для автоматизации процессов поддержки принятия решений и управления в АИС.

Для реализации всех этапов методологии применяются также существующие методы:

– методы системного анализа (декомпозиция, формализация, абстрагирование, анализ, синтез, алгоритмизация) и функционального моделирования (IDEF0, UML) для исследования процессов предмет-

ной области, формализации структур компонентов системы, описания функциональных компонентов системы;

– методы графического и функционального моделирования, теории графов для реализации структурных и функциональных моделей компонентов информационных систем на основе нейросетевой архитектуры;

– методы теоретико-множественного анализа для разработки математических моделей АИС на базе нейросетевой архитектуры и формализации процессов движения информации в них;

– методы искусственного интеллекта, в том числе машинного обучения, для автоматизации процессов генерации, классификации и обработки информации, поддержки принятия решений, реализации программных модулей на базе нейронных сетей;

– методы и парадигмы функционального, объектно-ориентированного и процедурного программирования для практической реализации информационных систем;

– методы математического программирования, структурно-параметрического синтеза, подходы к решению многокритериальных задач (аналитические и численные) для постановки задачи оптимизации информационных систем;

– методы качественной и количественной оценки информационных систем для реализации методики оценки эффективности процесса разработки, эксплуатации информационных систем и взаимодействия пользователей с ними;

– методы тестирования информационных систем, оценки информационных систем с привлечением экспертов и фокус-групп для получения субъективных оценок качества функционирования полученных решений;

– методы статистического анализа и обработки больших данных для извлечения и использования информации о предметной области, деятельности пользователей, массивах хранящихся данных.

3.5. ОСНОВНЫЕ ЭТАПЫ МЕТОДОЛОГИИ

Рассмотрим основные этапы предлагаемой методологии. Формализуем ее в нотации диаграмм IDEF0, тогда общая структура методологии структурно-параметрического синтеза АИС примет вид, представленный на рис. 3.7. В основе реализации методологии семь основных блоков, соответствующих структуре методологии (рис. 3.6). Рассмотрим их декомпозицию.

А1. Планирование. На данном этапе осуществляется системный анализ предметной области, направленный на получение требований к информационной системе (рис. 3.8).

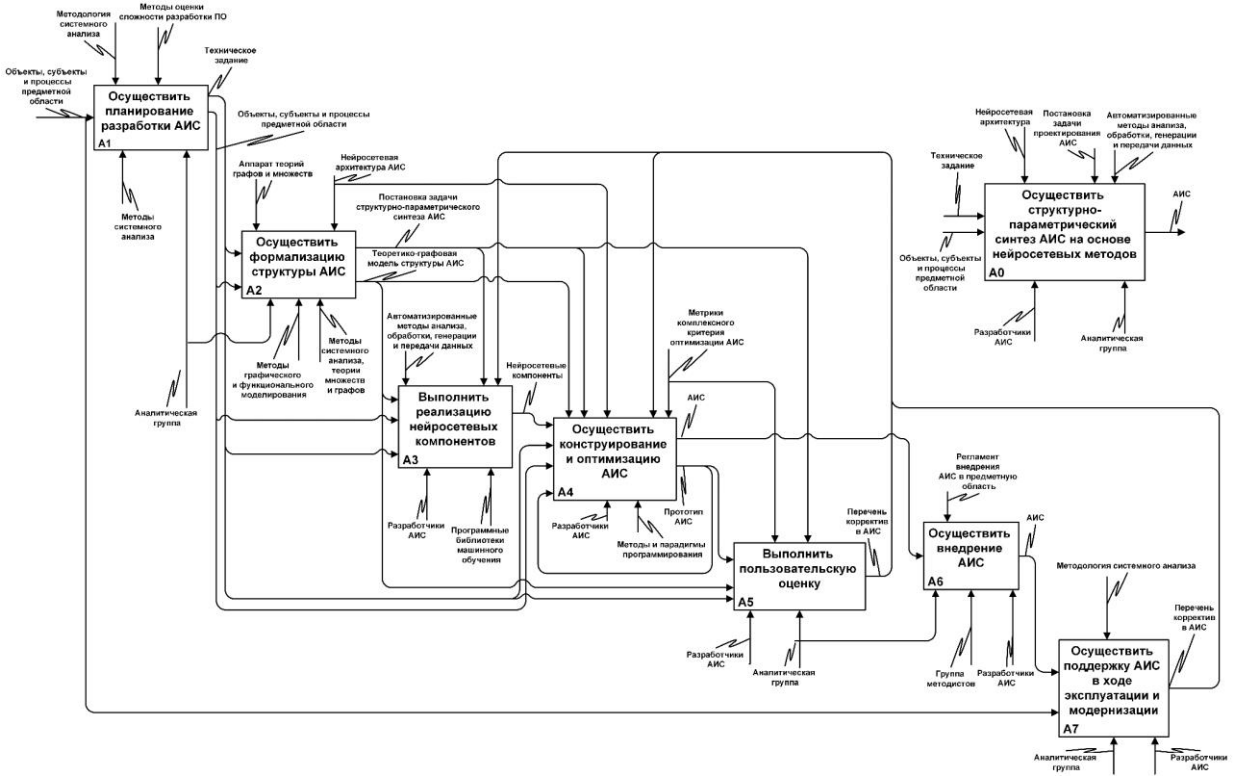


Рис. 3.7. Функциональная диаграмма методологии структурно-параметрического синтеза АИС

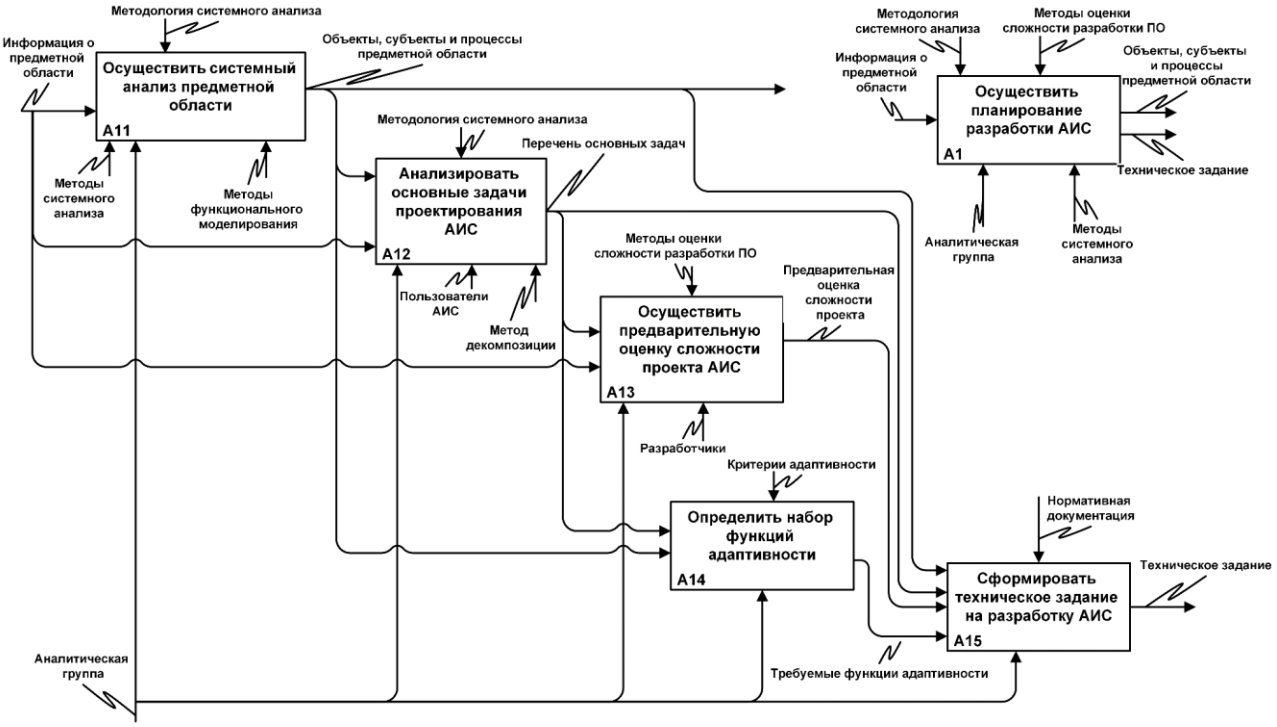


Рис. 3.8. Декомпозиция этапа Планирование

На первом шаге осуществляется системный анализ основных объектов, субъектов и процессов, принадлежащих предметной области (блок А11). Данная информация используется в дальнейшем на следующих этапах.

Группа разработчиков АИС, заказчик и будущие пользователи определяют основные задачи проекта путем декомпозиции предметной области и анализа объектов, субъектов и процессов, протекающих в ней (блок А12), после чего осуществляется предварительная оценка сложности проекта. Это позволяет выявить сложность и объем проекта, оценить стоимость и время его реализации, возможные технические требования (блок А13).

Далее выявляются необходимые функции адаптивности системы к различным внешним воздействиям. Примерами таких функций могут быть: вариативность интерфейса и навигации, возможность настройки оформления интерфейса системы, разграничение функциональности, кросс-платформенность и т.д. (блок А14).

В завершении этапа на основе собранной информации формируется техническое задание на разработку АИС (блок А15).

А2. Формализация. На втором этапе осуществляется однозначное и подробное описание архитектуры информационной системы с использованием различных математических аппаратов и методов системного анализа (рис. 3.9).

В блоке А21 группа аналитиков осуществляет системный анализ технического задания, полученного на предыдущем этапе. На основе нейросетевой архитектуры АИС и декомпозиции технического задания формируется перечень компонентов АИС.

Далее необходимо формализовать объекты и субъекты предметной области (блок А22). Для этого используется аппарат теорий множеств и графов, а также метод формализации информационных потоков на основе модели многоуровневых графов [50]. Для формализации процессов предметной области предлагается использование функционального моделирования – диаграмм в нотации IDEF0 (блок А23) либо блок-схем. Математическое обеспечение данных процессов излагается на этапе А22 в процессе формализации взаимодействия объектов и субъектов в виде математических формул и соотношений. Закономерности процессов обработки информации в предметной области могут быть оформлены в виде доказательства лемм, теорем и следствий.

Полученная теоретико-графовая модель объектов и диаграммы процессов предметной области интегрируются в компоненты АИС в соответствии с нейросетевой архитектурой (блок А24). Часть процессов при этом заменяется нейросетевыми каналами данных, соответ-

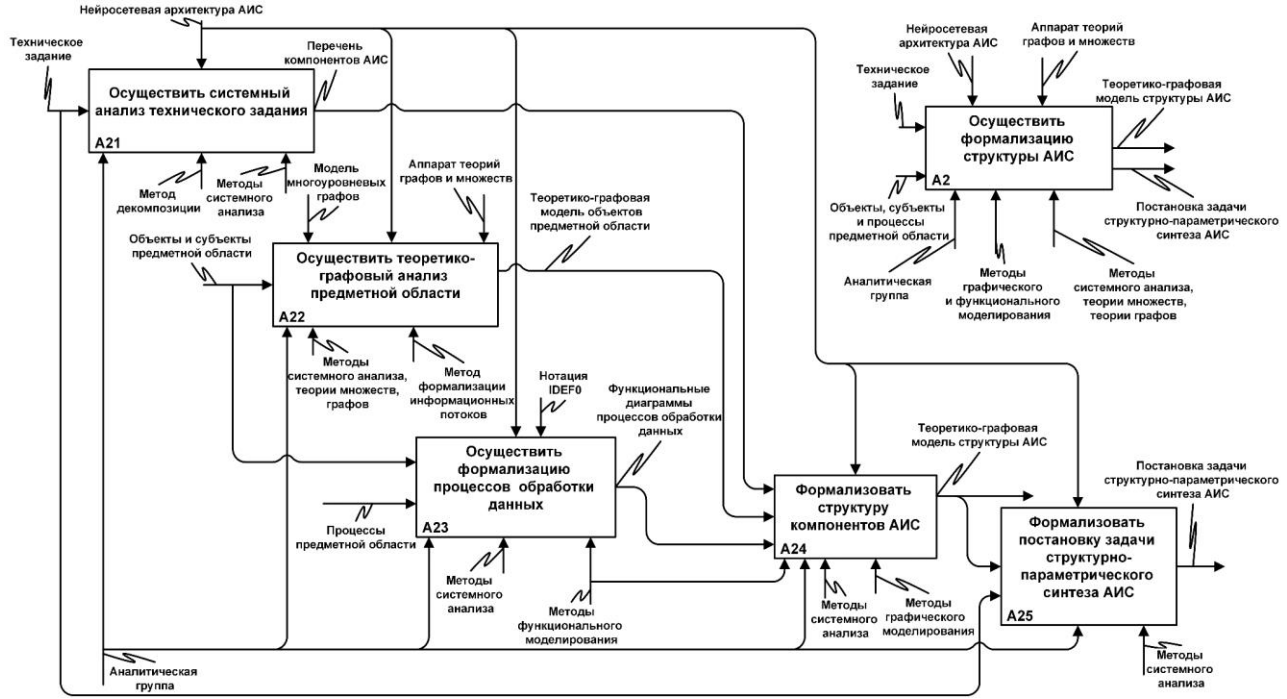


Рис. 3.9. Декомпозиция этапа Формализация

ствующими им по функциональности и типу. Объекты и субъекты распределяются по сущностям нейросетевой архитектуры (Окружению, Управлению, Модели и Представлению). Полученная формализованная схема элементов системы, математические и функциональные модели, описывающие объекты, субъекты и процессы предметной области, объединяются в единую теоретико-графовую модель структуры АИС.

На основе данной модели осуществляется формализованная постановка задачи структурно-параметрического синтеза АИС (блок А25). Основные требования и ограничения к системе, области определения параметров к этому этапу уже заложены в специфику компонентов АИС и теоретико-графовую модель структуры АИС. Решение данной задачи будет осуществлено на следующих этапах методологии.

А3. Реализация нейросетевых компонентов – этап анализа и обработки данных предметной области, формирования структуры и обучения нейронных сетей, используемых в ходе автоматизированного решения задач работы с информацией (рис. 3.10). Данный этап ранее не использовался в методологиях разработки информационных систем.

В блоке А31 на основе формализованной модели структуры АИС, свойств объектов, субъектов и процессов предметной области формируется перечень компонентов, основанных на нейросетевых технологиях. Их совокупность будет составлять сущность Нейронные сети в предлагаемой архитектуре АИС.

Определив набор необходимых компонентов, осуществляется анализ и предварительная обработка исходных данных (блок А32). Выборка данных осуществляется в соответствии с требованиями технического задания, т.е. каждой поставленной задаче ставится в соответствие необходимый набор данных для ее решения.

На следующем шаге (блок А33) подбирается оптимальная архитектура и параметры нейронных сетей для каждого нейросетевого компонента АИС. Эта задача может быть решена аналитически разработчиками или экспертами в области машинного обучения, а также с применением генетических алгоритмов и различных программных средств (например, AutoKeras, AutoGAN, Google AutoML и т.д. [130, 132, 134]).

Обученные нейронные сети оцениваются (блок А34). В случае, если определенная на данном шаге точность, качество или иные характеристики нейронных сетей признаны неудовлетворительными, то этап реализации нейросетевых компонентов начинается заново с блока А31. Если же полученные нейронные сети удовлетворяют пороговым значениям по выбранным метрикам, то осуществляется переход к этапу реализации нейросетевых компонентов.

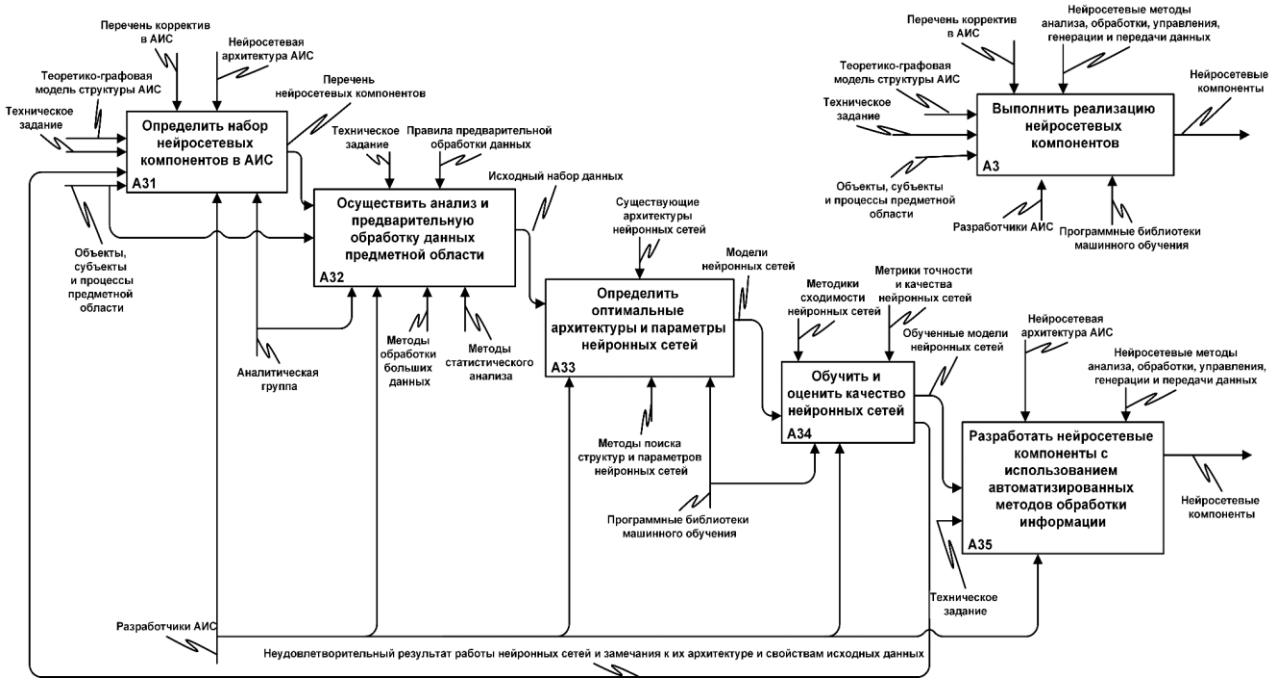


Рис. 3.10. Декомпозиция этапа реализации нейросетевых компонентов

Процесс разработки нейросетевых компонентов (блок А35) связан с применением перечисленных в рамках методологии нейросетевых методов обработки информации, функционирующих на основе технологий машинного обучения (рис. 3.11).

Блок А351. На данном этапе реализуются нейросетевые каналы данных различных типов, позволяющие автоматизировать процессы анализа, обработки и передачи данных. Структура, принцип работы и классификация нейросетевых каналов данных рассмотрены в разделе 3.3. Нейросетевая архитектура АИС, а также главе 5. Сформированные нейросетевые каналы также применяются на следующих этапах методологии при реализации других нейросетевых компонентов.

Блок А352. Для автоматизации решения задач анализа и классификации данных в АИС применяется нейросетевой метод классификации и распределения данных. В его основе использование нейронной сети – классификатора, распределяющей информацию и файлы по заданным категориям. Применение нейросетевого метода и нейросетевых каналов распределения данных позволяет реализовать нейросетевые компоненты для проверки введенных пользователем данных, перемещения их в правильные категории, автоматического распределения файлов, поиска дубликатов и схожих файлов (по заданным признакам).

Блок А353. Для организации процессов переадресации и маршрутизации данных в АИС реализуются нейросетевые компоненты, функционирующие на базе нейронных сетей, осуществляющих оценку загруженности пользователей, уровня их компетенций, сложности выполняемых операций. Решение данных задач осуществляется путем использования нейросетевых каналов данных и программной реализации нейросетевого метода автоматической переадресации данных.

Блок А354. Для адаптации модулей, интерфейса или иных компонентов АИС используются нейросетевые каналы адаптации данных, реализуемые в рамках нейросетевого метода адаптации информационных систем. Обработанные нейронной сетью данные о пользователях, характеристиках их программного и аппаратного обеспечения, внешних факторах используются для прогнозирования оптимальных настроек компонентов АИС.

Блок А355. Для подготовки тестовых данных, а также решения задач восстановления поврежденной информации, прогнозирования и оценки состояния объектов осуществляется программная реализация нейросетевого метода автоматической генерации данных. Это позволяет автоматизировать процессы создания новых информационных объектов и решение задач. Разработанные генераторы могут использоваться также в процессе разработки АИС для автоматического формирования ее компонентов (программного кода, элементов интерфейса, данных и т.д.).

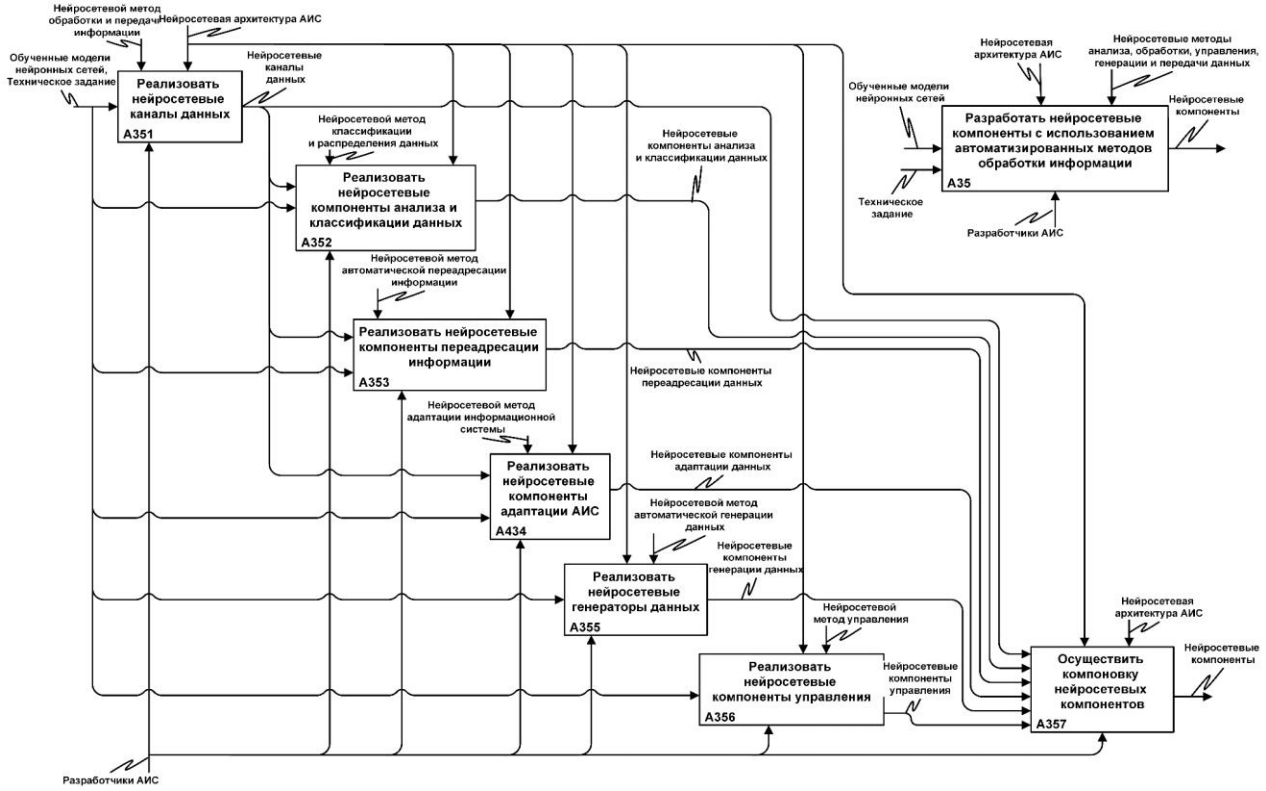


Рис. 3.11. Декомпозиция блока разработки нейросетевых компонентов

Блок А356. Для автоматизации процессов управления в АИС реализуются соответствующие нейросетевые компоненты. В зависимости от требований технического задания эти компоненты могут реализовывать частичное или полное нейросетевое управление, а также автоматизацию процессов поддержки принятия решений.

Блок А357. Завершающий этап, на которой разработчики осуществляют проверку работоспособности и тестирование полученных нейросетевых компонентов, корректность их совместной работы. Полученный блок компонентов отправляется на следующий этап.

Реализация нейросетевых компонентов также имеет итерационный характер и осуществляется до нахождения оптимального набора исходных данных и структур нейронных сетей. В случае возникновения новых задач и условий осуществляется реализация и модернизация соответствующих нейросетевых компонентов.

А4. Конструирование и оптимизация – основной этап разработки АИС (рис. 3.12).

На данном этапе разработчики интегрируют разработанные нейросетевые компоненты в АИС, а также вносят необходимые исправления в систему на основе замечаний, сформированных пользователями на следующем этапе. Таким образом, данный этап также повторяется многократно, на первой итерации формируется первичный прототип, на последующих осуществляется его доработка до получения финальной версии АИС.

На первой итерации разработки определяется область значения для всех варьируемых переменных (блок А41). Далее на основе поставленной задачи структурно-параметрического синтеза АИС, метрик комплексного критерия оптимизации АИС и корректив пользователей (отсутствующих на первой итерации) осуществляется решение задачи структурно-параметрического синтеза АИС (блок А42). Так как в решении задачи оптимизации применяется несколько метрик, входящих в состав комплексного критерия оптимизации АИС, необходимо использование одного из методов решения многокритериальных задач. После успешного решения задачи формируется структура и параметры АИС. Они анализируются и адаптируются под нейросетевую архитектуру АИС. Таким образом, в блоке А43 определяется структура модулей прототипа АИС.

Далее начинается непосредственно разработка АИС. Сначала оценивается готовность прототипа АИС к внедрению (блок А44). На первой итерации данный блок пропускается ввиду отсутствия программного обеспечения АИС.

Блок А45 включает все необходимые процессы по конструированию программных модулей АИС. На этом этапе разработчики

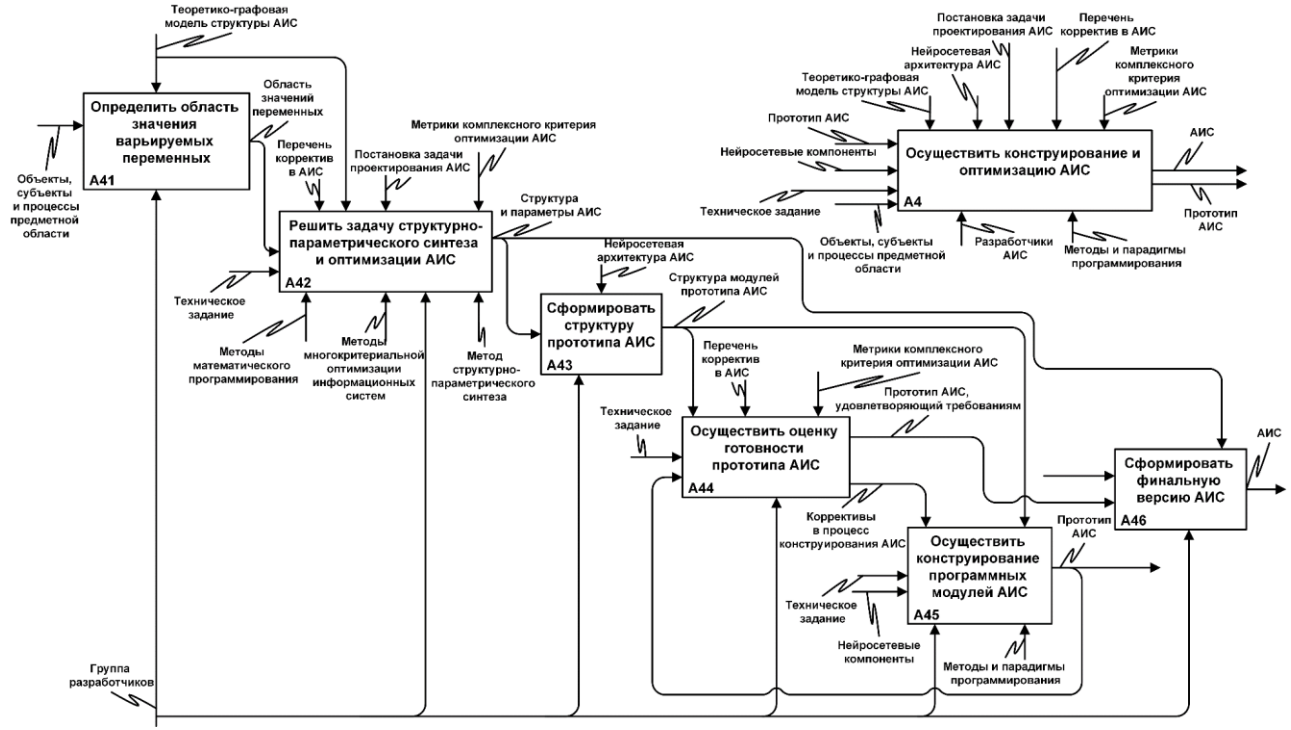


Рис. 3.12. Декомпозиция этапа Конструирование и оптимизация

осуществляют программирование модулей АИС, реализацию дизайна, а также интеграцию нейросетевых компонентов, полученных на предыдущем этапе, в прототип АИС. Разработка осуществляется в соответствии с существующими методами и парадигмами программирования в рамках разработанной ранее структуры модулей. Полученный на каждой итерации прототип АИС отправляется на анализ пользователям (в следующем этапе), а после оценивается на готовность в блоке А44.

Данная процедура разработки, оценки и внесения корректив пользователей продолжается многократно. Когда в блоке А44 текущая версия прототипа начинает удовлетворять требованиям технического задания в полном мере, осуществляется переход к блоку А46, где формируется финальная версия АИС.

А5. Пользовательская оценка. Этап реализует взаимодействие разработчиков и пользователей в целях получения оптимальной с точки зрения качества и удобства АИС (рис. 3.13). На данном этапе в аналитическую группу добавляются пользователи для тестирования и оценки прототипа АИС.

На первом шаге (блок А51) осуществляется проверка на соответствие текущей версии прототипа АИС требованиям технического задания, после чего проводится его оценка по количественным и качественным метрикам комплексного критерия оптимизации АИС (блок А52). Данная оценка позволяет контролировать ход выполнения проекта и в случае превышения пороговых значений или выхода за допустимую область внести необходимые коррективы.

В блоке А53 осуществляется пользовательская оценка прототипа АИС, а также его тестирование. Это позволяет выявить несоответствия между взглядами разработчиков и пользователей на проект, а также программные ошибки, неудобство в интерфейсе или сложность выполнения операций. Таким образом, прототип АИС субъективно оценивается на качество, быстродействие и степень адаптивности, что позволяет собрать и учесть индивидуальные особенности и пожелания пользователей.

Заключительный этап Пользовательской оценки – объединение собранной информации о несоответствии техническому заданию, выходу за предельные значения количественных и качественных метрик оценки АИС, замечаний пользователь (блок А54). Собранный перечень корректив в АИС передается на предыдущий этап для внесения изменений в решение задачи структурно-параметрического синтеза, реализацию нейросетевых компонентов, программных модулей, интерфейса и т.д.

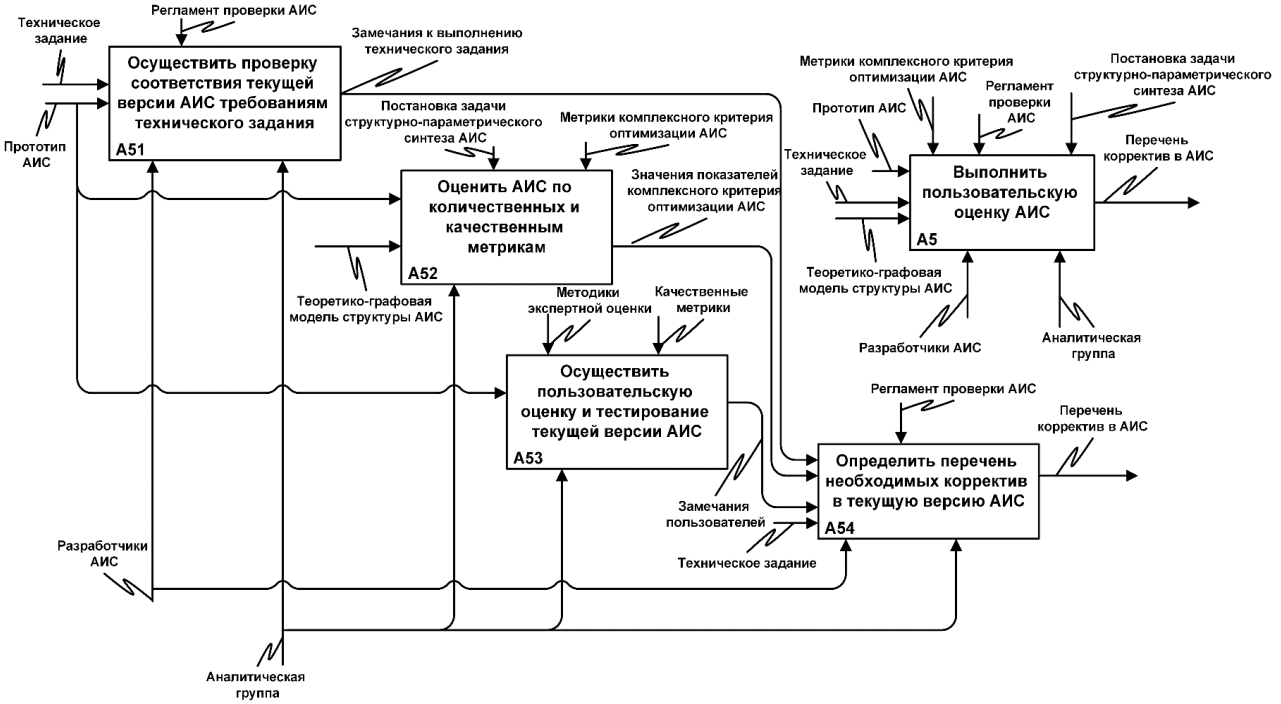


Рис. 3.13. Декомпозиция этапа Пользовательская оценка

Таким образом, пользователи задействованы в процессе разработки программного обеспечения и осуществляют его корректировку, что позволяет сразу устранить несоответствия в АИС. Данный этап достаточно длителен и повторяется многократно при появлении следующей итерации прототипа АИС.

Этапы А3 – А5 тесно взаимосвязаны и циклически следуют друг за другом (или проводятся параллельно), что позволяет получить законченное, непротиворечивое решение, удовлетворяющее требованиям заказчика, пользователей и технического задания. Применение разработанных нейросетевых методов для автоматизации процессов анализа, генерации, обработки и передачи данных позволяет снизить нагрузку на коллектив разработчиков и ускорить процесс реализации системы.

После получения приемлемой оценки от пользователей, завершения реализации всех нейросетевых компонентов, соответствия АИС метрикам комплексного критерия оптимизации и доказательства ее готовности осуществляется переход к завершающим этапам разработки АИС.

А6. Внедрение – заключительный этап реализации проекта, включающий финальное тестирование, внедрение информационной системы и обучение пользователей работе с ней (рис. 3.14).

После завершения работы над АИС необходимо осуществить финальное тестирование системы в реальных условиях той предметной области, в которой планируется ее использование (блок А61).

Обнаруженные технические ошибки и замечания от рядовых пользователей (на данном этапе к ним относятся все пользователи, взаимодействующие с АИС, а не отдельная группа, как на этапе А5) исправляются (этап А62).

Для прошедшей тестирование и работоспособной АИС подготавливается пакет сопроводительной документации под руководством группы методистов: руководства пользователя, руководство по эксплуатации, различные инструкции и справочные модули (блок А63). Методисты также организуют процесс обучения пользователей работе с АИС (блок А64). На данном этапе может возникнуть необходимость внесения небольших корректив, облегчающих взаимодействие пользователей с системой либо подготовка дополнительного справочного материала.

После завершения обучения пользователей АИС внедряется в предметную область в режиме штатного функционирования (блок А65). Начинается период ее эксплуатации.

Так как на предыдущих этапах пользователи активно участвовали в процессе разработки и тестирования системы, то этап А6 осуществляется достаточно быстро, проблемы в совместимости с техническим оборудованием и обучением персонала минимальны.

А7. Эксплуатация и модернизация – этап функционирования в штатном режиме и поддержки АИС (рис. 3.15).

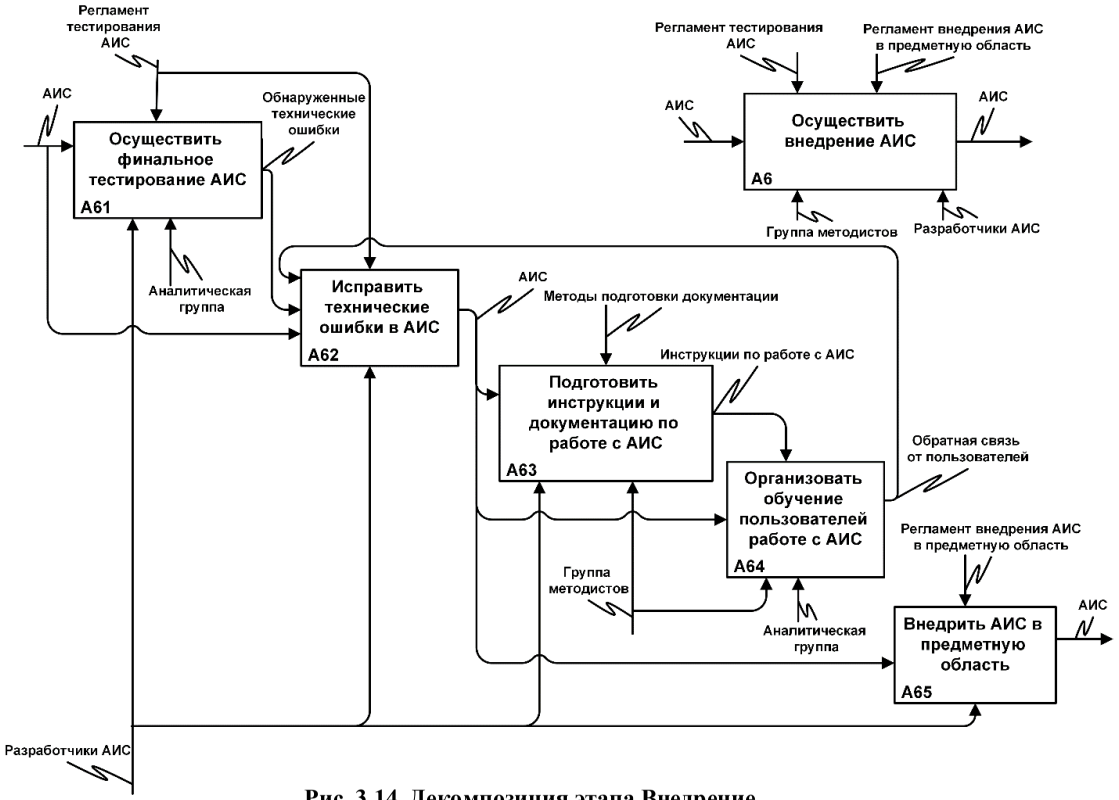


Рис. 3.14. Декомпозиция этапа Внедрение

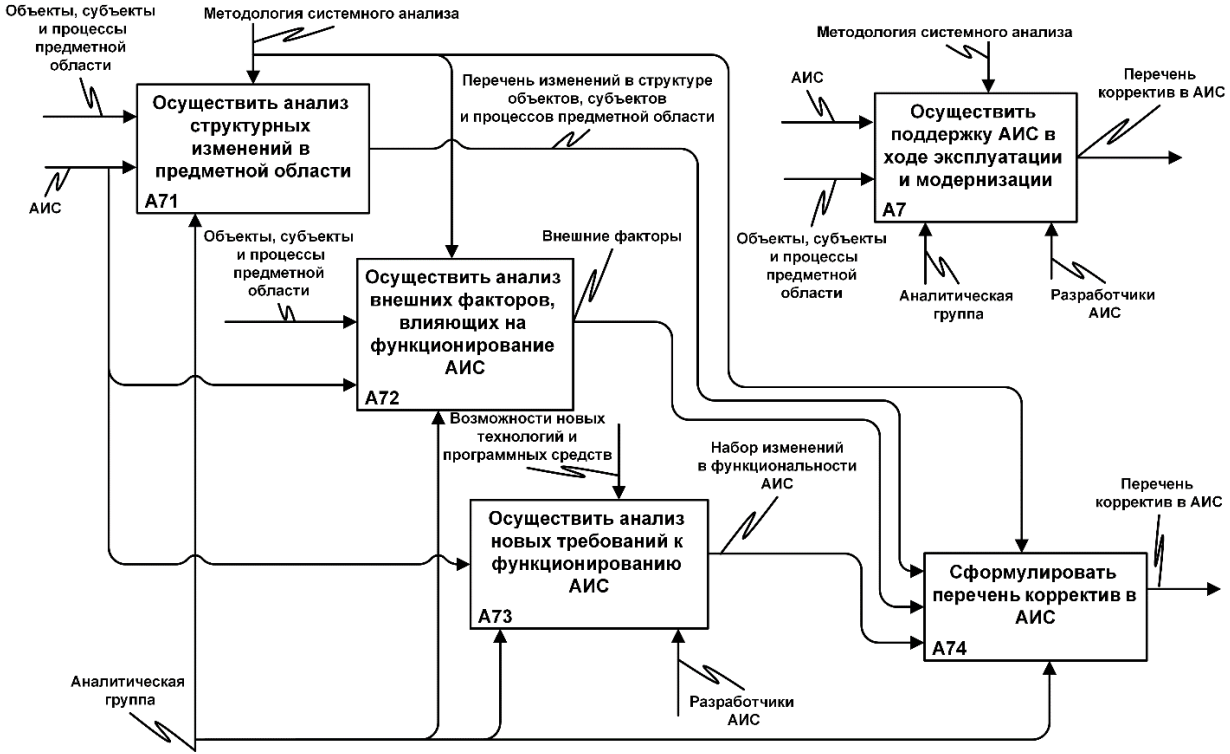


Рис. 3.15. Декомпозиция этапа Эксплуатация и модернизация

В ходе жизненного цикла АИС может изменяться под влиянием структурных изменений, внешних воздействий, появлением новых технологий, технических и программных средств.

Поэтому периодически осуществляется анализ изменений в структуре информационных объектов, субъектов, процессов в предметной области (блок А71). Также проводится мониторинг внешних факторов, влияющих на функционирование АИС: сбоев, технических изменений, нагрузки на систему и т.д. (блок А72).

Так как постоянно появляются и совершенствуются программные средства и инструменты, развиваются технологии (в том числе – машинного обучения), то через некоторое время часть компонентов может быть модернизирована в соответствии с возможностями новых инструментов (блок А73).

Таким образом, формируется список необходимых изменений и корректив в АИС (блок А74), передаваемый на этапы А3 и А4, на основе которого осуществляется доработка АИС по сокращенному сценарию, учитывая все полученные ранее результаты в виде моделей, методов, программных и аппаратных решений.

На этом декомпозиция этапов методологии структурно-параметрического синтеза АИС на основе нейросетевых методов завершена. Методология основана на существующих подходах в области разработки информационных систем (RAD), но ее научная новизна заключается в выделении в отдельный этап реализации нейросетевых компонентов и ориентации на использование нейросетевых технологий, методов и компонентов для решения задач автоматизации процессов анализа, обработки, генерации и передачи данных.

3.6. МЕТОД ФОРМАЛИЗАЦИИ ИНФОРМАЦИОННЫХ ПОТОКОВ НА ОСНОВЕ МОДЕЛЕЙ МНОГОУРОВНЕВЫХ ГРАФОВ

На втором этапе реализации методологии структурно-параметрического синтеза АИС основной задачей является формализация объектов, субъектов и процессов предметной области. От точности и полноты сформированной на этом этапе модели зависит соответствие разработанной АИС правилам и закономерностям предметной области, а допущенные ошибки приведут к значительным архитектурным коррективам в дальнейшем.

Проведенный анализ подходов по формализации и моделированию информационных систем позволил сделать вывод о применимости и достаточной эффективности аппарата теорий множеств и графов. Поэтому на основе данных инструментов предлагается реализация нового метода формализации информационных потоков, а также объ-

ектов и субъектов предметной области [50]. Метод заключается в формализации структуры информационных потоков, включающих основные информационные объекты, их жизненные циклы, процессы взаимодействия друг с другом и субъектами предметной области путем выполнения некоторых операций.

Представим систему информационных потоков в виде кортежа:

$$S = (U,\ P,\ O,\ RE,\ T,\ G), \tag{3.1}$$

где S – система (структура) информационных потоков предметной области, включающая множество информационных объектов, меняющих свои состояния в результате проведения операций множеством пользователей; $U = \{u_i \mid i = 1,\ ...,\ nU\}$ – множество информационных объектов, nU – общее количество объектов; $P = \{p_q \mid q = 1,\ ...,\ nP\}$ – множество пользователей, nP – общее количество пользователей; $O = \{o_l \mid l = 1,\ ...,\ nO\}$ – множество операций, выполняемых над объектами, nO – общее их количество; RE – множество воздействий на объекты, как внешних, так и внутренних; T – множество дискретных моментов времени; G – множество графов информационных потоков.

Каждому информационному объекту u_i соответствует некоторое множество состояний C_i, каждое из которых определяет содержимое и атрибуты объекта в определенный промежуток его жизненного цикла:

$$u_i \to C_i, \tag{3.2}$$

где $C_i = \{c_{ij} \mid j = 1,\ ...,\ nC_i\}$ – множество состояний объекта u_i, nC_i – количество таких состояний.

Для обозначения состояния c_{ij} объекта u_i, используемого при анализе или обработке данных в конкретных операциях, примем обозначение: $u_i(c_{ij})$.

Каждое состояние определяется как кортеж из множества атрибутов объекта и их значений в заданный промежуток времени TC_{ij}:

$$c_{ij} = (\{(a_{in},\ d_{ijn}) \mid a_{in} \in A_i,\ d_{ijn} \in D_{ij},\ n = 1,\ ...,\ N_i\},\ TC_{ij}), \tag{3.3}$$

где $A_i = \{a_{in} \mid n = 1,\ ...,\ nA_i\}$ – множество атрибутов объекта u_i с соответствующими им множеством значений атрибутов $D_i = \{D_{ij} \mid j = 1,\ ...,\ nC_i\}$ для каждого состояния c_{ij}:

$$D_{ij} = \{d_{ijn} \mid n = 1,\ ...,\ nD_{ij}\}, \tag{3.4}$$

где nA_i – количество атрибутов объекта; $TC_{ij} = \{t_m\}$ – множество моментов времени, в которые существует состояние c_{ij}.

Воздействия разделяются на внешние EE и внутренние IE: $RE = EE \cup IE$.

Внешние воздействия $EE = \{ee_w \mid w = 1, ..., W_e\}$ включают в себя распоряжения, заявления, заказы от сторонних организаций и прочие воздействия, осуществляемые извне. Таким образом, это информационные объекты внешних предметных областей и систем.

Внутренние воздействия $IE = \{ie_w \mid w = 1, ..., W_i\}$ формируются на основе внешних либо самостоятельно внутри предметной области. Воздействие направлено на получение конкретного результата – информационного объекта или некоторого их множества, которые удовлетворяют условиям, поставленным воздействием, что в общем виде можно представить следующим образом:

$$re_w = (U^*, P^*, O^*, T^*), \tag{3.5}$$

где U^*, P^*, O^*, T^* – множество объектов, пользователей, операций и временных ограничений, заданных воздействием re_w.

Структуру жизненного цикла информационного объекта (т.е. переход объекта из одного состояния в другое) опишем графическим способом с помощью ориентированных графов. Во-первых, это позволяет проследить весь жизненный цикл объекта, во-вторых, отобразить в удобной и понятной форме осуществляемые над объектом воздействия, и, наконец, такая форма наглядно показывает структуру информационных потоков в целом, позволяя выделить излишне перегруженные действиями участки [144, 145].

Однако в процессе формализации предметной области интерес представляет не только жизненный цикл отдельных объектов, но и процессы, протекающие в предметной области на более высоких уровнях масштабирования. Поэтому предлагается использовать следующую модель многоуровневых графов для их формализации.

Первый уровень масштабирования – уровень обработки состояния объекта, отражающий процессы перемещения информации в рамках одного состояния при выполнении конкретной операции (рис. 3.16, *а*). В формализованном виде граф имеет следующий вид: $G_1 = \{G_{1ij}\}$, $G_{1ij}(c_{ij}, O^1)$.

Второй уровень масштабирования – уровень обработки жизненного цикла объекта, отражающий процессы преобразования информации при выполнении ряда операций, начиная с его создания и заканчи-

вая уничтожением в рамках АИС (рис. 3.16, *b*). В формализованном виде принимает следующий вид: $G_2 = \{G_{2i}\},\ G_{2i} = (C_i,\ O^2)$.

Третий уровень масштабирования – уровень обработки информационного потока, отражающий процессы создания, движения и уничтожения множества объектов под влиянием некоторого множества воздействий RE, в результате которых заданным набором пользователей формируется необходимое подмножество объектов требуемого типа (рис. 3.16, *c*). В формализованном виде принимает следующий вид: $G_3 = \{G_{3w}\},\ G_{3w} = (U,\ RE)$.

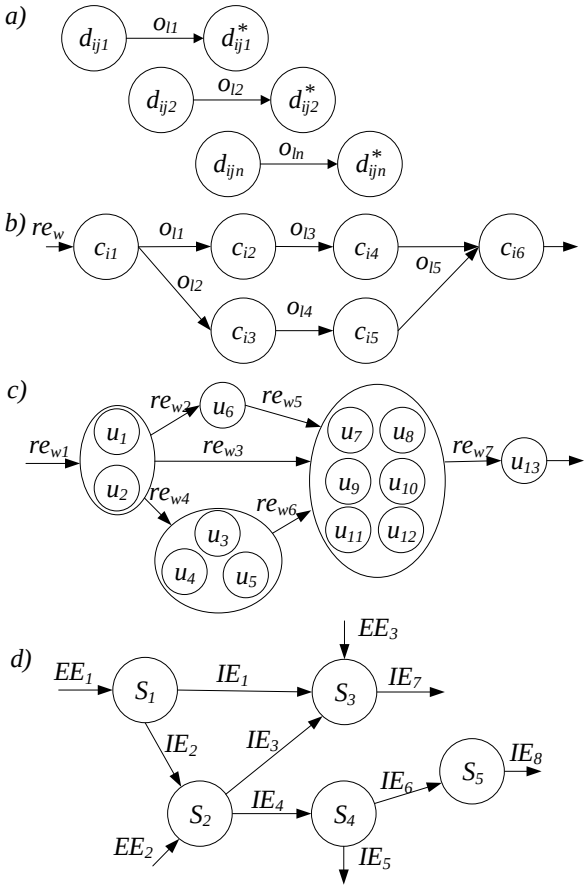


Рис. 3.16. Модель многоуровневых графов обработки информации в АИС

Четвертый уровень масштабирования – уровень обработки данных в информационном пространстве предметной области, отражающий процессы формирования, перенаправления, оптимизации информационных потоков, организации внутренних и внешних связей между ними под влиянием некоторого множества воздействий RE, в результате которого руководством организации формируется требуемая структура информационных потоков (рис. 3.16, *d*). В формализованном виде такой граф можно представить следующим образом: $G_4 = \{G_{4k}\},\ G_{4k} = (S_k,\ RE)$.

Таким образом, становится возможным формализовать процессы движения и обработки информации на разных уровнях, начиная с самого низкого и заканчивая высшим – уровнем информационного пространства предметной области. Представленная модель многоуровневых графов отличается от классического представления графов тем, что каждой вершине графа *k*-го уровня соответствует граф $(k - 1)$-го уровня. Обозначим эти связи в общем виде следующим образом:

$$\begin{gathered} G_k = (GV_k,\ GE_k), \\ GV_k \to G_{k-1}, \end{gathered} \tag{3.6}$$

где G_k – граф *k*-го уровня; GE_k – ребро графа *k*-го уровня; GV_k – вершина графа *k*-го уровня.

Далее рассмотрим формализованные представления типовых операций над информационными объектами. Данное представление может дополнять теоретико-графовую модель предметной области АИС и использоваться для формализации процессов обработки информации, протекающих в предметной области.

Сформулируем общий вид операций o_l следующим образом:

$$\begin{gathered} o_l : (U_{in},\ P_{in},\ t_l,\ re_w) \to (U_{out},\ P_{out},\ t_l + \Delta t_l), \\ \Delta t_l \le t_l{*},\ t_l{*} \in T{*},\ re_w = (U{*},\ P{*},\ O{*},\ T{*}), \end{gathered} \tag{3.7}$$

где U_{in}, U_{out} – подмножества информационных объектов на входе операции и на выходе соответственно; P_{in}, P_{out} – множества пользователей отправителей (инициаторов операции) и получателей (исполнителей операции); $t_l, \Delta t_l, t_l{*}$ – момент времени начала выполнения операции, общее время осуществления операции, максимальное время завершения операции соответственно; re_w – воздействие (внешнее или внутреннее), в соответствии с которым осуществляется операция.

Представленное выше соотношение отражает общий вид операций над информационными объектами, однако при создании АИС необходима реализация и, соответственно, формализация на этапе

проектирования конкретных операций обработки информации. Их также можно классифицировать в зависимости от того уровня масштабирования обработки информации, к которому они относятся.

На первом уровне осуществляется работа непосредственно с информацией в рамках одного состояния объекта, поэтому к данному классу операций $O^1 \in O$ будут относиться операции чтения o_r^1 и редактирования o_w^1 данных.

Ко второму уровню операций обработки данных относятся операции $O^2 \in O$, работающие уже с отдельными состояниями информационных объектов и формирующие их жизненный цикл, начиная от создания и заканчивая удалением из АИС. К таким операциям относятся:

o_a^2 – операция создания объекта;

o_d^2 – операция сжатия состояний;

o_w^2 – операция сохранения изменений объекта в новое состояние;

o_u^2 – операция объединения состояний.

Третий уровень обработки данных включает операции $O^3 \in O$, воздействующие уже на объекты и их объединения в рамках выполнения заданий. Примерами таких операций являются:

o_c^3 – операция копирования объекта;

o_d^3 – операция удаления;

o_p^3 – операция перемещения;

o_{rd}^3 – операция переадресации;

o_m^3 – операция движения объекта в рамках информационного потока.

На четвертом уровне обработки данных операции $O^4 \in O$ осуществляют изменение не конкретных объектов, а управляют информационными потоками организации в целом. На этом уровне абстракции не выделяются отдельные объекты, рассматриваются только некоторые их множества в рамках информационных потоков структурных подразделений [146]. Такими операциями являются:

o_{awf}^4 – формирование нового информационного потока в виде некоторой подсистемы информационных потоков S_k;

o_{pwf}^4 – перенаправление информационного потока, приводящее к изменению структуры управляющих воздействий RE_k в рамках системы S_k;

o^4_{ewf} – расширение информационного потока, приводящее к добавлению новых воздействий на информационный поток;

o^4_{uwf} – сжатие информационного потока, приводящее к сокращению количества воздействий на информационный поток;

o^4_{twf} – передача данных между информационными потоками.

Таким образом, представленный метод формализации информационных потоков на основе модели многоуровневых графов позволяет осуществить описание объектов, субъектов и процессов предметной области. Научной новизной данного метода является использование модели многоуровневых графов, что позволяет формализовать процессы движения и обработки информации на разных уровнях, начиная со структуры информационного объекта и заканчивая информационным пространством всей предметной области. Разработанная модель позволяет проанализировать процесс обработки и перемещения информации в рамках отдельных состояний, информационных объектов, информационных потоков и предметной области в целом, что позволяет использовать ее в АИС для различных сфер деятельности.

3.7. ВЫВОДЫ

В главе 3 рассмотрена методология структурно-параметрического синтеза АИС на основе нейросетевых методов. На основе проведенного в предыдущих главах анализа существующих методологий, архитектур построения информационных систем, а также методов обработки данных сформулирована и формализована задача структурно-параметрического синтеза АИС. Определены основные термины и понятия методологии.

Разработанная методология основана на выполнении следующих принципов:

– принцип нейросетевой организации структуры АИС определяет, что структура АИС представлена в виде множества изолированных и независимых сущностей, сгруппированных по 5 категориям: Окружение, Нейронные сети, Модель, Представление и Управление. Каждый элемент АИС для взаимодействия с другими компонентами и передачи информации использует нейросетевые каналы данных;

– принцип подготовки и соответствия информации состоит в том, что для реализации нейросетевых методов анализа, обработки, генерации и передачи, функционирующих на основе нейронных сетей, в АИС необходимо обеспечить предварительную обработку, корректность, достаточность и соответствие исходных данных для последующего обучения нейронных сетей и алгоритмов искусственного интеллекта;

– принцип применимости нейросетевой архитектуры заключается в возможности использования методов и моделей, основанных на подходах теории искусственного интеллекта, и формирует перечень требований к достаточности исходных данных, формализации процессов обработки информации, возможности применения аппроксимирующих функций и необходимости реализации функций адаптивности в выбранной предметной области.

В соответствии с этими принципами разработана нейросетевая архитектура АИС, основанная на шаблоне проектирования MVC, однако отличающаяся подходом к организации межмодульной связи и обработке данных с помощью автоматизированных нейросетевых компонентов. Рассмотрена декомпозиция нейросетевой архитектуры, принципы работы нейросетевых каналов данных – программных компонентов, используемых для автоматизации процессов анализа, обработки и передачи информации.

Сформулирована общая структура методологии и перечень методов, необходимых для ее реализации. Среди них выделен набор автоматизированных методов обработки информации, функционирующих на основе нейронных сетей. Представлена декомпозиция основных этапов методологии и ее формализация в нотации диаграмм IDEF0.

Представленная методология основывается на существующей методологии RAD. Научная новизна методологии заключается в использовании нейросетевой архитектуры и добавлении этапа реализации нейросетевых компонентов на основе новых нейросетевых методов. Методология направлена на решение следующей научной проблемы: повышение эффективности разработки АИС, выраженное в снижении сложности, стоимости процесса реализации программного обеспечения, повышении адаптивности системы.

В рамках методологии разработан и подробно рассмотрен метод формализации информационных потоков на основе моделей многоуровневых графов, используемых на первых этапах методологии в целях формирования четкого описания объектов, субъектов и процессов их взаимодействия на разных уровнях детализации, начиная со структуры информационного объекта и заканчивая информационным пространством всей предметной области.

Таким образом, в рамках данной главы подробно рассмотрена задача исследования, ее формализация, основные принципы и термины методологии, нейросетевая архитектура АИС, структура и этапы методологии структурно-параметрического синтеза АИС, а также метод формализации информационных потоков. В следующей главе будут подробно изложены автоматизированные методы взаимодействия с информацией, функционирующие на основе нейронных сетей.

Глава 4
РАЗРАБОТКА НЕЙРОСЕТЕВЫХ МЕТОДОВ УПРАВЛЕНИЯ, АНАЛИЗА, ОБРАБОТКИ, ГЕНЕРАЦИИ И ПЕРЕДАЧИ ИНФОРМАЦИИ

Реализация нейросетевых компонентов АИС, позволяющих автоматизировать процессы управления, анализа, обработки, генерации и передачи информации за счет технологий машинного обучения, основана на применении набора нейросетевых методов. В данной главе приводится формализация, теоретическое обоснование и описание разработанных нейросетевых методов, позволяющих снизить сложность программной реализации компонентов АИС и процессов работы с информацией.

4.1. НЕЙРОСЕТЕВОЙ МЕТОД ОБРАБОТКИ И ПЕРЕДАЧИ ИНФОРМАЦИИ

Рассмотрим решение задачи организации межмодульного взаимодействия в АИС на основе автоматизированных программных компонентов – нейросетевых каналов данных – в рамках нейросетевого метода обработки и передачи информации.

Сформулируем следующую задачу: необходимо осуществить передачу данных *X*, имеющих структуру *SX*, из *i*-го модуля АИС в *j*-й с возможностью осуществления преобразования $X \to Y$, где *Y* – ожидаемый в *j*-м модуле набор данных со структурой *SY*. Для решения поставленной задачи предлагается нейросетевой метод обработки и передачи данных, основанный на применении нейронных сетей для анализа, обработки информации и выбора способов ее передачи, отличающийся комплексным теоретическим обоснованием реализации и применением нейросетевых каналов данных для автоматизированной организации межмодульного взаимодействия.

Нейросетевой метод обработки и передачи данных заключается в получении по сетевому протоколу СР выходного вектора Y со структурой данных SY на основе преобразования входного вектора X со структурой данных SX. Принцип работы метода основан на применении нейросетевых каналов данных различной степени сложности. Дадим основные определения.

Определение 1. *Нейросетевой канал данных* (*Neural Network Data Channel, NNDC*) *– программный интерфейс, осуществляющий обработку и передачу данных между модулями АИС и включающий следующие элементы*: *входные и выходные данные, их структуры, искусственные*

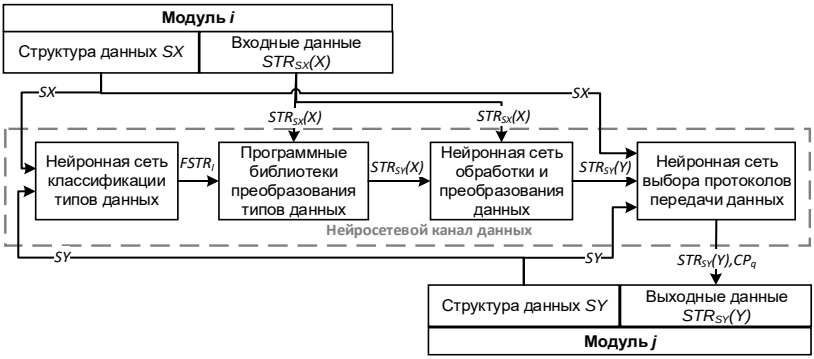


Рис. 4.1. Схематическая структура нейросетевого канала данных

нейронные сети для анализа и преобразования данных, выбора протоколов передачи данных, программные библиотеки преобразования и передачи данных.

Графически структуру нейросетевого канала данных можно представить в виде рис. 4.1.

Определение 2. *Под степенью сложности (NNS) нейросетевого канала данных будем понимать наличие нейронных сетей для выполнения соответствующих операций анализа (NNS = 1), обработки (NNS = 2) и передачи информации (NNS = 3). Допускается, что степень сложности может быть нулевой.*

Определение 3. *Категории (type) нейросетевых каналов данных определяют их функциональность, т.е. перечень задач, решаемых в рамках каждой конкретной категории.*

В общем виде процесс обработки и передачи данных в NNDC можно формализовать следующим соотношением:

$$NNDC_{i,j}^{type}(STR_{SX}(X),\ SY) = STR_{SY}(Y), \qquad (4.1)$$

где $NNDC_{i,j}^{type}$ – нейросетевой канал данных категории *type*, осуществляющий взаимодействие между *i*-м и *j*-м модулями АИС; $X = \{x_n \mid n = 1,\ ...,\ xN\}$ – данные размерностью *xN* со структурой данных SX, поступающие на вход NNDC от *i*-го модуля; $Y = \{y_m \mid m = 1,\ ...,\ yM\}$ – данные размерностью *yM* со структурой данных SY, отправляемые на выход NNDC и ожидаемые *j*-м модулем.

Примем обозначение $STR_{SX}(X)$ как выполнение операции приведения данных X к структуре вида SX. Аналогично $STR_{SY}(Y)$ – приведение данных Y к структуре вида SY.

Рассмотрим реализацию нейросетевого метода обработки и передачи информации, которая включает следующие этапы:

1. Формирование общей структуры нейросетевого канала данных:

$$NNDC = TR(CP_q) \cup a_1 NN^{ct} \cup a_2 NN^{proc} \cup a_3 NN^{tr}, \qquad (4.2)$$

где TR – процедура передачи данных по выбранному сетевому протоколу $CP_q \in CP$ (CP – множество всех сетевых протоколов); NN^{ct} – нейронная сеть классификации типов данных; NN^{proc} – нейронная сеть обработки и преобразования данных; NN^{tr} – нейронная сеть выбора протоколов передачи данных; a_i – коэффициенты, определяющие степень сложности нейросетевого канала данных, $i = 1, 2, 3$.

2. Анализ структур входных и выходных данных

Нейросетевой канал должен обеспечить соответствие между структурами входных и выходных данных. Для выполнения этого условия необходимо определить структуры SX и SY входных и выходных данных соответственно, используя нейронную сеть классификации типов данных NN^{ct}:

$$NN^{ct}(X) = SX, \; NN^{ct}(Y) = SY, \qquad (4.3)$$

3. Определение категории и степени сложности нейросетевого канала данных

Канал $NNDC_{i,j}^{type}$ устанавливает связь между модулями i и j. Если структуры и значения входных и выходных данных совпадают, а сетевой протокол CP_q заранее определен, то связь вырождается в нейросетевой канал данных нулевой степени. В противном случае требуется повышение степени нейросетевого канала для решения конкретных задач анализа, обработки и передачи данных.

На основе анализа отображения $X \to Y$ и $SX \to SY$ определяется категория нейросетевого канала, определяющая перечень задач по обработке и передаче данных. Данный процесс можно осуществить аналитически или автоматически с помощью различных систем поддержки принятия решений (в том числе на базе нейронных сетей).

4. Преобразование структур данных

Если $SX \neq SY$, то необходимо выполнить преобразование структур данных. В рамках данного метода предлагается реализация программной библиотеки преобразования структур. Далее на основе ре-

зультата функции сравнения структур данных C_{STR} выбирается функция преобразования $FSTR_l$:

$$C_{STR}(SX, SY) \to FSTR_l. \tag{4.4}$$

$$FSTR_l : STR_{SX}(X) \to STR_{SY}(X). \tag{4.5}$$

5. Обработка данных

Если $X \neq Y$, то необходимо осуществить обработку данных. Для этого используется нейронная сеть $NN^{proc}_{i,j,type}$:

$$NN^{proc}_{i,j,type} : STR_{SY}(X) \to STR_{SY}(Y). \tag{4.6}$$

Процесс обработки данных зависит от категории нейросетевого канала. Примерами такой обработки может быть регрессия, классификация, кластеризация, восстановление потерянных данных, генерация новой информации по заданным условиям и т.д.

6. Передача данных

Для автоматизации выбора сетевого протокола передачи данных CP_q необходимо использование нейронной сети $NN^{tr}_{i,j,type}$:

$$NN^{tr}_{i,j,type} : (SX, SY) \to CP_q. \tag{4.7}$$

Для нейросетевых каналов данных второй степени сложности или ниже сетевой протокол данных зафиксирован или выбирается аналитически. Выбранный протокол CP_q используется для передачи данных в процедуре TR.

Таким образом, изложенный нейросетевой метод обработки и передачи данных реализует весь цикл анализа, преобразования и передачи информации в АИС. Использование нейронных сетей на всех этапах, требующих привлечения человека для осуществления анализа и принятия решений, позволяет автоматизировать процедуру обработки и передачи информации, снизить негативное влияние человеческого фактора на процесс организации взаимодействия между компонентами АИС.

Рассмотрим ряд теорем, доказательство которых позволяет теоретически обосновать возможность реализации и применения нейросетевого метода обработки и передачи данных.

Теорема 1. *Нейросетевой канал данных произвольной степени сложности, в том числе нулевой, для любых i и j компонентов АИС позволяет осуществить процедуру передачи данных TR между этими*

компонентами при условии, что существует сетевой протокол CP_q, *программная реализация которого реализует передачу данных между i и j компонентами.*

Доказательство. В соответствии с выражением (4.2) и определением 1 процедура передачи данных в нейросетевых каналах данных любой степени сложности (в том числе и нулевой) основана на применении существующих сетевых протоколов CP_q в рамках процедуры передачи данных TR.

Рассмотрим вариант, когда степень сложности $NNS = 0$. В этом случае $a_1 = a_2 = a_3 = 0$, следовательно, $NNDC = TR(CP_q)$. Так как TR реализует передачу данных произвольной размерности и типа между модулями по существующему сетевому протоколу CP_q, то нейросетевой канал нулевой степени сложности сводится к программной реализации данного сетевого протокола передачи данных.

Предположим, что степень сложности $NNS \neq 0$, следовательно, некоторое a_i (хотя бы одно) является ненулевым: $a_i \neq 0$. Тогда функциональность нейросетевого канала данных будет включать процедуру передачи данных и некоторые операции над данными, выполняемые с использованием нейронных сетей. Это не противоречит исходному утверждению. Аналогично для любого количества a_i. Доказательство теоремы 1 подтверждает эквивалентность нейросетевого канала данных произвольной степени сложности и программной реализации сетевого протокола при решении задач передачи данных.

Следствие 1. *Если существует процедура передачи данных TR между i и j компонентами АИС и она реализуется средствами некоторого сетевого протокола* CP_q, *то в рамках нейросетевой архитектуры такая процедура может быть выражена как нейросетевой канал данных нулевой степени сложности.*

Из теоремы 1 следует, что нейросетевой канал нулевой степени соответствует программной реализации процедуры передачи данных: $NNDC = TR$, значит, верно и обратное: $TR = NNDC$, если степень канала $NNS = 0$. Доказательство следствия 1 позволяет сделать вывод, что в частном случае возможно организовать взаимодействие модулей в нейросетевой архитектуре на основе только сетевых протоколов передачи данных, представляя их как нейросетевые каналы нулевой степени. Таким образом, нейросетевой канал нулевой степени и процедура передачи данных по сетевому протоколу в рамках нейросетевой архитектуры эквивалентны и взаимозаменяемы.

Определение 4. *Нейросетевой канал данных первой степени сложности осуществляет решение задачи классификации структур и типов входных и выходных данных, а также их преобразование для обеспечения соответствия типов данных при организации взаимодействия модулей АИС.*

Теорема 2. *Нейросетевой канал данных первой степени сложности (или выше) может поставить в соответствие произвольному вектору данных* $X = \{x_n\}$ *размерности* xN *вектор* $S = \{s_e \mid e = 1, \ldots, nS\}$ *типов данных. Тогда элемент* $s_q \in S$, $s_q = \max\limits_{e=1, \ldots, nS} \{s_e = NN^{ct}(X)\}$ *будет однозначно определять правильный тип данных* $SX \in S$ *с ошибкой не более* ε:

$$\left| s_q - SX \right| \le \varepsilon \tag{4.8}$$

при условии, что отображение $X \to S$ *является непрерывным и не является инъекцией, а для любого X существует только один* SX.

Доказательство. Так как соответствие между входными наборами данных X и их типами данных S является непрерывной функцией f_{STR} на множестве X, то процедура определения типа данных заключается в нахождении непрерывной функции $f_{STR}(X) = S$. Непрерывность отображения $X \to S$ следует из утверждения, что любому вектору X_d можно поставить в соответствие тип данных $s_d \in S$. Предположим, что произвольному вектору X_d не задан тип данных $s_d \in S$, тогда всегда можно поставить ему в соответствие универсальный тип данных $s_k \in S$. Отображение $X \to S$ не является инъекцией, так как равенство типов данных $f_{STR}(X_1) = s_1$, $f_{STR}(X_2) = s_2$ не означает равенства исходных данных: $X_1 \ne X_2$.

Из теоремы Цыбенко [115] следует, что двухслойная нейронная сеть может аппроксимировать любую непрерывную функцию многих переменных с ошибкой, не превышающей ε. Следовательно, нейронная сеть NN^{ct} может аппроксимировать функцию f_{STR} с некоторой ошибкой ε. Так как $NN^{ct}(X) = \{s_e \mid s_e \in S, e = 1, \ldots, nS\}$ (нейронная сеть формирует вектор S вероятности принадлежности к каждому типу данных s_e), то $s_q = \max\limits_{e=1, \ldots, nS} \{s_e\}$ будет единственным решением, наиболее близким к SX. Это обеспечивает выполнение нера-

венства (4.8). Таким образом, доказательство теоремы 2 гарантирует существование и единственность аргументов SX и SY выражения (4.4), а также возможность их определения выражением (4.3).

Определение 5. *Нейросетевой канал данных второй степени сложности включает функциональность канала первой степени и осуществляет решение задачи преобразования исходных данных к форме и содержимому, ожидаемых модулем АИС.*

Теорема 3. *Нейросетевой канал данных второй степени сложности (или выше) может осуществить обработку произвольного многомерного вектора данных* $X = \{x_n\}$ *размерности* xN *, поставив ему в соответствие вектор* $Y = \{y_m \mid y_m = NN_{i,j,type}^{proc}(X)_m\}$ *размерности* yM *с ошибкой не более* ε *относительно ожидаемого выхода* $Y^r = \{y_m^r\}$:

$$\left| \frac{1}{yM} \sum_{m=1}^{yM} (y_m - y_m^*)^2 \right| \le \varepsilon, \tag{4.9}$$

при условии, что отображение $X \to Y$ *является непрерывным, инъекцией (так как различные векторы* X *преобразуются в различные векторы Y) и сюръекцией (так как для* $\forall Y \exists X (NN_{i,j,type}^{proc}(X) = Y)$*, таким образом,* $X \to Y$*-биекция.*

Доказательство. Из второй теоремы Хехт-Нильсена [116] и следствия из теоремы Колмогорова–Арнольда [117] имеем, что для любого множества пар векторов произвольной размерности $\{(X,Y)\}$ существует трехслойная нейронная сеть с конечным числом нейронов, формирующая для каждого входного вектора X соответствующий ему выходной вектор Y. Тогда, если процесс отображения $X \to Y$ является некоторой непрерывной функцией на множестве X, то по теореме Цыбенко ее можно аппроксимировать нейронной сетью $NN_{i,j,type}^{proc}$ с ошибкой, не превышающей ε. Это обеспечивает выполнение неравенства (4.9).

В противном случае, если отображение $X \to Y$ является дискретным, т.е. заданным табличными значениями, то его можно заменить сплайном $S_n(X)$ некоторой степени n, являющимся непрерывным. $S_n(X)$ можно аппроксимировать нейронной сетью $NN_{i,j,type}^{proc}$ с ошибкой, не превышающей ε. Доказательство теоремы обеспечивает возможность выполнения преобразования (4.6), повторяемость результатов работы нейросетевого канала данных (так как $X \to Y$ – инъекция),

взаимно однозначное соответствие исходных и преобразованных векторов (так как $X \to Y$ – биекция).

Определение 6. *Нейросетевой канал данных третьей степени сложности} включает функциональность каналов первой и второй степеней и осуществляет решение задачи выбора сетевого протокола передачи данных.*

Теорема 4. *Нейросетевой канал данных третьей степени сложности может осуществить автоматический выбор сетевого протокола* $CP_q \in CP$ *передачи данных на основе анализа упорядоченной пары типов данных* (SX, SY) *при условии, что отображение* $\{(SX,\ SY)\} \to CP$ *(в том числе для случая* $SX = SY$*) непрерывно и не является инъекцией (ввиду универсальности сетевых протоколов и возможности их применения для различных сочетаний* $(SX,\ SY)$*).*

Доказательство. Аналогично доказательству теоремы 2. Соответствие между типами данных (SX, SY) и протоколами передачи данных CP_q является непрерывной функцией f_{TR} на множестве упорядоченных пар векторов $\{(SX,\ SY)\}$. Тогда по теореме Цыбенко аппроксимируем функцию $f_{TR}(SX,\ SY) = CP_q$ двухслойной нейронной сетью $NN^{tr}_{i,\,j,\,type}$. Доказательство теоремы 4 обеспечивает возможность выполнения преобразования (4.7) и автоматизации процедуры передачи данных за счет перехода от аналитического выбора сетевого протокола к автоматическому.

В ходе доказательства вышеперечисленных теорем и исследования закономерностей функционирования нейросетевых каналов данных определены следующие ограничения для использования нейросетевого метода (определены принципом подготовки и соответствия информации):

- набор данных для обучения нейронных сетей задан на области определения возможных значений входных и выходных данных;
- каждому входному вектору данных должен соответствовать выходной вектор, причем единственный;
- содержимое входных и выходных данных можно привести к некоторому численному значению.

Изложенные определения, теоремы и ограничения отражают основные закономерности и принципы функционирования нейросетевого метода и нейросетевых каналов данных.

Новизна нейросетевого метода обработки и передачи информации заключается в разработке теоретических основ применения нейросетевых каналов данных, формализованного математического и алгоритмического обеспечения, позволяющего осуществить их программную реализацию.

4.2. НЕЙРОСЕТЕВОЙ МЕТОД АВТОМАТИЧЕСКОЙ ГЕНЕРАЦИИ ДАННЫХ

Рассмотрим задачу генерации данных в АИС. Обозначим классы задач следующим образом $K = \{K1, ..., K4\}$, где:

К1. Генерация сжатого представления о состоянии объекта.

К2. Генерация подобных (псевдореальных) объектов.

К3. Генерация отсутствующих данных в задачах обработки информации.

К4. Прогнозирование данных в задачах управления и обработки информации.

Перечисленные задачи можно объединить в единое направление в рамках АИС – процедуру генерации данных. Тогда актуальной задачей является разработка универсального и автоматического методов генерации данных, что позволит повысить гибкость и надежность системы в нестандартных режимах работы, восстанавливать поврежденные данные, осуществлять анализ процессов и объектов, в том числе с возможностью прогнозирования.

Нейросетевой метод автоматической генерации данных заключается в получении элементов вектора выходных данных $Y = (y_1, ..., y_m)$ заданной размерности со структурой данных SY на основе обработки вектора входных данных $X = (x_1, ..., x_n)$ заданной размерности со структурой данных некоторой нейронной сетью NN_i, входящей во множество всех возможных архитектур нейронных сетей NN. Процесс генерации данных осуществляется в автоматическом режиме за счет анализа структур исходных данных и классификации конечной подзадачи генерации, что позволяет конкретизировать параметры подготовки данных, процесса построения нейронной сети, ее обучения и использования.

Определение 7. *Под понятием «структура данных» будем понимать класс данных, однозначно определяющий множество характеристик и значений каждого элемента класса, а также множество операций, которые могут быть к ним применены, и пусть* S *– множество всех классов данных. Тогда под структурой данных SA неко-*

торого вектора $A=(a_1, ..., a_t)$ *будем понимать образ множества* A *при отображении* $fs: A \to SA$, *где* $SA \in S$.

Теорема 5. *В нейросетевом методе автоматической генерации данных для любой пары* (SX, SY) *структур входных и выходных данных существует однозначно определенный класс задачи генерации данных* $Ki \in K$ *при условии, что отображение* $(SX, SY) \to K$ *непрерывно.*

Доказательство. На основе теоремы Хехт-Нильсена [116] и следствия из теоремы Колмогорова–Арнольда [117] имеем, что всегда существует трехслойная нейронная сеть, для любого множества пар $\{(A, B)\}$ осуществляющая отображение $A \to B$. Тогда, подставив $A=(SX, SY)$ и $B=Ki$, получим отображение $(SX, SY) \to Ki$. Данное отображение является непрерывным на множестве упорядоченных пар $\{(SX, SY)\}$, так как для каждой пары можно однозначно определить ее класс Ki. Предположим, что такого класса нет. Тогда множество классов можно расширить элементом $Kn \in K$, к которому будут относиться задачи с исходными данными (SX, SY). Возможность аппроксимации непрерывного отображения с заданной точностью подтверждается теоремой Цыбенко [115]. Однозначность класса задач гарантируется выбором максимального значения вероятности p принадлежности к K^* классу в выходном слое нейронной сети (например, функцией Softmax): $K^* = \max\limits_{i=1, ..., nK} (p((SX, SY) \mid Ki))$, $nK=4$. Доказательство теоремы 5 подтверждает возможность разбиения некоторого множества задач $Z=\{z_q\}$ генерации данных по подмножествам $\{Z_{Ki}\}_{Ki \in K}$, для каждого из которых можно применить соответствующий алгоритм решения задачи Ki-го класса.

Теорема 6. *В нейросетевом методе автоматической генерации данных каждому вектору входных данных* $X=(x_1, ..., x_n)$ *со структурой данных* SX *можно поставить в соответствие многомерный выходной вектор* $Y=(y_1, ..., y_m)$ *со структурой* SY *с отклонением от вектора требуемых значений* $Y^*=(y_1^*, ..., y_m^*)$ *не более, чем* ε:

$$\left| \frac{1}{m} \sum_{i=1}^{m} (y_i - y_i^*)^2 \right| \le \varepsilon \qquad (4.10)$$

при условии, что отображение $X \to Y$ *непрерывно и является биекцией.*

Доказательство. Убедимся, что отображение $X \to Y$ отвечает заданным условиям. Предположим, что оно является дискретным, т.е. заданным табличными значениями. Тогда его можно заменить сплайном некоторой степени *n*, являющимся непрерывным. Если отображение $X \to Y$ – биекция, то, во-первых, различные векторы *X* преобразуются в различные векторы *Y* (свойства инъекции), во-вторых, для каждого вектора *Y* существует соответствующий ему вектор *X* (свойства сюръекции). Действительно, оба эти условия выполняются, что следует из принципов работы нейронных сетей.

Тогда в соответствии с теоремами Хехт-Нильсена и Колмогорова–Арнольда [116, 117] получаем отображение $X \to Y$, реализуемое трехслойной нейронной сетью с конечным числом элементов для любой размерности векторов *X* и *Y*. По теореме Цыбенко данное отображение можно аппроксимировать нейронной сетью с ошибкой, не превышающей ε. Доказательство теоремы 6 подтверждает возможность осуществления операции генерации выходного вектора *Y* на основе входного вектора *X*.

Далее рассмотрим формализованное представление нейросетевого метода автоматической генерации данных, включающее следующие этапы.

1. Формализация вектора входных $X = (x_1, ..., x_n)$ и выходных $Y = (y_1, ..., y_m)$ данных, определение их структур (классов данных): SX и SY соответственно.

2. Анализ связей между входными и выходными данными, что позволит определить класс решаемой задачи. Анализ может осуществляться в автоматизированном режиме на основе методов машинного обучения. Тогда существует некоторая классифицирующая нейронная сеть, осуществляющая отображение структур в класс задачи: $(SX, SY) \to Ki$.

3. Подготовка и обработка исходных данных. На данном этапе осуществляется преобразование исходных данных *X* путем выполнения различных операций: нормализация, округление, переход от непрерывных значений к категориальным, использование разреженных матриц вместо дискретных значений, лемматизация, очистка исходных данных от лишней или служебной информации и т.д. Реализация перечисленных операций возможна в автоматическом режиме за счет набора библиотек программного кода, а их применение позволяет

ускорить работу нейронных сетей и повысить их точность. Каждый класс задач Ki реализует свою процедуру обработки данных.

4. В соответствии с выбранной задачей Ki в автоматическом режиме выбирается архитектура нейронной сети $NN^* \in NN$: итерационно осуществляется анализ точности нейронных сетей $NN_i \in NN$ на небольшой выборке данных $X_0 \in X$. Подмножество сетей $NN_a \subset NN$, показывающее наилучшую точность, используется дальше и модифицируется, подмножество $NN \setminus NN_a$ отбрасывается. Этап 4 повторяется до тех пор, пока не будет достигнута требуемая точность либо разрешенное время на выполнение этапа. Для каждого класса задач Ki определяется определенное подмножество возможных архитектур и параметров нейронных сетей, что позволяет сократить время поиска оптимальной архитектуры и исключить изначально неэффективные решения. Если структура входных данных соответствует изображениям (определяется на шаге 2 и 3), то используются сверточные слои, если многомерным векторам временных последовательностей – рекуррентные и т.д.

5. Нейронная сеть NN^* обучается на всем тренировочном наборе $X_{train} \in X$. Оценивается ее точность A. В случае превышения ошибки $A(NN^*)$ заданного порога ε осуществляется возврат к этапу 4. Способ вычисления ошибки (например, на основе среднеквадратичного отклонения, перекрестной энтропии или иных метрик) определен для каждой задачи.

6. Если обученная нейронная сеть обеспечивает требуемую точность $(A(NN^*) \le \varepsilon)$, то осуществляется контрольный замер на данных $X_{test} \in X$, $X_{test} \cap X_{train} = \varnothing$. Полученные средствами нейронной сети значения Y_{test} оцениваются на принадлежность области допустимых значений Y.

7. Нейронная сеть внедряется в программное обеспечение АИС, заменяя аналитические методы генерации данных.

Таким образом, разработанный нейросетевой метод автоматической генерации данных основан на использовании нейронных сетей при анализе исходных данных, классификации и решении задач генерации данных.

Рассмотрим алгоритмы решения поставленных задач, реализованные в соответствии с предлагаемым нейросетевым методом автоматической генерации данных.

K1. Генерация сжатого представления о состоянии объекта.

Пусть задан некоторый многомерный вектор $X = (x_1, ..., x_n)$, которому необходимо поставить в соответствие вектор $Y = (y_1, ..., y_m)$. Y будем называть сжатым представлением вектора *X*, следовательно, $m < n$.

На первом этапе осуществляется обработка исходных данных *X* путем преобразования их в массивы через нормализацию, перевод в лексемы, векторизацию и т.д. Далее формируется нейронная сеть, в состав которой включаются плотные слои, а также, в зависимости от структуры данных: сверточные (для изображений), рекуррентные (для текстовых данных), слои выравнивания (в случае, если размерность $X > Y$), слои забывания (для снижения вероятности переобучения).

В качестве архитектуры нейронной сети выбирается AutoEncoder (AE), данная сеть позволяет осуществить отображение $X \to X$ с ошибкой ε. Архитектура AE предполагает наличие скрытого слоя *H*, формируемого на выходе энкодера, и используемого на входе декодера для восстановления исходного вектора *X*. Таким образом, обучив AE на некотором выборе векторов *X*, в качестве сжатого состояния для каждого из этих векторов можно использовать $Y = H$.

При таком подходе вектор сжатого представления *Y* обладает меньшей или сравнимой размерностью и меньшей длиной *m*, определяемой архитектурой сети. В ходе обучения можно последовательно увеличивать длину m до тех пор, пока погрешность не станет меньше ε.

Возможна следующая модификация алгоритма: для каждого объекта заданы сжатые состояния, например в виде категорий или набора признаков. Тогда задача генерации сжатого состояния вырождается в задачу классификации. Для ее решения в качестве выхода AutoEncoder будем использовать заданный набор выходных значений $X*$, тогда нейронная сеть будет реализовывать отображение $X \to H \to X*$, в качестве сжатых состояний будет использоваться $Y = X*$. Причем за счет упрощения исходных данных в результате преобразования $X \to H$ точность классификации может быть улучшена относительно обычной многослойной нейронной сети.

K2. Генерация подобных (псевдореальных) объектов.

Пусть задан некоторый многомерный вектор $X = (x_1, ..., x_n)$, относящийся к классу объектов DC ($X \in DC$), которому необходимо поставить в соответствие многомерный вектор $Y = (y_1, ..., y_m)$,

причем $Y \in DC$. Таким образом, объект Y является подобным для объекта *X*.

Аналогично решению задачи класса *K1* осуществляем преобразование вектора *X* (методами нормализации, перевода в лексемы, векторизации и т.д.). Далее формируется нейронная сеть. Выходной слой зависит от структуры данных, например, для изображений это может быть трехмерный массив размера «ширина-высота-цветовые каналы». Скрытые слои выбираются аналогично задаче класса *K1*.

В качестве архитектуры нейронной сети выбирается генеративно-состязательная сеть (GAN), включающая две сети: генеративная сеть учится создавать псевдореальные объекты на основе некоторых векторов (чаще всего – случайных), а дискриминативная сеть используется для распознавания объектов, принадлежащих заданному классу DC (подлинные и поддельные). Обе сети обучаются параллельно на выборке, включающей как подлинные образцы объектов (*X*), так и полученные генеративной сетью (*Y*). Дискриминативная сеть определяет подлинность объекта *Y*, в случае сбалансированного и длительного обучения обоих сетей ошибка стремится к ε, а свойства сгенерированных объектов *Y* – к свойствам *X*.

Ввиду особенностей функционирования сетей типа GAN полученные объекты Y будут относиться к классу DC ($Y \in DC$), но конкретные значения y_n вектора *Y* будут уникальными в рамках класса DC. Возможность решения задач класса *K2* с заданной точностью ε обеспечивается доказательством теоремы 6, с учетом того, что структуры векторов *Y* и *X* совпадают.

Рассмотрим модификацию рассмотренного алгоритма для получения псевдоподобных объектов с заданными свойствами. Пусть задан многомерный вектор свойств $X_i = (x_1, ..., x_n) \in X$, характеризующий некоторый класс объектов $z_t \in Z$ (т.е. имеется отображение $X_i \to z_t$), которому необходимо поставить в соответствие многомерный вектор $Y_i = (y_1, ..., y_m)$, причем также существует отображение $Y_i \to z_t$. Таким образом, Y_i классифицируется как объект класса $z_t \in Z$.

В качестве архитектуры нейронной сети также выбирается генеративно-состязательная сеть (GAN), включающая две сети: генеративная учится создавать объекты Y_i на основе вектора свойств X_i, дискриминативная используется для распознавания класса объекта $z_i \in Z$. Если объект является поддельным, то он относится к нулевому классу ($z_0 \notin Z$). Все сети обучаются параллельно на выборке, включающей

как подлинные образцы объектов (Y_T), так и полученные генеративной сетью (Y). Задача обучения сводится к тому, чтобы научить генератор не только формировать подлинные объекты Y , но обеспечить их соответствие классам из множества Z. Тогда GAN-сеть считается обученной в том случае, когда $\forall X_i \in X$ верно, что $Y_i \in Y$, $Y_i \to z_t$, $X_i \to z_t$.

К3. Генерация отсутствующих данных в задачах обработки информации.

В многомерном векторе $X = (x_1, ..., x_n)$ необходимо найти множество отображений следующего вида:

$$f_r : \{x_i \mid x_i \in X_{F,r}\} \to \{x_j \mid x_j \in X_{U,r}\},$$

где $X_{F,r} \in X$ (подмножество известных элементов в r-м отображении) и $X_{U,r} \in X$ (подмножество неизвестных элементов в r-м отображении). Тогда имеем, что $X_{F,r} \cup X_{U,r} = X$, $X_{F,r} \cap X_{U,r} = \varnothing$. Существующие подходы основаны на определении элементов подмножества $X_{F,r}$ с помощью методов отбрасывания, замены на фиксированные или усредненные значения, K-ближайших соседей, нейронных сетей [128].

Множество отображений $F = \{f_r : r = 1, ..., R\}$ позволяет осуществить отображение любого $X_{F,r}$ в множество $X_{U,r}$, если его мощность $\left|X_{F,r}\right| \geq 1$. Максимальное количество таких отображений зависит от мощности вектора X:

$$K = \sum_{i=1}^{|X|-1} C_{|X|}^{i} = \frac{(|X|)!}{(i)!(|X|-i)!}. \qquad (4.11)$$

Тогда для решения поставленной задачи необходимо выбрать структуру и осуществить обучение множества нейронных сетей (ансамбля) $NN = \{NN_r : r = 1, ..., R\}$, где $\forall f_r$ существует отображение $(f_r \to NN_r)$. Тогда каждая r-я нейронная сеть по теореме 6 аппроксимирует r-е отображения f_r:

$$NN_r : \{x_i \mid x_i \in X_{F,r}^{i}\} \to \{x_j \mid x_j \in X_{U,r}^{i}\}. \qquad (4.12)$$

Далее в соответствии с предлагаемым методом осуществляется автоматический подбор структуры и параметров каждой сети (или всего множества сетей одновременно). В качестве архитектуры сетей предлагается полносвязанная многослойная модель с минимум тремя слоями (следует из теоремы 6).

Процедура обучения в данной задаче отличается необходимостью предварительной обработки данных. Для каждого вектора $X = (x_1, ..., x_n)$ осуществляется формирование тренировочного набора из входных векторов $X_{tr,\,r} = \{(x_i : x_i \in X_{F,\,r}) : r = 1, ..., R\}$ и соответствующих им выходных векторов $Y_{tr,\,r} = \{(x_i : x_i \in X_{U,\,r}) : r = 1, ..., R\}$.

Применение ансамбля для восстановления отсутствующих данных основано на анализе тестового входного вектора $X_{test} = (x_1, ..., x_n)$ для определения корректных значений $\{x_i : x_i \in X_F\}$. Анализ может проводиться путем фильтрации исходного вектора и присвоения отсутствующим значениям отличительных значений. Далее вектор X_F обрабатывается нейронной сетью NN_r (только одной из всего множества NN), что позволяет определить вектор $\{x_i : x_i \in X_U\}$.

К4. Прогнозирование данных в задачах управления и обработки информации.

Пусть задан некоторый многомерный вектор $X_i = (x_1, ..., x_n)$, $X_i \in X$, определяющий состояние некоторого объекта в некоторый момент времени $t_i \in T$, т.е. имеет место отображение $t_i \to X_i$. Пусть существует такой момент времени $t_0 \in T$, что не существует $t_i \le t_0$. Обозначим t_0 как начальный момент времени отслеживания объекта, а X_0 – как начальное состояние объекта ($X_0 \in X$, $t_0 \to X_0$). Множество упорядоченных пар $H = \{(t_i, X_i) : i = 0, ..., h\}$ обозначим как историю состояний объекта вплоть до момента t_h.

Тогда в рамках данной задачи необходимо найти множество упорядоченных пар $H_q = \{(t_i, X_i) : i = h+1, ..., h+q\}$, которое обозначим как прогноз состояния объекта на q шагов ($H_q \subset H$).

Сначала рассмотрим возможность прогнозирования одного состояния объекта X_{h+1} в момент времени t_{h+1} на основе данных о предыдущих состояниях $H_p = \{(t_i, X_i) : i = p, ..., h\}, H_p \subset H$. Для подготовки тренировочного набора будем использовать для каждого выходного вектора $Y_{tr,\,j} = X_j$ входной вектор $X_{tr,\,j} = (X_{j-p}, X_{j-p+1}, ..., X_{j-p+p-1})$, соответствующий p предшествующим состояниям объекта.

Так как $X_{tr,j}$ является упорядоченным множеством и порядок состояний нем определяет будущие состояния объекта, будем использовать в нейронной сети слои типа LSTM (Long short-term memory), реализующие долгосрочную память и позволяющие отслеживать временные зависимости.

Если мощность множества $\left|H_q\right| > 1$, т.е. требуется прогнозировать более одного состояния вплоть до момента времени t_{h+q}, т.е. $\left|H_q\right| = q$, то сформулируем процесс обучения нейронной сети следующим образом: для каждого выходного вектора $Y_{tr,j} = (X_{j+1}, X_{j+2}, ..., X_{j+q})$ задан входной вектор $X_{tr,j} = (X_{j-p}, X_{j-p+1}, ..., X_{j-p+p-1})$, соответствующий p предшествующим состояниям относительно X_j. Таким образом, в общем виде в процессе решения задачи на вход нейронной сети подается p состояний, на выходе формируется q следующих состояний.

Необходимо отметить, что при решении задачи *K2* необходимо осуществить оценку качеству произвольной GAN. Существующие подходы не позволяют реализовать такую оценку, поэтому разработан метод оценки произвольных GAN.

4.2.1. Метод оценки качества произвольных GAN на основе модифицированных метрик Inception Score и Fréchet Inception Distance

В соответствии с проведенным анализом для GAN существует две наиболее распространенных метрики: Inception Score (IS) и Fréchet Inception Distance (FID).

Анализ показал, что существенным ограничением применения метрик IS и FID для оценки GAN является их ориентирование на анализ изображений. Для черно-белых и цветных изображений произвольного размера применение IS и FID возможно за счет преобразования исходных данных. Однако подобные преобразования вносят искажения в структуру данных при изменении количества цветовых каналов и масштабировании.

Для объектов, представленных векторами целых или вещественных значений произвольной размерности, использование IS и FID невозможно. Нейронная сеть Inception V3, применяемая для расчета IS и FID, не поддерживает входные данные произвольных форматов и ориентирована на анализ изображений размером от 75×75

до 299×299 пикселей. Даже ее модификация с добавлением дополнительных слоев на входе не позволит получить корректный результат: в структуре Inception V3 присутствует большое число сверточных слоев Convolution, MaxPooling и других, которые не обнаружат в исходных данных тех признаков, что можно выявить на изображениях. Следовательно, такая сеть не сможет сформировать корректный выходной вектор признаков. Классификация данных с использованием такой сети также не будет оптимальной.

Тогда предлагается следующий метод оценки качества произвольных GAN, отличающийся использованием для вычисления IS и FID вместо Inception V3 нейронных сетей (декодера и классификатора) с заданными параметрами и архитектурой. Формализуем основные этапы метода.

Пусть задан вектор исходных данных $X = (x_1, ..., x_n)$ произвольной размерности XN, для которого необходимо получить выходной вектор $Y = (y_1, ..., y_n)$ такой же размерности. Для решения этой задачи требуется выбрать структуру и обучить GAN NN_{GAN}, осуществляющую отображение $X \to Y$, причем, если исходные данные принадлежат некоторому классу ($X \in Z_1$), то и выходные данные также входят в него ($Y \in Z_1$). Для получения оптимального результата решения задачи необходимо добиться такой структуры нейронной сети, при которой метрики ее качества достигают наилучших значений:

$$\begin{aligned} IS(NN_{GAN}) &\to \max, \\ FID(NN_{GAN}) &\to \min, \\ LOSS(NN_{GAN}) &\to \min, \end{aligned} \tag{4.13}$$

где IS – расчет метрики IS для произвольной сети NN_{GAN}; FID – расчет метрики FID для произвольной сети NN_{GAN}; $LOSS$ – расчет функции потерь для произвольной сети NN_{GAN}, определяемый как бинарная энтропия.

Внесем следующие модификации, которые позволят применять оценки IS и FID для произвольных исходных данных.

1. Переход от сети Inception V3 к произвольному классификатору для вычисления IS. Возможны два варианта:

– Для реальных данных существуют метки классов, всего *N* классов. Тогда обучается нейронная сеть NN_C, осуществляющая класси-

фикацию через отображение $X \to \{y_k \mid k = 1, \ ..., \ N\}$. Тип сети зависит от исходных данных: для изображений используются последовательность сверточных слоев, для произвольных векторов с численными значениями – плотные многослойные сети, для классификации временных рядов или лексем – рекуррентные сети.

– Для реальных данных не существует меток классов, либо представлены данные единственного класса ($N = 1$). Тогда задается произвольное число категорий (кластеров) $N > 1$, и обучается алгоритм кластеризации NN_C на основе K-средних, осуществляющий отображение $X \to \{y_k \mid k = 1, \ ..., \ N\}$.

2. Переход от сети Inception V3 к произвольному автоэнкодеру NN_{AE} для вычисления FID при определении значений векторов признаков a_1 и a_2 для реальных и сгенерированных данных соответственно. Автоэнкодер NN_{AE} осуществляет следующее преобразование: $X \to HV \to X$, где HV – скрытый слой между декодером и энкодером длиной H. Тогда векторы a_1 и a_2 будут заменены на соответствующие реальным и сгенерированным данным вектора HV_1 и HV_2:

$$mu_i = \frac{\sum_{j=1}^{H} HV_i}{H}, \ C_i = \mathrm{cov}(HV_i), \ i = 1, \ 2. \qquad (4.14)$$

Указанные модификации в рамках рассмотренного метода оценки качества GAN позволят применять метрики IS и FID для произвольных наборов данных независимо от их структуры и типа. Это позволяет оценивать используемые GAN при реализации нейросетевого метода автоматической генерации данных.

Таким образом, научная новизна представленного нейросетевого метода автоматической генерации данных в АИС заключается в использовании нейронных сетей при анализе исходных данных, классификации и решении задач генерации данных, реализации алгоритмов решения задач генерации данных, автоматическом подходе к определению класса задач генерации данных, выбору архитектуры и обучению нейронных сетей, адаптации и реализации оценок IS и FID для произвольных структур данных. Предложенный метод теоретически обоснован, для каждой задачи (*К1 – К4*) предложен алгоритм решения.

4.3. НЕЙРОСЕТЕВОЙ МЕТОД АВТОМАТИЧЕСКОЙ ПЕРЕАДРЕСАЦИИ ИНФОРМАЦИИ

Одной из важных задач при проектировании и функционировании АИС является организация процесса переадресации информации между исполнителями. Особенно остро этот вопрос встает в условиях кризиса и удаленной работы сотрудников [147, 148], когда организация может столкнуться с необходимостью оперативного перенаправления информационных потоков от одного исполнителя к другому. Например, в случае отсутствия ответственного исполнителя из-за болезни или высокой загруженности необходимо оперативно переключить выполнение ряда операций на другого пользователя, обладающего необходимым набором компетенций. Под процессом переадресации будет пониматься изменение маршрутов движения информационных потоков в виде перераспределения операций над некоторыми информационными объектами между пользователями в АИС. Целью переадресации является повышение производительности работы организации.

Для автоматизации данного процесса и снижения сложности его реализации в АИС предлагается нейросетевой метод автоматической переадресации, функционирующий на основе технологий машинного обучения. Рассмотрим его формализованное представление.

Пусть в АИС задано множество информационных объектов $X = \{x_k\}$, включающих документы, файлы, численные и текстовые данные и т.д. С АИС взаимодействует множество исполнителей $U = \{u_i\}$, выполняющих множество операций $O = \{o_l\}$ над информационными объектами в течение множества временных промежутков $T = \{t_w\}$.

Матрицей приоритетов операций

$$R = \{r_{lm} \mid r_{lm} = [-1;\ 1],\ l,\ m = 1,\ ...,\ Q\}$$

обозначим бинарное отношение порядка на множестве O: $R \subseteq O \times O$, определяющее приоритет $r_{lm} \in R$ операции $o_l \in O$ над операцией $o_m \in O$. В общем виде матрица приоритетов операций имеет вид

$$R = \begin{bmatrix} 0 & r_{12} & ... & r_{1Q} \\ r_{21} & 0 & ... & ... \\ ... & ... & 0 & ... \\ r_{Q1} & r_{Q2} & ... & 0 \end{bmatrix}, \qquad (4.15)$$

где $Q = |O|$ – общее количество операций.

Матрица R обладает следующими свойствами:

1) Если $l = m$, то $r_{lm} = 0$;

2) $\forall l \le Q,\ m \le Q\ (r_{lm} = -r_{ml})$;

3) если $l \ne m$ и операция $o_l \in O$ имеет больший приоритет над операцией $o_m \in O$, то $r_{lm} = 1$, иначе $r_{lm} = -1$.

Операция $o_l \in O$ для каждого информационного объекта $x_k \in X$ осуществляет следующее преобразование:

$$o_l : (x_k,\ u_i) \to (x_h,\ t_w), \tag{4.16}$$

где $x_h \in X$ – результат выполнения операции o_l исполнителем $u_i \in U$ за время $t_w \in T$ в виде нового или измененного информационного объекта x_h.

Для каждого исполнителя u_i можно сформулировать очередь задач $Z_i \in Z$, под которой будем понимать упорядоченное множество операций в порядке приоритета их выполнения:

$$\begin{gathered} u_i \to Z_i,\ Z_i = (z_{ij} \mid j = 1,\ ...,\ nZ_i), \\ z_{ij} \to o_l(x_k,\ u_i), \end{gathered} \tag{4.17}$$

Порядок операций в очереди определяется следующим образом: $\forall z_{ij} \in Z_i$, $\forall z_{ij+1} \in Z_i$, где $z_{ij} \to o_m(x_k,\ u_i)$, $z_{ij+1} \to o_l(x_n,\ u_i)$ выполняется условие $r_{lm} = 1$.

На основе проведенного анализа и формализации предметной области поставим задачу оптимизации процесса переадресации: необходимо минимизировать суммарное время выполнения задач Z в АИС за счет автоматизации процесса переадресации операций O над информационными объектами X между исполнителями U. Для решения задачи предлагается нейросетевой метод автоматической переадресации. Формализуем его этапы в нотации теории множеств.

В соотношении (4.17) информационный объект x_k задан, а исполнитель u_i операции o_l должен быть выбран таким образом, чтобы время операции t_w стремилось к минимуму. При выборе u_i необходимо обеспечить возможность успешного выполнения операции o_l и учитывать текущую загруженность Z_i исполнителя.

Для осуществления этого выбора в рамках метода предлагается следующая последовательность действий. Для сопоставления испол-

нителей и операций, которые они могут успешно выполнить, формируется матрица компетенций исполнителей, являющаяся бинарным отношением на множествах U и O: $C \subseteq U \times O$. Тогда, если $\exists u_i, \exists o_l$, то значение элемента матрицы $c_{il} \in C$ формируется на основе анализа выполненных и невыполненных пользователем операций. Одним из вариантов вычисления значения элементов матрицы C может быть статистический анализ успешности выполнения операций. С другой стороны, данный процесс можно автоматизировать с применением нейросетевых технологий без необходимости разработки аналитических подходов для расчета элементов матрицы C.

Пусть задана многослойная нейронная сеть NN_C, принимающая на вход сведения о исполнителе u_i и операции o_l, прогнозирующая на выходе вероятность успешного выполнения операции f_{il} за время t_w. Таким образом, NN_C осуществляет отображение

$$NN_C : (U,\ O) \to (F,\ T), \tag{4.18}$$

где $F = \{f_{il}\}$ – множество вероятностей успешного выполнения операций исполнителями, элементы которого принимают значения от 0 до 1.

Возможность реализации отображения (4.18) подтверждается фундаментальными исследованиями в области нейронных сетей, изложенных в работах [116, 117].

Тогда в процессе функционирования АИС осуществляется обучение сети NN_C на основе собранных данных $(U,\ O,\ F,\ T)$. Обученная на достаточном объеме данных сеть сможет прогнозировать вероятность F и время выполнения T произвольных операций O для различных пользователей U. За счет аппроксимирующих свойств нейронных сетей станет возможным получение информации о тех сочетаниях «исполнитель u_i – операция o_l», что не использовались в процессе обучения, за счет анализа опыта исполнителя u_i при выполнении других операций $o_m (m \neq l)$, так и оценки сложности выполнения операции o_l другими пользователями $u_j (j \neq i)$. Таким образом, скрытые слои нейронной сети NN_C являются программной реализацией матрицы компетенций исполнителей.

Обученная сеть NN_C в процессе функционирования АИС используется для определения подмножества исполнителей $U_C \in U$,

которые могут выполнить текущую операцию с высокой вероятностью $f_{il} \to 1$ и наименьшим временем $t_w \to 0$.

Следующим этапом реализации метода является ранжирование элементов множества $U_C \in U$ на основе общей загруженности исполнителей Z. В ходе ранжирования выбирается исполнитель $u_a \in U_C$ с наибольшей вероятностью успеха f_{al} и наименьшим временем t_w выполнения операции o_l с учетом текущей загруженности Z_a. При расчете времени выполнения операции учитываются приоритеты операции и положение текущей операции o_l в очереди задач Z_i для каждого потенциального исполнителя $u_i \in U_C$.

Пусть дано:

$$Z_i = \{z_{i1},\ z_{i2},\ ...,\ z_{in}\},\ z_{is} \to o_l\ . \tag{4.19}$$

Тогда для каждого $u_i \in U_C$ проверяется место s операции o_l в очереди Z_i:

$$\begin{aligned} &\{z_{i1},\ ...,\ z_{in}\} \cup \{z_{is}\} \to \{z_{i1},\ ...,\ z_{is-1},\ z_{is},\ z_{is+1},\ ...,\ z_{in+1}\}, \\ &s = \min(\{j \mid j \in [1;\ n],\ j \to o_m,\ r_{ml} \le 0\})\,. \end{aligned} \tag{4.20}$$

Таким образом, на основе матрицы R приоритет операции o_l в очереди Z_i определяется порядковым номером s, являющимся минимальным индексом j задачи z_{ij}, у которой приоритет r_{ml} операции o_m меньше, чем у o_l.

Вычислив положение s операции o_l в каждой очереди Z_i, можно определить время ее выполнения:

$$Z_i^s = \{z_{i1},\ ...,\ z_{is-1},\ z_{is}\},\ Z_i^s \subseteq Z_i, \tag{4.21}$$

$$T_{is} = \sum_{j}^{s} t_j,\ t_j \in T,\ z_{ij} \to t_j\ . \tag{4.22}$$

Тогда в качестве исполнителя u_a операции o_l выбирается тот, у которого время выполнения операции минимальное:

$$u_a = \{u_i \mid T_{is} = \min(T_{is})\}. \tag{4.23}$$

Рассмотренный нейросетевой метод автоматической переадресации отличается теоретико-множественным анализом процесса переад-

ресации информационных объектов в АИС, применением нейронных сетей для определения оптимального исполнителя операции, оценки времени и вероятности ее успешного выполнения, реализацией матриц приоритетов операций и компетенций исполнителей для ранжирования операций и сопоставления их с уровнем подготовки исполнителей.

Для дальнейшей программной реализации нейросетевого метода автоматической переадресации и его интеграции в АИС разработан алгоритм.

На этапе подготовки осуществляется формализация основных объектов предметной области: исполнителей и операций. Для операций экспертной группой формируется матрица приоритетов. Разработчики проектируют соответствующую структуру базы данных с необходимым набором таблиц. Так же на этом этапе разрабатывается программный модуль, позволяющий осуществить сбор информации о проведенных исполнителями операциях, времени их выполнения и факте успешного завершения.

На следующем этапе проводится сбор необходимой информации. АИС работает в штатном режиме. Однако каждая операция, назначенная исполнителю, фиксируется и записывается в базу данных.

Этап обучения нейронной сети начинается, когда объем собранных данных набирает достаточный объем (минимум несколько записей о каждой операции для каждого исполнителя). Используя язык программирования Python и библиотеку машинного обучения Keras, разработчик проектирует структуру нейронной сети. Обученная нейронная сеть с найденными весовыми коэффициентами сохраняется.

Этап интеграции нейронной сети в АИС включает программную реализацию сервиса (например, на основе фреймворка Flask (Python)). В рамках данного сервиса осуществляется загрузка модели обученной нейронной сети, прием сообщений от АИС с информацией о новых операциях и поиск подмножества оптимальных исполнителей $U_C \in U$. Для каждого элемента подмножества U_C осуществляется анализ загруженности исполнителей, расчет времени выполнения операции по формуле (4.22). На основе полученных значений выбирается исполнитель, способный выполнить операцию за наименьшее время. Его идентификатор отправляется в качестве ответа в АИС.

Заключительный этап реализует модификацию и актуализацию нейронной сети. Каждая назначенная исполнителям операция, время и успешность ее выполнения записываются в базу данных. Собранный набор используется для обучения нейронной сети. Таким образом,

можно улучшить качество, повысить гибкость, учитывать изменения в структуре организации (уход и появление новых исполнителей, расширение типов операций).

Сформулированный нейросетевой метод автоматической переадресации основан на теоретико-множественном анализе процесса переадресации и отличается применением нейронных сетей для определения оптимального исполнителя операции, времени и вероятности ее успешного выполнения. В ходе реализации метода сформулированы такие понятия, как матрица приоритетов операций и матрица компетенций исполнителей для ранжирования операций и сопоставления их с уровнем подготовки исполнителей.

4.4. НЕЙРОСЕТЕВОЙ МЕТОД КЛАССИФИКАЦИИ И РАСПРЕДЕЛЕНИЯ ДАННЫХ

АИС в процессе функционирования обрабатывает большие объемы информации, поступающие от пользователей. Из-за человеческого фактора всегда существует некоторая вероятность ошибки. Например, пользователь может перепутать поля для ввода данных, забыть прикрепить файл либо перепутать категории файлов, случайно добавить несколько файлов одного типа. Часть из этих ошибок АИС может обнаруживать на основе аналитических алгоритмов, однако в ряде случаев без применения технологий искусственного интеллекта разрешить такие задачи невозможно [149].

Таким образом, на основе проведенного ранее анализа можно выделить актуальные задачи по классификации и распределению данных в АИС:

– соответствие введенных данных (файлов) той категории, в которой их расположил пользователь [110];

– определение для данных (файлов) их категорий в случае ошибки: перемещение их в правильные категории либо выдача пользователю предупреждения [111, 112];

– проверка дублирования данных (файлов) [113];

– автоматическое распределение файлов по категориям.

Для решения поставленных задач предлагается разработать метод, обобщающий существующие подходы к классификации информации и основанный на применении нейронных сетей для автоматизации обработки данных и повышения гибкости работы АИС.

Первая стадия метода заключается в подготовке данных для обучения и тестирования нейронной сети. Сеть будет использоваться для классификации данных по заданным категориям.

Пусть задан набор информационных объектов $X = \{x_1, ..., x_N\}$, который может быть представлен текстовой или численной информацией, введенной пользователем в поля форм в АИС, файлами, загруженными через интерфейс АИС. В процессе подготовки данных для численных данных производится нормализация, для текстовых – токенизация и лемматизация [149]. Для файлов выполняется операция извлечения текстовой информации и свойств (атрибутов), которые затем также преобразуются к численной форме.

Каждому информационному объекту соответствует категория $y_j \in Y$. Таким образом, имеется отображение $X \to Y$. Данное отображение является непрерывным, но не инъекцией. При достаточно большом размере множества X отображение $X \to Y$ может быть сюръекцией, т.е. для каждой категории $y_j \in Y$ будет существовать хотя бы один прообраз $x_i \in X$.

Обозначим как NN нейронную сеть, осуществляющую отображение $X \to Y$:

$$NN(X) = Y. \tag{4.24}$$

Так как отображение $X \to Y$ непрерывно, то его можно аппроксимировать нейронной сетью (следует из теоремы Цыбенко [115]). Тогда получим

$$\forall x_i \in X(NN(x_i) = y_j \mid j = 1, ..., M). \tag{4.25}$$

На второй стадии осуществляется распределение информации в АИС в процессе ее функционирования.

Пусть в АИС введен набор новых данных (файлов) $X = \{x_1, ..., x_N\}$. Тогда в соответствии с соотношением (4.25) необходимо осуществить следующую проверку:

– если $\forall x_i \in X(NN(x_i) = y_j)$ и данные (файл) x_i введены пользователем в категорию y_j, то они остаются без изменений;

– если $\exists x_i \in X(NN(x_i) = y_m)$, где y_m – категория вредоносного кода или зараженных файлов, то данные (файл) x_i не вносятся в АИС, отправляется предупреждение администратору АИС о попытке вредоносного проникновения в системы, пользователь, внесший данные x_i, временно блокируется;

– если $\exists x_i \in X(NN(x_i) = y_j)$, но данные (файл) x_i введены пользователем в категорию y_k ($y_k \neq y_j$), то возможны следующие варианты в зависимости от того, является ли x_i файлом или данными:

Если x_i – введенные пользователем в форму данные, то осуществляется проверка следующих условий:

1) если категория y_j свободная (пустая), то данные x_i переносятся из категории y_k в y_j;

2) если категория y_j занята данными x_q и $NN(x_q) = y_j$, то данные x_i сохраняются в некоторый буфер, а пользователю выдается предупреждение о дублировании данных и неправильном вводе x_i;

3) если категория y_j занята данными x_q и $NN(x_q) \neq y_j$, то данные x_q сохраняются в некоторый буфер, данные x_i переносятся из категории y_k в y_j, пользователю выдается предупреждение о неправильном вводе x_q.

Если x_i – файл, то осуществляется проверка следующих условий:

1) если категория y_j свободная (пустая), то файл x_i переносится из категории y_k в y_j;

2) если категория y_j заполнена, но может включать подмножество файлов $X_j = \{x_q \mid NN(x_q) = y_j\}$ ($\left|X_j\right| > 1$, $X_j \subseteq X$), то файл x_i переносится из категории y_k в y_j;

3) если категория y_j заполнена, но может включать подмножество файлов $X_j = \{x_q \mid NN(x_q) = y_j\}$ ($\left|X_j\right| > 1$, $X_j \subseteq X$), но $\exists x_q \in X_j(NN^*(x_q) \approx NN^*(x_i))$, то файл x_i является дубликатом файла x_q, где NN^* – нейронная сеть NN с исключением нескольких последних слоев, что позволяет вместо категории определить ключевые признаки данных; тогда пользователю выдается предупреждение о дублировании файлов и неправильном вводе x_i;

4) если категория y_j заполнена файлом x_q, но не может включать несколько файлов ($\left|X_j = \{x_q\}\right| = 1$), и $NN(x_q) = y_j$, то файл x_i

сохраняется в некоторый буфер, а пользователю выдается предупреждение о дублировании файлов и неправильном вводе x_i;

5) если категория y_j заполнена файлом x_q, но не может включать несколько файлов ($\left|X_j = \{x_q\}\right| = 1$), и $NN(x_q) \neq y_j$, то x_q сохраняются в некоторый буфер, x_i переносятся из категории y_k в y_j, пользователю выдается предупреждение о неправильном вводе x_q.

Таким образом, представленный нейросетевой метод классификации и распределения данных включает основные случаи добавления информации в АИС. За счет применения нейронной сети (или их множества) для классификации входящих данных и файлов возможно автоматизировать распределение информации по заданным категориям.

4.5. НЕЙРОСЕТЕВОЙ МЕТОД АДАПТАЦИИ ИНФОРМАЦИОННОЙ СИСТЕМЫ

Рассмотрим формализованное представление нейросетевого метода адаптации параметров интерфейса, основанное на теоретико-множественном представлении. В общем виде нейросетевой метод осуществляет следующее преобразование:

$$NN(IP) = V, \tag{4.26}$$

где NN – многослойная плотная нейронная сеть; IP – множество характеристик пользователя; V – множество параметров интерфейса: разметка интерфейса, качество визуализации, размер шрифта, тип интерфейса (обычный, расширенный, специализированный под категорию пользователей).

Нейронная сеть реализует преобразование $IP \to V$, возможность такого преобразования обоснована доказательством теорем Хехт-Нильсена и Колмогорова–Арнольда: всегда существует трехслойная нейронная сеть, для любого множества пар $\{(A, B)\}$ осуществляющая отображение $A \to B$ [116, 117].

Множество характеристик пользователя P стоит разделить на три подмножества:

$$IP = IP_P \cup IP_{SW} \cup IP_{HW}, \tag{4.27}$$

где IP_P – множество личностных характеристик пользователя: пол, возраст, образование, опыт работы, квалификация и т.д.; IP_{SW} – множество программных характеристик терминала пользователя: опера-

ционная система, браузер (для Web-систем), параметры окружения (размер окна, масштабирование), язык системы и т.д.; IP_{HW} – множество аппаратных характеристик терминала пользователя: тип оборудования (компьютер, мобильное устройство, планшет и т.д.), мощность процессора (количество ядер, частота), скорость подключения к серверу, наличие графического процессора и т.д.

Таким образом, нейронная сеть после успешного обучения сможет определять индивидуальные параметры интерфейса для каждого пользователя по уникальным исходным данным.

Формализуем основные этапы метода адаптации интерфейса.

1. Анализ структуры информационной системы и варьируемых параметров интерфейса. В ходе данного этапа выделяется множество изменяемых параметров интерфейса V для конкретной информационной системы.

2. Анализ характеристик пользователей. Осуществляется выбор элементов характеристик для каждого подмножества IP, определение области значений каждого элемента.

3. Выбор критерия оценки адаптивности информационной системы. При решении задачи адаптации параметров интерфейса необходимо осуществить оценку исходной системы и полученного решения, а также положительный эффект от применения того или иного инструмента или метода. Тогда в качестве целевой функции выберем критерий адаптивности, представленный формулой (2.22).

4. Формирование набора исходных данных (X, Y) для обучения нейронной сети. Набор входных данных X содержит обработанный и нормированный вектор характеристик пользователя, выходных данных Y – значения параметров интерфейсов. Для данных, если они представлены перечнем элементов множества, рекомендуется переход к бинарным матрицам, так как это способно повысить точность обучения нейронной сети.

5. Формирование структуры нейронной сети. В рамках данной задачи достаточно многослойной плотной сети, однако для ряда задач возможно использование рекуррентных (LSTM) сетей. Данный этап имеет большое значение и не всегда структура входных, скрытых и выходных слоев сети может быть выбрана за один подход правильно. Поэтому в процессе нахождения оптимальной конфигурации нейронной сети первый этап может выполняться многократно с учетом полученных корректив.

6. Обучение нейронной сети, достижение необходимой точности и сохранение модели сети для последующего использования.

7. Подготовка API (прикладных программных интерфейсов) адаптации параметров интерфейса. На данном этапе осуществляется разработка программных интерфейсов на основе архитектуры REST и обученной модели нейронной сети с сохраненными весовыми коэффициентами. Это позволяет отправить запрос из любого приложения к API и получить необходимый ответ. В запросе будут храниться данные для входного слоя – характеристики пользователя, а в ответ на него будут возвращаться значения выходного слоя (параметры интерфейса). Это позволит использовать программную реализацию метода адаптации для различных архитектур, платформ, модулей и информационных систем.

8. Интеграция API в информационную систему. На данном этапе необходимо реализовать сбор и отправку пользовательских данных из информационной системы в API адаптации, а также прием информации о настройках интерфейса и их автоматическое применение.

9. Адаптация интерфейса информационной системы. При первом входе в систему и регистрации пользователь вносит необходимые данные, система собирает сведения о его программном и аппаратном обеспечении, а также личные данные, после чего набор данных через REST API передается в программный модуль адаптации. Полученные в ответ значения выходного слоя нейронной сети устанавливаются в качестве текущих настроек интерфейса для данного пользователя. Значения настроек записываются в базу данных и при повторном входе в систему выставляются автоматически.

10. Оценка адаптивности интерфейса и сравнение полученного значения критерия с исходным (до применения метода). В случае, если значение оценки адаптивности не удовлетворяет установленному порогу, принимается решение о проведении дополнительных исследований.

Таким образом, представленный нейросетевой метод позволит успешно решать задачу адаптации параметров интерфейса информационной системы, применяя методы машинного обучения. Программная реализация предложенного метода может быть осуществлена путем применения нейросетевых каналов адаптации данных.

Научная новизна подхода заключается в разработке алгоритмического обеспечения для автоматизации сбора, анализа данных и адаптации интерфейса за счет использования и интеграции нейронных сетей.

4.6. НЕЙРОСЕТЕВОЙ МЕТОД УПРАВЛЕНИЯ В АИС

Как показал анализ существующих систем управления, во многих из них присутствует эффект запаздывания реакции системы на действия пользователя или иные внешние факторы. Особое значение это

имеет в системах, где объекты обладают инертностью, и даже мгновенная реакция системы управления не позволяет осуществить выполнение действия с достаточной скоростью.

Существующие алгоритмы управления, основанные на линейных законах, не позволяют осуществить достоверное прогнозирование действий пользователя, объектов и внешней среды, чтобы компенсировать запаздывание системы управления. Поэтому перспективным направлением является применение нейронных сетей для прогнозирования состояния пользователя или внешней среды и управления компонентами АИС. Здесь можно выделить два варианта решения задачи.

Первый вариант основан на использовании нейронных сетей для классификации действий пользователя. Обозначим данный подход как частично нейросетевое управление, так как нейронные сети будут использоваться только для выбора режима функционирования системы. Для его реализации необходимо выполнить следующие действия:

1. Осуществить сбор данных UC о пользователе, системе и внешних объектах (обозначим их совокупность как множество характеристик информационных объектов).

2. Определить перечень возможных состояний US информационных объектов, а также состояний S системы управления.

3. Соотнести последовательности данных об информационных объектах с выполняемыми системой управления в этот момент действиями A.

4. Использовать полученные пары «набор координат–действие» в качестве исходных данных для обучения нейронной сети. В качестве архитектуры рекомендуется использовать многослойную плотную либо рекуррентную сеть.

5. Осуществить обучение нейронной сети.

6. Разработать вторую нейронную сеть, которая на основе значений i-го набора формирует значения $(i + 1)$-го набора. Таким образом, возможно осуществить прогнозирование состояния информационных объектов и минимизировать влияние запаздывания процессов обработки и передачи данных в системе управления.

7. Интеграция нейронных сетей в систему управления для прогнозирования и последующей классификации действий информационных объектов и выбора на основе полученных от нее результатов необходимых режимов функционирования или действий. Так как режим определяется не на основе текущих данных, а с некоторым прогнозом, система реагирует на действия раньше, что приводит к сокращению негативных последствий и эффекта запаздывания.

Второй вариант основан на использовании нейронных сетей для полноценного управления системой. Обозначим данный подход как полное нейросетевое управление. Его реализация включает следующие этапы:

1. Осуществить сбор данных U о состоянии информационных объектов.
2. Осуществить сбор данных S о состоянии системы управления.
3. Реализовать функцию награды Q, зависящую от текущего состояния системы S. Данная функция стремится к 1 при оптимальном состоянии системы (информационные объекты функционируют в соответствии с требованиями, все ограничения выполнены, метрики показывают допустимые значения) и к 0 – в остальных случаях (понижение значений метрик качества функционирования системы, неправильные параметры информационных объектов и т.д.).
4. Сформировать набор действий A, которыми можно воздействовать на систему.
5. Обучить рекуррентную нейронную сеть, на вход которой поступает сочетание состояния системы S, состояния информационных объектов U и возможные действия A. Для каждого сочетания вычисляется функция Q. Тогда для наиболее правильных действий в текущем состоянии системы выход нейронной сети будет стремиться к 1, а для остальных (неправильных) – к нулю.
6. Нейронная сеть может тренироваться на подготовленных или сгенерированных данных, в качестве целевой функции можно использовать максимальную сумму значений функций наград или максимальное время нахождения системы в оптимальном режиме функционирования.
7. Обученная нейронная сеть интегрируется в систему управления. Тогда по текущему состоянию системы и действиям пользователя она подбирает действие, обеспечивающее максимальную награду и, следовательно, оптимальное в данной ситуации.

Второй подход выглядит более перспективным, однако отличается большей трудоемкостью и сложностью реализации, необходимостью длительного обучения и тщательного тестирования.

Перед формализацией нейросетевого метода управления рассмотрим основные объекты в данной области. Пусть задана математическая модель системы управления M :

$$M(U,\ S) = A, \tag{4.28}$$

где U – множество информационных объектов, включающее множество их характеристик UC и состояний US в моменты времени T:

$$U = UC \times US \times T. \tag{4.29}$$

Для конкретного информационного объекта получим

$$u_k = \{UC_{k,\,i},\ US_{k,\,i},\ t_i \mid i = 1,\ ...,\ I_k\},\ u_k \in U, \tag{4.30}$$

где I_k – общее количество замеров состояний объекта.

Каждому замеру соответствует V элементов $uc_{k,\,i,\,j}$, отражающих характеристики объекта во время t_i:

$$UC_{k,\,i} = \{uc_{k,\,i,\,j} \mid j = 1,\ ...,\ V\},\ UC_{k,\,i} \in UC. \tag{4.31}$$

В каждый момент времени t_i объект имеет определенное и единственное состояние из множества всех возможных $US = \{uc_1,\ ...,\ uc_m\}$:

$$US_{k,\,i} = uc_l,\ uc_l \in US. \tag{4.32}$$

S – состояние системы управления, включающее множество элементов s_n, характеризующих параметры системы в текущий момент времени:

$$S = \{s_1,\ ...,\ s_n\}. \tag{4.33}$$

A – действие системы управления в ответ на состояние системы и информационных объектов:

$$A = \{a_1,\ ...,\ a_n\}. \tag{4.34}$$

Формализуем метод частичного нейросетевого управления.

Сформируем структуру нейронной сети NN_C, осуществляющей преобразование:

$$NN_C : UC \to US. \tag{4.35}$$

Отметим, что в качестве входных данных о характеристиках объекта могут использоваться как только текущие значения (имеем преобразование $UC_{k,\,i} \to US_{k,\,i}$), так и некоторая их совокупность за определенный отрезок времени *w*. Обозначим ее следующим образом:

$$UC_{k,\,i}^{w} = \{UC_{k,\,i-w},\ UC_{k,\,i-w+1},\ ...,\ UC_{k,\,i}\}. \tag{4.36}$$

Второй вариант может быть предпочтительнее, так как позволяет выделить динамику действий пользователя. В общем виде тогда обозначим преобразование NN_C в следующей форме:

$$NN_C : UC^w \to US,\ w = 0,\ ...,\ H, \tag{4.37}$$

где H – длина истории характеристик информационного объекта.

Для устранения эффекта запаздывания при принятии решений в системе управления необходимо оперировать не только текущими характеристиками объектов и их состояниями, но и прогнозировать предстоящие. Обозначим *d*-прогнозом характеристик объекта относительно текущего момента времени t_i значения его характеристик через время t_{i+d} :

$$UC^d_{k,\,i} = \{UC_{k,\,i+1}, UC_{k,\,i+2},\ ...,\ UC_{k,\,i+d}\}. \tag{4.38}$$

Для получения *d*-прогноза обучим нейронную сеть прогнозирования NN_F , принимающую на вход историю характеристик информационного объекта, а на выходе формирующую *d*-прогноз:

$$NN_F : UC^w \to UC^d,\ w = 0,\ ...,\ H,\ d = 0,\ ...,\ H_2, \tag{4.39}$$

где H_2 – максимальная величина прогноза характеристик информационного объекта.

Тогда для решения задачи частичного нейросетевого управления с учетом эффекта запаздывания на *d*-моментов времени необходимо сформулировать отображение:

$$NN_A : NN_C(NN_F(UC^w)) \to A. \tag{4.40}$$

Таким образом, три нейронные сети работают последовательно:

– 1-я нейронная сеть осуществляет прогноз характеристик информационного объекта $NN_F(UC^w) \to UC^d$;

– 2-я сопоставляет с характеристиками объекта его возможное состояние: $NN_C(UC^d) \to US^d$;

– 3-я сопоставляет состояние объекта и необходимое действие в системе управления: $NN_A(US^d) \to A$.

Таким образом, решаются две обозначенные проблемы: минимизируется эффект запаздывания за счет обработки не текущих данных, а прогноза; осуществляется управление системой на основе анализа состояния информационных объектов в автоматическом режиме.

На этапе (4.40) возможны различные модификации: к состоянию информационных объектов могут быть добавлены их характеристики или параметры самой системы управления, текущий режим и т.д. Увеличение количества входных переменных позволит реализовать систему на основе нейросетевого управления более гибко.

Рассмотрим реализацию полного нейросетевого управления.

На первом этапе осуществим сбор данных о характеристиках $UC_{k,\,i}$ и состоянии $US_{k,\,i}$ информационного объекта.

На следующем этапе необходимо реализовать функцию наград. Первый подход основан на оценке отклонения $\Delta uc_{k,\,i,\,j}$ значений характеристик $uc_{k,\,i,\,j}$ объекта от заданного эталона (нормы) $uc^{*}_{k,\,i,\,j}$:

$$Q = \begin{cases} 1, \text{ если } \left| uc_{k,\,i,\,j} - uc^{*}_{k,\,i,\,j} \right| < \Delta uc_{k,\,i,\,j}, \\ 0, \text{ если } \left| uc_{k,\,i,\,j} - uc^{*}_{k,\,i,\,j} \right| \geq \Delta uc_{k,\,i,\,j}, \end{cases} \tag{4.41}$$

где $\forall uc_{k,\,i,\,j}\, \exists (\Delta uc_{k,\,i,\,j},\ uc^{*}_{k,\,i,\,j})$.

Второй подход основан на использовании некоторых функциональных зависимостей или метрик, оценивающих характеристики или состояния объекта относительно их эталонного поведения:

$$Q = \begin{cases} 1, \text{ если } \left| TU(u_i,\ t) - TS(u_i,\ t) \right| < \Delta T, \\ 0, \text{ если } \left| TU(u_i,\ t) - TS(u_i,\ t) \right| \geq \Delta T, \end{cases} \tag{4.42}$$

где $TU(u_i,\ t)$ – эталонное поведение объекта u_i в момент времени t; $TS(t)$ – реальное поведение объекта u_i в момент времени t; ΔT – допустимое отклонение поведения (метрик).

Далее сформулируем набор действий A, которые может осуществлять система управления. Обучим на наборе собранных данных о взаимодействии информационных объектов с системой рекуррентную нейронную сеть, осуществляющую отображение:

$$NN_R : (S,\ U,\ A) \to Q. \tag{4.43}$$

Обучение осуществляется итерационно. Для каждого возможного набора S, U, A вычисляется значение функции Q. Затем обученная сеть используется для определения для текущих S и U оптимального действия $a_j \in A$, при котором $Q \to \max$.

Представленный нейросетевой метод управления может использоваться в системах с большим количеством информационных объектов, законы функционирования которых не обозначены в явном виде. Научная новизна предложенного подхода заключается в формализации и применении трех моделей нейронных сетей (классификации, прогнозирования и управления) с целью компенсации эффекта запаздывания в системах управления. В полном нейросетевом управлении используется механизм обучения функцией наград (Q-learning), что обеспечивает большую гибкость работы системы управления в условиях отсутствия законов функционирования информационных объектов.

4.7. ВЫВОДЫ

В данной главе рассмотрены нейросетевые методы, направленные на автоматизацию анализа, обработки и передачи данных. Их применение в рамках разработанной методологии структурно-параметрического синтеза АИС обеспечивает выполнение ее ключевого этапа (Нейросетевого проектирования), который целиком основан на применении и апробации изложенных в главе методов.

Нейросетевой метод обработки и передачи информации в АИС реализует основную концепцию связей в нейросетевой архитектуре АИС – нейросетевые каналы данных, применяемые для анализа, обработки информации и выбора способов ее передачи. Новизна нейросетевого метода обработки и передачи информации заключается в разработке теоретических основ применения нейросетевых каналов данных, формализованного математического и алгоритмического обеспечения, позволяющего их осуществить программную реализацию.

Нейросетевой метод автоматической генерации данных в АИС заключается в использовании нейронных сетей для решения задач генерации данных, реализации алгоритмов решения задач генерации данных, автоматическом подходе к определению класса задач генерации данных, выбору архитектуры и обучению нейронных сетей, адаптации и реализации оценок IS и FID для произвольных структур данных. Для каждого класса задач генерации данных предлагается подробный и формализованный алгоритм решения.

Нейросетевой метод автоматической переадресации основан на теоретико-множественном анализе процесса переадресации и отличается применением нейронных сетей для определения оптимального исполнителя операции, времени и вероятности ее успешного выполнения, а также реализацией и формализацией понятий матрицы приоритетов операций и матрицы компетенций исполнителей для ранжирования операций и сопоставления их с уровнем подготовки исполнителей.

Нейросетевой метод классификации и распределения данных реализует процедуру анализа информации или файлов, позволяя на основе обученной нейронной сети осуществить автоматическое распределение, классификацию, обнаружение дубликатов и вредоносных данных. Данный метод направлен на снижение сложности процессов поддержки принятия решений, анализа и классификации в АИС.

Нейросетевой метод адаптации информационной системы формализован в теоретико-множественном представлении, рассмотрены этапы его применения. Научная новизна метода заключается в автоматизации сбора, анализа данных и настройки интерфейса за счет использования и интеграции нейронных сетей в информационную систему. Применение метода позволит повысить степень адаптивности АИС и упростить процедуру персонализации системы.

Нейросетевой метод управления основан на использовании нейронных сетей для автоматизации процессов анализа, прогнозирования и принятия решений. В частичном нейросетевом управлении для этого используются три связанные модели нейронных сетей, каждая из которых позволяет решить отдельную задачу. В полном нейросетевом управлении реализуется механизм Q-learning, а нейронная сеть в процессе обучения и функционирования пытается максимизировать функцию наград, характеризующую состояние системы управления.

Методы, рассмотренные в данной главе, отличаются научной новизной и разработаны для автоматизации деятельности разработчика АИС и позволяют снизить сложность реализации компонентов АИС.

Глава 5
АПРОБАЦИЯ И ОЦЕНКА ЭФФЕКТИВНОСТИ НЕЙРОСЕТЕВЫХ МЕТОДОВ ПРИ РЕШЕНИИ ЗАДАЧ АНАЛИЗА, ОБРАБОТКИ И ПЕРЕДАЧИ ДАННЫХ

В предыдущей главе успешно осуществлена формализация нейросетевых методов взаимодействия с информацией в АИС. Далее необходимо осуществить программную реализацию данных методов, оценить их эффективность и применимость для решения прикладных задач анализа, обработки и передачи данных в АИС.

5.1. ПРАКТИЧЕСКАЯ РЕАЛИЗАЦИЯ НЕЙРОСЕТЕВЫХ КАНАЛОВ ДАННЫХ

На основе изложенного в рамках нейросетевого метода обработки и передачи данных формализованного описания нейросетевых каналов данных осуществлена их программная реализация. В качестве языка программирования использовался Python, обладающий рядом преимуществ при решении задач прототипирования программных продуктов, а также обеспечивающий широкую поддержку сторонних библиотек в области машинного обучения. Для реализации нейронных сетей применялась распространенная и многократно апробированная библиотека Keras.

Разработка нейросетевого канала данных осуществлялась в два этапа. На первом этапе сформирован класс *NNDC*, реализующий программную логику, изложенную в формулах (4.1) – (4.9).

Далее необходимо проанализировать эффективность нейросетевых каналов данных относительно существующих методов передачи данных и организации межмодульного взаимодействия в информационных системах.

Определение 8. *Под термином «нейросетевое взаимодействие модулей» будем понимать организацию связей между модулями на основе нейросетевых каналов данных различных категорий. Тогда термин «нейросетевой метод» в данной работе будет означать применение нейросетевого взаимодействия для решения задач организации взаимодействия между модулями.*

В качестве оценочных критериев будем использовать следующие:

– вычислительная сложность алгоритма *O*, вычисляемая путем оценки сложности используемых процедур и функций по O-нотации [76];

– точность P решения поставленной задачи, выражающая количество корректно решенных задач от их общего количества в тестовой выборке [150];

– цикломатическая сложность *CC*, отражающая количество переходов, циклов, генераторов, обработчиков исключений и логических операторов [151];

– метрики сложности программного кода Холстеда *HAL* [152, 153];

– индекс поддерживаемости кода *MI*, отражающий, насколько сложно будет поддерживать или редактировать фрагмент программы [78];

– метрика сложности программного кода Джилба *J*, определяемая по формуле $J = c / n$, где c – количество управляющих операторов, n – общее число операторов программы [154];

– экспертная оценка времени реализации T на основе методики COCOMO II [155], определяемая формулой $T = c_b (W)^{d_b}$, где $W = a_b (KLOC)^{b_b} EAF$ – трудоемкость, $KLOC$ – количество тысяч строк кода, остальные коэффициенты EAF заданы таблицей [156];

– экспертная оценка стоимости реализации V на основе методики COCOMO II [157], зависящая от времени разработки и стоимости труда разработчиков $V = WV_{dev} K_{cc}$, где V_{dev} – зарплата разработчика в месяц, руб.; K_{cc} – коэффициент, учитывающий начисления на заработную плату (отчисления на социальные нужды), равный 1.3.

Отметим некоторые особенности расчета представленных метрик.

Вычислительная сложность алгоритма нейросетевых каналов рассчитывалась отдельно для процесса обучения нейронной сети (определенная в соответствии с исследованием [158]) и для процесса функционирования при передаче и обработке данных.

Циклометрическая сложность, метрики сложности программного кода Холстеда и индекс поддерживаемости кода определяются с помощью библиотеки оценки программного кода Radon (Python), что позволит автоматизировано получить объективную и точную оценку. Циклометрическая сложность оценивается индексами, начиная от наилучшего (A) и далее (до F), а также количественно (заданным числом McCabe); записывается в форме «тип (F – функция, M – метод, C – класс), имя функции, значение сложности». Под *Sum*(*CC*) будем понимать суммарную сложность, под *Average*(*CC*) – среднюю сложность программного кода, обе оценки заданы числом McCabe. Метрики Холстеда включают определение словаря, длины, объема и сложности программы, трудозатрат разработчика, времени разработки и количе-

ства возможных ошибок [159]. На основе данных о количестве различных операций *h1* и операндов *h2* и общем количестве операций *N1* и операндов *N2* соответственно.

Метрика сложности программного кода Джилба определяется вручную по заданной формуле для классического и нейросетевого подходов.

Оценка времени и стоимости реализации осуществляется на основе методики COCOMO II экспертным путем. V_{dev} при расчетах принята равной 100 000 руб.

В качестве эталона, с которым будут сравниваться нейросетевые каналы данных, примем программное обеспечение, реализующее сетевые протоколы, используемые в примерах NNDC, а также необходимые процедуры обработки данных. Таким образом, будет повышена объективность эксперимента, так как можно будет оценить отклонение метрик от эталонных значений при одинаковых условиях эксперимента. Данное соответствие используемых сетевых протоколов является обязательным условием сравнения, так как влияние используемой технологии передачи данных на метрики может быть существенным.

Схема эксперимента имеет следующий вид: сначала реализуется классический подход к взаимодействию модулей на основе сетевого протокола и необходимого алгоритмического обеспечения. Осуществляется оценка процесса разработки классического взаимодействия модулей по выбранному набору критериев. Далее процесс повторяется для метода на основе нейросетевого взаимодействия модулей. В основе нейросетевых каналов используются те же сетевые протоколы для обеспечения объективности эксперимента. Полученное решение оценивается.

Далее рассмотрим программную реализацию и специфику каналов передачи данных *NNDC_T* и канала адаптации *NNDC_A*. В рамках их апробации и оценки эффективности будут оцениваться два соответствующих метода – обработки и передачи данных и адаптации.

5.1.1. Апробация и оценка нейросетевого метода обработки и передачи данных

Канал передачи данных *NNDC_T* в данном примере будет являться нейросетевым каналом данных первой степени, реализуя лишь распознавание и приведение к требуемому виду типов данных.

В ходе экспериментальных исследований работоспособности данного типа канала в рамках информационной системы на базе фреймворка Flask решалась задача передачи текстовых данных трех

типов: JSON, list (список в Python) и случайный текст (содержимое форм на веб-страницах), для каждого из которых задано по две категории (табл. 5.1). Нейросетевой канал передачи данных должен успешно определить тип отправляемых данных и, в случае соответствия структуры, заданной в принимающем модуле, осуществить передачу данных.

Рассмотрим задачу передачи данных от одного модуля к другому с необходимостью классификации категорий данных (представленных в табл. 5.1). Осуществим сравнение нейросетевого канала передачи данных первой степени с классическим (алгоритмическим) подходом в рамках решения данной задачи.

5.1. Описание категорий данных

Обозначение	Описание	Примеры
JSON(LP)	JSON с данными авторизации (логин: пароль). Длина логина: 8 – 30 символов, пароля: 8 – 32 символа	{'username1': 'qwerty12345678'} {'example-name12345': '#$123456abcd'}
JSON(PARAM)	JSON со значением некоторого параметра (параметр: значение). Длина имени параметра: 2 – 10 символов, значения – до 30 символов	{'param': '100'} {'ab': 'none'} {'d99': '0.79'}
List(data)	Список с 5 значениями датчика, целые числа	[120, 90, 240, 190, 130]
List(control)	Список с управляющим сигналом (ИД операции, ИД оборудования, 3 значения переменных), целые числа	[2, 1911, 60, 70, 65]
Text(name)	Текст с именем и фамилией пользователя	'Ivan Ivanov'
Text(other)	Случайный текст	'Some random text example'

Алгоритмический подход будет заключаться в формировании некоторого набора условий, позволяющих определить сначала тип данных (JSON, list, текст), а затем – категорию на основе некоторых общих признаков (используемые символы, длина данных).

Нейросетевой подход в данном случае работает как «черный ящик», принимая массив исходных данных с размеченными категориями, и осуществляет классификацию поступающей информации по заданным шести категориям (табл. 5.1).

Для обучения нейронной сети классификации типов данных использовалось 60 000 сгенерированных примеров (по 10 000 на каждую категорию). Итоговая точность нейронной сети на тестовых данных стремилась к 1.

Далее необходимо осуществить сравнение по первой метрике – точности решения задачи. В случае процесса распознавания типов данных эта оценка является определяющей. Проведено шесть испытаний, в каждом оценивался процент корректного распознавания на 100 сгенерированных примерах каждой категории. Сравнение точности решения задачи представлено на рис. 5.1, обозначения категорий аналогичны указанным в табл. 5.1.

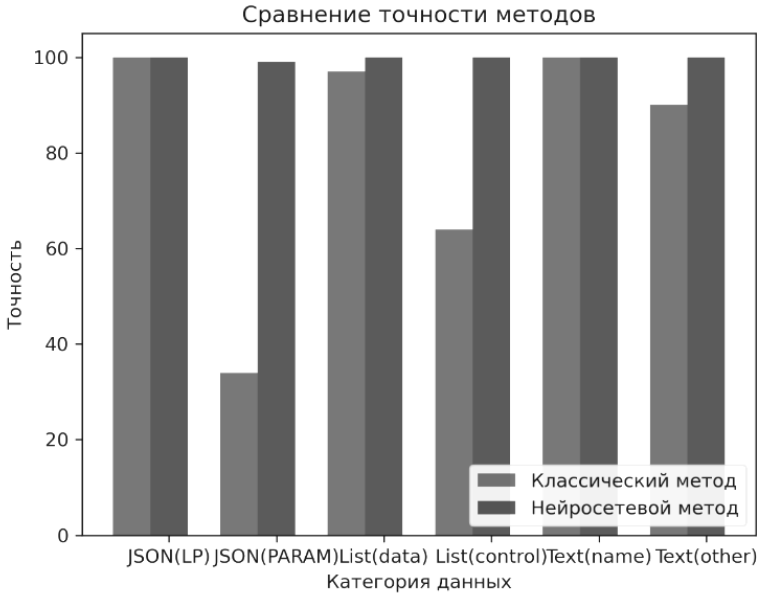


Рис. 5.1. Сравнение точности алгоритмического и нейросетевого подходов при классификации типов данных

Далее рассмотрим итоговую таблицу сравнения двух подходов по выбранным ранее метрикам (табл. 5.2).

Нейросетевой канал передачи данных за счет анализа большого объема информации показывает немного большую точность при классификации категорий данных.

Циклометрическая сложность и метрики сложности программного кода Холстеда (сложность, трудоемкость и время) нейросетевого взаимодействия показывают лучшие значения относительно классического подхода, что обусловлено меньшим количеством операторов и более простой структурой программного кода. Сложность по метрике Джилба сравнима, однако общее количество условных операторов намного меньше у нейросетевого метода (почти в 12 раз).

Время разработки и стоимость в метрике COCOMO II при использовании нейросетевых каналов данных сокращается, однако данная метрика недостаточно точно отражает общее снижение трудоемкости, так как количество строк у нейросетевого канала больше, хотя сложность их реализации – меньше.

Индекс поддерживаемости кода у обоих подходов сравним.

5.2. Сравнение классического и нейросетевого методов по передаче данных

Критерий	Классический метод	Нейросетевой метод
O	*O*(*N*) – для алгоритма определения параметров интерфейса	*O*(*N*) – для функционирования; $O(m^r)$ – для процесса обучения на *m*-примерах, $r > 1$ [158]
P (%)	80.8 (от 34 до 100)	99.8 (от 99 до 100)
CC	*Sum* (*CC*) = 23 *Average* (*CC*) = B (5.75)	*Sum* (*CC*) = 19 *Average* (*CC*) = A (1.41)
HAL	*h1*: 10, *h2*: 47, *N1*: 29, *N2*: 54	*h1*: 2, *h2*: 4, *N1*: 2, *N2*:4
MI	A	A
J	0.255	0.25
T (человеко-месяцы)	1,12	0,95
V (рублей)	15 994	10 388

Вычислительная сложность алгоритмов при функционировании у обоих методов аналогична, но процесс обучения нейронной сети алгоритмически более сложен. Однако существующие библиотеки по машинному обучению (Keras, Caffe, TensorFlow, Torch, Theano и т.д.) обладают высоким уровнем абстракции, просты в освоении и использовании. Таким образом, параметр r в оценке сложности будет стремиться к единице с развитием программных инструментов обучения. Поэтому можно принять допущение, что для конечного разработчика вычислительная сложность процесса обучения нейронной сети оказывает незначительное влияние на общую сложность процесса разработки.

5.1.2. Апробация и оценка нейросетевого метода адаптации

Нейросетевой метод адаптации информационных систем реализуется на основе нейросетевого канала адаптации *NNDC_A*. Он имеет третью степень сложности и реализует функции анализа, преобразования и выбора протокола передачи данных.

Основной функцией рассматриваемого нейросетевого канала данных является адаптация интерфейса под параметры пользователя. Исходными параметрами, поступающими на вход нейросетевого канала данных, являются: роль пользователя, пол пользователя, возраст пользователя, образование пользователя, опыт работы за компьютером, ширина окна браузера, высота окна браузера, тип клиента, браузер, язык браузера, количество ядер процессора, скорость подключения к серверу, наличие графического процессора.

На первом этапе осуществляется определение структуры входных данных. В рассматриваемом примере это 13 входных значений, запакованных в формат JSON. Нейронная сеть NN^{ct}, аналогичная использованной в прошлом примере, успешно определяет тип данных – JSON – после чего осуществляет их преобразование в список. Полученный массив поступает на вход второй нейронной сети преобразования данных $NN_{i,j,A}^{proc}$. Значения входных параметров преобразуются в бинарную матрицу. Далее третья нейронная сеть $NN_{i,j,A}^{tr}$ определяет оптимальный протокол передачи данных. Она обучается на проанализированных сочетаниях типов данных с указанием выбранных в тех или иных ситуациях протоколов.

Осуществим сравнение нейросетевого канала адаптации и классического метода на основе сочетания сетевого протокола POST для передачи JSON с алгоритмическим обеспечением анализа данных о пользователе для последующей адаптации интерфейса системы. Данный алгоритм в общем виде представлен на рис. 5.2.

Рис. 5.2. Алгоритм адаптации интерфейса

Все этапы и операции, входящие в состав алгоритма адаптации, реализуются разработчиками вручную на основе некоторой экспертной информации.

Аналогично предыдущему эксперименту рассмотрим процесс обучения нейронной сети. Особое внимание уделим точности работы нейронной сети обработки информации, так как она определяет общую эффективность нейросетевого канала адаптации.

Для обучения нейронной сети обработки данных использовалось 7900 сгенерированных примеров (не менее 400 примеров на каждое значение выходных признаков). Объем тренировочной выборки достаточен, так как в соответствии с исследованиями, проведенными в работе [160], получено, что для каждого признака необходимо обеспечить около 50 – 100 входных наборов данных. Средняя точность нейронной сети на тестовых данных составила почти 93%.

Сравнительный анализ классического и нейросетевого методов к организации взаимодействия между модулями представлен в табл. 5.3.

Точность вычислений определяется эмпирическим методом на основе данных о 100 процедурах адаптации для каждого из подходов и последующих действиях пользователей: в случае, если выставленные

5.3. Сравнение классического и нейросетевого методов по адаптации данных

Критерий	Классический метод	Нейросетевой метод
O	*O*(*N*) – для алгоритма определения параметров интерфейса	*O*(*N*) – для функционирования; *O*(m^r) – для процесса обучения на *m*-примерах, *r* > 1 [158]
P (%)	82	93
CC	*Sum* (*CC*) = 67 *Average* (*CC*) = B (6.09)	*Sum* (*CC*) = 41 *Average* (*CC*) = A (2.43)
HAL	*h1*: 8, *h2*: 73, *N1*: 56, *N2*: 110	*h1*: 9, *h2*: 62, *N1*: 34, *N2*: 69
MI	A	A
J	0.55	0.22
T (человеко-месяцы)	2,08	1,86
V (рублей)	80 997	59 408

настройки не устраивали пользователя, задача считалась нерешенной. Нейросетевой канал данных за счет анализа пользовательских данных и предпочтений показывает немного большую точность, которая в дальнейшем с ростом количества обрабатываемых данных может быть улучшена. В текущем эксперименте нейронная сеть обучена на 7900 записях, в коммерческих системах тренировочная выбора должна быть увеличена для большей точности.

Циклометрическая сложность и метрики сложности программного кода Холстеда (сложность, трудоемкость и время) нейросетевого взаимодействия показывают лучшие значения относительно классического подхода, что обусловлено меньшим количеством операторов и более простой структурой программного кода. Большой эффект оказывает тот факт, что разработчику не требуется анализировать исходные данные, так как эту работу принимает на себя нейросетевой канал. Сложность по метрике Джилба за счет значительного сокращения ветвлений и циклов у нейросетевого канала данных показывает намного лучшие результаты.

Время разработки при расчете методом COCOMO II при использовании нейросетевых каналов данных сокращается, что также приводит к сокращению материальных затрат на оплату труда разработчиков.

Индекс поддерживаемости кода у обоих подходов сравним.

Вычислительная сложность алгоритмов при функционировании у обоих методов аналогична и определялась аналогично предыдущему эксперименту.

Таким образом, осуществлен сравнительный анализ метода организации взаимодействия между модулями на основе нейросетевых каналов данных с классическим методом, включающим существующие сетевые протоколы и требуемое алгоритмическое обеспечение для решения задачи. Эксперименты показали, что использование методов на основе нейросетевых каналов данных повысило точность решения поставленной задачи и в значительной мере снизило сложность программного кода. По большинству метрик предлагаемый нейросетевой метод показывает лучшие результаты, снижая сложность решаемой задачи организации связей и обработки данных между модулями для разработчиков, что позволило высвободить дополнительные ресурсы для решения более сложных и творческих задач.

5.2. ПРАКТИЧЕСКАЯ РЕАЛИЗАЦИЯ МЕТОДА ОЦЕНКИ ПРОИЗВОЛЬНЫХ GAN

Для проверки адекватности метода оценки качества GAN проведен ряд исследований на двух типах данных: в первом случае осуществляется генерация размеченного набора данных – изображений MNIST, во втором – исследуются неразмеченные датасеты Human Activity Recognition Using Smartphones (HAR, 7767 тренировочных наборов размерностью 561 значение, содержит сведения о человеческой деятельности с использованием набора данных смартфонов [161]) и Epileptic Seizure Recognition (ESR, 11 500 записей с 179 значениями [162]). Схема эксперимента устроена следующим образом: подбирается приемлемая структура GAN, после чего осуществляется исследование зависимости значений модифицированных метрик IS/FID от времени обучения (количества эпох). Дополнительно проверяется зависимость значений метрик от выбранных параметров H и N.

В ходе первого эксперимента построена многослойная сверточная GAN с применением слоев Conv1D, Conv2D и BatchNormalization для нормализации промежуточных значений активаций. Так как набор MNIST размечен на 10 классов, то $N=10$. Примем $H=100$ в рамках данного эксперимента, получив отображение из 768 значений пикселей

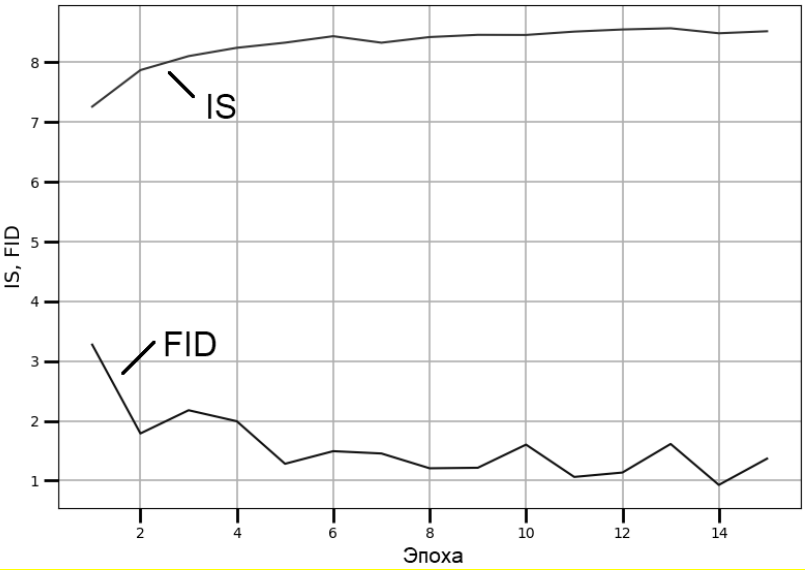


Рис. 5.3. Процесс обучения GAN

в вектор из 100 признаков. Обучение GAN осуществляем в течение 15 эпох на наборе из 50 000 тренировочных изображений. Результаты представлены на рис. 5.3.

Из графиков IS и FID можно отметить, что они достаточно адекватно отражают прогресс в обучении (рис. 5.4): от изначально нечеткого изображения с шумом (при IS = 7.25, FID = 3.27) получен достаточно качественный результат (IS = 8.52, FID = 1.36).

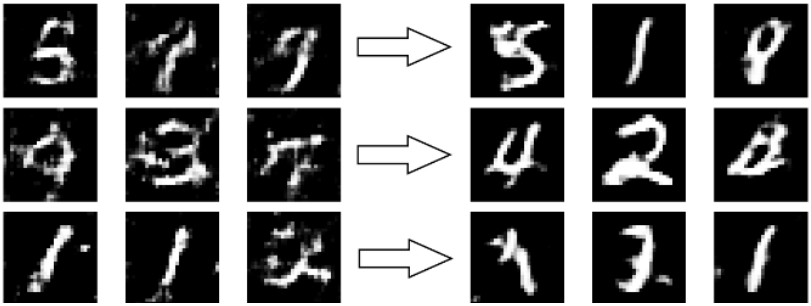


Рис. 5.4. Сгенерованные объекты датасета MNIST (1-я и 15-я эпохи)

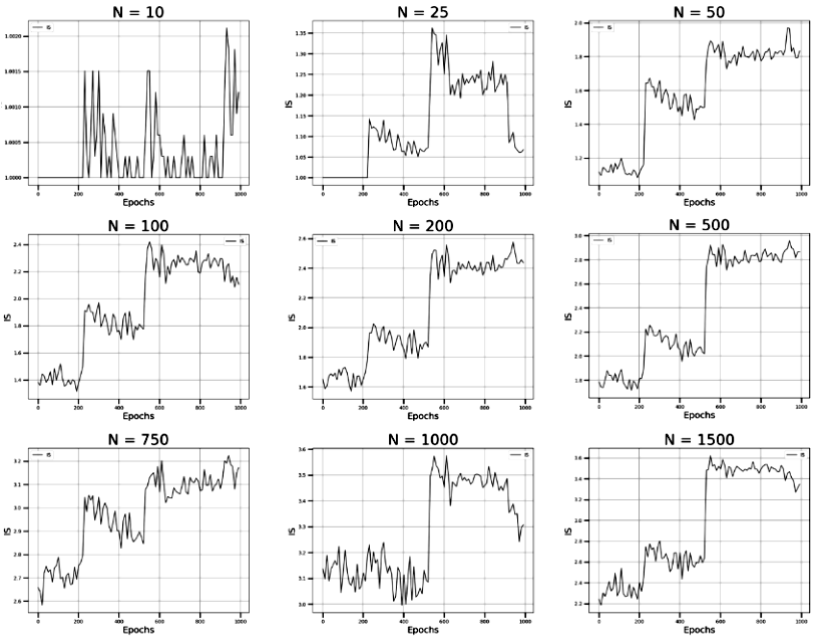


Рис. 5.5. Замеры IS при различных *N* при обучении GAN для датасета HAR

Второй эксперимент связан с генерацией численных данных на основе неразмеченных датасетов. Поэтому в соответствии с предлагаемым методом необходимо осуществить кластеризацию исходных данных по *N* категориям. Сначала рассмотрим обучение и оценку GAN для датасета HAR. Экспериментальные исследования показали высокую погрешность оценок IS и FID для GAN при генерации неразмеченных данных произвольной размерности. Для IS проведен ряд опытов с различным количеством N категорий: от 10 до 1500 (рис. 5.5). Процесс обучения GAN осуществлялся в течение 1000 эпох, каждые 10 эпох производился замер IS при различных *N*. Это позволило избегать погрешности, возможной при повторном обучении GAN.

Полученные результаты привели к следующим выводам: при $N \le 25$ процесс обучения оценивается некорректно. IS в этих диапазонах либо колеблется в приделах одной величины, либо падает, что не позволяет использовать ее при оценке качества GAN. С другой сто-

роны, при $N \geq 25$ процесс обучения отражен корректно: с ростом количества итераций оценка IS растет вместе с уменьшением ошибок дискриминатора и генератора. Причем корректность оценки IS не падает даже при $N > 1500$. Отметим, что при $N > 500$ увеличение значения IS весьма незначительно. Это можно объяснить избыточностью кластеров, имеющих небольшие отличия между собой. Избыточность кластеров также приводит к увеличению вычислительной нагрузки. Таким образом, лучшие результаты получены при $N \approx 500$.

Выдвинем следующую гипотезу: IS для произвольного неразмеченного набора данных $X = \{x_i\}$, где каждый x_i представлен XN элементами, определяется на основе распределения данных на N кластеров, где N выбирается на основе условия

$$1 + \frac{XN}{20} \leq N \leq 1 + XN. \tag{5.1}$$

Полученные результаты удовлетворяют данному условию. Левая и правая части двойного неравенства (5.1) получены на основе анализа экспериментальных данных (рис. 5.5). Причем правая часть неравенства обусловлена снижением вычислительной нагрузки и на точность определения IS влияния не оказывает.

Для проверки выдвинутой гипотезы о величине N дополнительно проведен эксперимент на наборе данных ESR (11 500 записей с 179 значениями). На основе условия (5.1) количество кластеров для неразмеченных данных должно находиться в интервале $10 \leq N \leq 180$. Результаты эксперимента (рис. 5.6) подтверждают выдвинутую гипотезу. При $N < 10$ величина оценки IS значительно падает. Оптимальная величина достигается при $N \to 180$.

Подведем итог по расчету IS для произвольных данных: размер N для размеченных данных соответствует количеству классов, для неразмеченных выбирается исходя из условия (5.1).

Далее рассмотрим возможность применения FID для оценки качества GAN при произвольных наборах данных. При вычислении IS на датасетах HAR и ESR также параллельно осуществлялся расчет FID при различных H. Итоговые результаты представлены на рис. 5.7 для обоих датасетов. Первый эксперимент показал, что оценка FID пропорциональна величине H, в данном случае замечена та же закономерность.

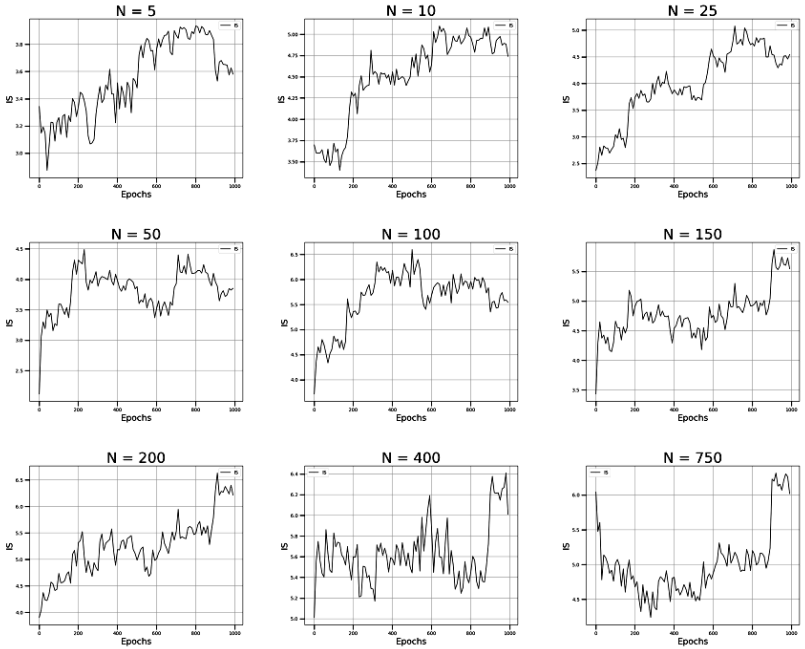


Рис. 5.6. Замеры IS при различных *N* при обучении GAN для датасета ESR

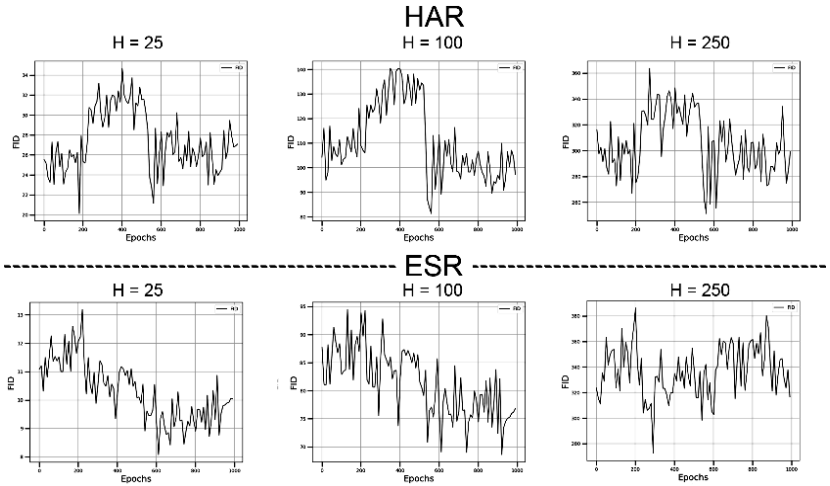


Рис. 5.7. Замеры FID при различных *H* при обучении GAN для датасетов HAR и ESR

Проведенные испытания показали, что для произвольных данных в соответствии с модификациями (4.14) возможен расчет FID, однако его значения не позволяют охарактеризовать качество GAN. В ряде случаев значение FID с ростом количества эпох увеличивалось, что не соответствовало процессу обучения. Для обоих датасетов изменение FID в конце обучения относительно его начала составило не более 12%, в среднем от 6 до 10%. Для сравнения – в первом эксперименте FID был сокращен почти на 60%.

Предложенный метод оценки качества GAN для произвольных данных был успешно апробирован в ходе трех экспериментов на датасетах MNIST, HAR и ESR. Для каждого эксперимента применялись модифицированные метрики IS и FID.

В ходе первого эксперимента на размеченном наборе данных MNIST получены удовлетворительные результаты. В ходе обучения GAN качество сгенерированных образцов значительно улучшено. Поведение метрик IS и FID соответствует запланированному в постановленной задаче (4.13): IS увеличивается с ростом качества образцов, а FID – уменьшается. Для FID обнаружена следующая закономерность: увеличение длины вектора признаков H пропорционально увеличивает величину метрики FID, поэтому рекомендуется ее нормирование.

Второй и третий эксперименты показали, что для неразмеченных данных выбор количества кластеров N должен осуществляться в соответствии с условием (5.1). Это позволяет метрике IS соответствовать прогрессу в обучении GAN. Также необходимо отметить, что условием (5.1) задано необязательное ограничение на максимальную величину N, после которой увеличение количества кластеров нецелесообразно с точки роста вычислительной нагрузки. Рекомендуется выбор числа кластеров, приблизительно равный размеру вектора входных данных XN.

Метрика FID, несмотря на внесение необходимых модификаций (4.14), на неразмеченных данных показывает себя неэффективно, является недостаточно точной и объективной. Применение FID в качестве целевой функции при оптимизации GAN также считается нецелесообразным. Обусловлено это высокой дисперсией полученных значений FID по сравнению с первым экспериментом, а также незначительным улучшением FID в процессе обучения.

Подведем итог практическому применению разработанного метода оценки произвольных GAN. В настоящее время такие популярные метрики, как IS и FID, применяются исключительно для оценки качества сгенерированных изображений. Проведенная модификация метрик IS и FID позволила использовать их для различных наборов дан-

ных, а не только изображений. В ходе экспериментальных исследований доказана эффективность разработанного метода для размеченных данных. Для неразмеченных данных модифицированная метрика IS также может использоваться для оценки качества GAN. Метрика FID нуждается в улучшении и доработке, что будет являться предметом дальнейших исследований.

5.3. ПРАКТИЧЕСКАЯ РЕАЛИЗАЦИЯ НЕЙРОСЕТЕВЫХ ГЕНЕРАТОРОВ ИНФОРМАЦИИ

Одним из важных компонентов нейросетевой архитектуры АИС являются генераторы данных, функционирующие на основе машинного обучения. Рассмотрим практическую реализацию таких генераторов, выполненных в соответствии с представленным ранее нейросетевым методом автоматической генерации данных.

На первом этапе практических исследований осуществлена программная реализация нейросетевого метода автоматической генерации данных и алгоритмов задач *K1 – K4* в виде программных классов *GTask1-GTask4* соответсвенно. Для разработки использовался язык программирования Python и библиотека по работе с машинным обучением Keras.

5.3.1. Исследование эффективности нейросетевого метода генерации информации

Рассмотрим специфику реализации классов *GTask1, …, GTask4* и оценим эффективность применения предлагаемого нейросетевого метода для каждого типа задач. Для реализации нейронных сетей использовалась библиотека Keras. Для оценки точности и определения функции потерь нейронных сетей использовались встроенные в библиотеку программные методы. Для обучения сетей применяются известные датасеты.

GTask1. Для решения задачи получения сжатого представления объекта используется нейронная сеть класса AutoEncoder. Для оценки метода использовались датасеты MNIST (60 000 изображений размером 28x28 пикселей), HEC (Individual household electric power consumption, выборка 40 000 записей с 7 значениями, отражающими измерение потребления электроэнергии в одном домохозяйстве с частотой выборки в одну минуту в течение 4 лет, данные взяты без учета даты и времени [163]), Adult (48 842 записи с 14 значениями [164]), HAR (10 299 векторов размерностью 561 значение, содержит сведения о человеческой деятельности с использованием набора данных смарт-

фонов [161]), Treadmill (собранные коллективом 3675 записей с 9 значениями, описывающими координаты трех трекеров, закрепленных на человеке, при движении на беговой дорожке). При решении задачи *K1* необходимо осуществить сжатие исходного объекта, а затем восстановить его. Результаты для каждого датасета представлены в табл. 5.4 в виде исходных и восстановленных двумя методами данных. Будем использовать следующие обозначения:

NNGE – обучение нейронной сети с использованием нейросетевого метода автоматизированной генерации данных;

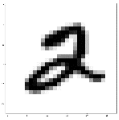
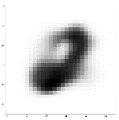
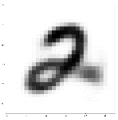
DEF – обучение нейронной сети с настройками по умолчанию;

H – длина сжатого состояния;

T – время поиска структуры сети;

L_0, *L* – значение функции потерь (ошибки) до и после применения метода.

5.4. Применение нейросетевого метода для решения задачи *K1*

Датасет	Исходный объект	DEF	NNGE	Параметры поиска
MNIST				H = 10 T = 478 L_0 = 0.15 L = 0.11
HEC	[4.46, 0.13, 234.6, 19.0, 0.0, 37.0, 16.0]	[2.97, 0.33, 238.6, 12.41, 3.52, 20.0, 19.1]	[4.7, 0.02, 236.2, 20.2, 0.0, 32.8, 16.5]	H = 1 T = 550 L_0 = 0.03 L = 0.005
Adult	[0.35, 2.0, 0.15, 5.0, 0.6, 1.0, 10.0, 1.0…]	[0.35, 2.0, 0.11 3.13, 0.67, 0.85, 4.75, 0.56…]	[0.33, 2.3, 0.13, 4.60, 0.64, 0.82, 6.51, 1.20 …]	H = 1 T = 594 L_0 = 0.027 L = 0.017
HAR	[0.03, 0.0, -0.02, -0.2, -0.36, -0.6, …]	[0.09, 0.02, -1.0, -0.29, -0.38, -0.63, …]	[0.03, 0.05, 0.02, -0.13, -0.34, -0.54, …]	H = 30 T = 197 L_0 = 0.018 L = 0.005
Treadmill	[-0.21, 1.24, -0.16, -0.1, 0.38, -0.07, -0.24, 0.33, -0.31]	[-0.29, 1.23, -0.16, -0.19, 0.34, -0.28, -0.25, 0.36, -0.27]	[-0.19, 1.23, -0.16, -0.09, 0.35, -0.15, -0.25, 0.35, -0.38]	H = 1 T = 119 L_0 = 0.11 L = 0.06

Полученные результаты в виде сокращения ошибки на 26…83% (в среднем на 52%) подтверждают эффективность применения нейросетевого метода.

GTask2. Для решения задачи получения подобных объектов используется нейронная сеть типа GAN. Для GAN процедура обучения имеет свои особенности: в случае если обучение генератора и дискриминатора несбалансированное (на основе анализа метрик), то процесс обучения перезапускается. Кроме того, помимо использования функции потерь «бинарная классификация» мы применяем такие оценки, как Fréchet Inception Distance (FID) и Inception Score (IS). Они позволяют получить более объективную оценку качества генерируемых объектов [165]. Определение оценок осуществляется по общепринятым формулам на основе 20 000 сгенерированных изображений, однако для использования оценок для различных наборов данных, в том числе численных, осуществлены соответствующие модификации, изложенные ниже.

При определении IS мы используем заранее обученный классификатор, распределяющий исходные данные по заданным классам. Это ограничивает возможность использования оценки IS для различных наборов данных, не имеющих разметки по классам. Возможным решением задачи является кластеризация данных с помощью методов машинного обучения (например, KMeans [166]) и использование обученной сети при расчете IS.

При определении FID используется отдельно обученный на исходном наборе данных автоэнкодер *AE*, позволяющий по скрытому слою получить оценку сгенерированных и исходных данных. Использование уникального автоэнкодера для каждого набора данных позволяет получить FID для любых данных, а не только изображений. Так, например, использование предобученной сети InceptionV3 возможно только для оценки сгенерированных изображений, а не любых массивов данных.

Для оценки метода использовались датасеты MNIST, Treadmill, Adult, CIFAR1, CIFAR-10 (для 1-го и 10-го классов соответственно). При решении задачи К2 также осуществлялась контрольная оценка сгенерированных данных по метрикам IS и FID, а также визуальное сравнение полученных объектов. Результаты для каждого датасета представлены в табл. 5.5.

Величина функции потерь для рассмотренных датасетов была снижена в 1, 2–3 раза в зависимости от набора данных. Средний прирост IS после применения нейросетевого метода составил 83%, FID сократился на 44,3%.

5.5. Применение нейросетевого метода для решения задачи *K2*

Датасет	Время поиска	IS/FID		Пример сгенерированного объекта	
		DEF	NNGE	DEF	NNGE
MNIST	1437	5.78 5.6e-04	7.74 1.1e-04		
Treadmill	1941	1.67 5.15	5.27 6.00	[-0.43, 1.31, -0.86…]	[-0.46, 1.02, -0.98…]
Adult	3913	3.96 10.54	6.87 9.24	[-0.9, -7.73, -0.9,-14.1…]	[1.0, -8.0, -1.0, 15.0…]
CIFAR1	1325	1 195	1 15.8		
CIFAR-10	22703	4.37 131	4.8 60.6		

Так как CIFAR1 основан на изображениях одного класса, то IS = 1 является корректным значением.

Разработанный метод достаточно хорошо показывает себя при небольшой размерности данных, однако при росте размерности и увеличения классов данных (в случае CIFAR-10) качество работы метода снижается. Возможным решением может стать параллельное обучение нескольких GAN-сетей для каждого класса.

Необходимо отметить, что использование метрик FID и IS до сих пор не дает однозначной и объективной оценки. Например, по мнению авторов работы [167], данные метрики зависимы не только от содержимого данных выборки, но и от ее размера.

GTask3. Для решения задачи получения восстановления данных используется ансамбль регрессионных нейронных сетей, для каждого сочетания известных параметров, прогнозирующий неизвестные.

Для оценки метода использовались датасеты Adult, HEC и HAR. Для набора Adult проверялось удаление 7% (1 из 14) значений из набора данных, для HEC – удаление 28% (2 из 7), 57% (4 из 7) значений, для HAR – 10% (56 из 561), 20% (102 из 561) и 50% значений (280 из 561). Результаты для каждого датасета представлены в табл. 5.6.

5.6. Применение нейросетевого метода для решения задачи *К3*

Датасет (процент удаленных данных)	Время поиска структуры нейронной сети	Ошибка		Точность	
		DEF	NNGE	DEF	NNGE
Adult (7%)	320	0.136	0.1059	0.805	0.849
HEC (28%)	154	0.0048	0.0044	0.96	0.97
HEC (57%)	257	0.023	0.02	0.78	0.82
HAR (10%)	85	0.52	0.007	0.81	0.88
HAR (20%)	74	0.52	0.008	0.67	0.86
HAR (50%)	49	0.51	0.015	0.24	0.41

Средний прирост точности после применения нейросетевого метода составил 20% (от 1 до 71%). Ошибка в среднем сократилась на 56% (от 8 до 98%).

GTask4. Для решения задачи прогнозирования состояния объекта используется регрессионная нейронная сеть с применением слоев типа LSTM.

Для оценки метода использовались датасеты Individual household electric power consumption (HEC), Treadmill и HAR accelerometer (40 000 записей по 3 значения, определяющих ускорение акселерометра при движении человека [161]). При решении задачи *К4* будем рассматривать историю объекта $q = 5$ (т.е. для прогнозирования следующего состояния будут использоваться пять предыдущих). Результаты для каждого датасета представлены в табл. 5.7.

Ошибка сократилась в среднем на 73%. Точность после применения метода либо осталась на прежнем высоком уровне, либо возросла.

5.7. Применение нейросетевого метода для решения задачи *К4*

Датасет (процент удаленных данных)	Время поиска структуры нейронной сети	Ошибка		Точность	
		DEF	NNGE	DEF	NNGE
HEC	1142	20.6	16	0.99	0.999
HAR accelerometer	868	32.2	0.58	0.33	0.929
Treadmill	514	2252	10.6	0.99	0.99

Таким образом, проведенные исследования показывают эффективность применения нейросетевого метода на ряде рассмотренных датасетов в большинстве случаев.

5.3.2. Апробация и сравнение нейросетевого метода генерации данных с существующими решениями

Для проверки адекватности предлагаемого нейросетевого метода автоматической генерации данных проведем сравнительный анализ с существующими подходами к автоматическому формированию нейронных сетей. В качестве альтернативных решений рассмотрим библиотеки AutoKeras [130], DEvol [131], AutoSklearn [133], AutoGAN [134]. В таблице 5.8 представлены сравнительный анализ функциональных возможностей библиотек в сравнении с предлагаемым методом.

5.8. Сравнение существующих подходов по автоматическому формированию нейронных сетей

Признак	AutoKeras	DEvol	AutoSklearn	AutoGAN	NNGE
Классификация данных	1	1	1	0	1
Регрессия	1	1	1	0	1
Генерация новых данных заданного класса	0	0	0	1	1
Сжатие состояния объекта	0	0	0	0	1
Прогнозирование данных	1	1	1	0	1
Восстановление отсутствующих значений	1	1	1	0	1
Преобразование данных	1	1	0	1	1
Автоматическое определение класса задач	1	0	0,5	0	1
Подбор структуры нейронной сети	1	1	1	1	1
Многомерность векторов X и Y	0	1	0	1	1
Итого	7	7	5,5	4	**10**

Проведенный анализ позволил сделать следующие выводы. AutoKeras не поддерживает многомерные входные данные, а только одномерный вектор. Выходные данные могут быть представлены единственным значением, что приводит к необходимости создания нескольких выходов для представления векторов. Это приводит к значительным трудностям при обработке данных и ограничивает применяемость данной библиотеки без внесения существенных корректив в структуру исходных данных, которые приходится осуществлять вручную.

DEvol требует ручной настройки функции потерь, иначе понижается эффективность библиотеки при решении задач регрессии.

AutoSklearn также не поддерживает многомерные входные и выходные данные. Кроме того, невозможно формирование нескольких независимых входов или выходов.

Вышеперечисленные библиотеки не поддерживают работу с нейронными сетями типа GAN или AutoEncoder, что означает для них невозможность решать задачи типа *К1* и *К2*.

В отличие от них библиотека AutoGAN реализует подбор структуры GAN-сети. Она показывает высокую эффективность при генерации данных на основе датасетов CIFAR10 и STL-10, однако ее архитектура крайне тяжело адаптируется под другие наборы данных. Особенно это касается данных, представленных в виде многомерных численных значений, а не изображений, так как средства оптимизации структуры нейронной сети в данной библиотеке основаны на Inception Score (IS) и использовании модели Inception V3. Это значительно ограничивает использование данной библиотеки для решения задачи *К2*. Остальные задачи данная библиотека не поддерживает.

Из таблицы 5.8 следует, что провести полное сравнение методов для каждой задачи (*К1 – К4*) невозможно ввиду отсутствия возможности решения отдельных задач в сравнимых методах. В ходе сравнительного анализа решено было остановиться на следующих тестах.

В первом тесте сравниваются классифицирующие возможности методов, так как они в полной мере реализованы во всех подходах (кроме AutoGAN). В нейросетевом методе для моделирования задачи классификации будем использовать задачу *К4*, где в качестве выхода зададим не $(i + 1)$-е значение, а метку категории. Вторым вариантом имитации задачи классификации является решение задачи *К1*, в которой обучающий набор состоит из пар «объект–класс», тогда используемый в основе метода автоэнкодер будет работать как классификатор. Результат теста 1 представлен в табл. 5.9.

5.9. Тест на классификацию объектов (dataset MNIST)

	Время поиска структуры	Ошибка	Точность
AutoKeras	2162	**0.003**	**0.998**
DEvol	3434	0.03	0.990
AutoSklearn	1897	0.57	0.9695
NNGE (K4)	**192**	**0.003**	0.981
NNGE (K1)	**63**	**0.003**	0.983

По итогам первого теста AutoKeras показывает наилучшую точность классификации. Однако все методы показали достаточно высокие результаты, а среднеквадратичная ошибка для AutoKeras и NNGE в обоих вариантах сравнима. Время поиска оптимальной структуры и обучения сети у NNGE значительно меньше.

Во втором тесте мы исследовали возможности рассматриваемых программных методов в задаче восстановления потерянных данных. Для этого используется набор данных Adult, из которого было удалено 1 значение, после чего нейронные сети должны восстановить исходные значения. Успешным считается отклонение не более 5% от искомой величины. Далее использовался набор HAR, включающий 561 значение о координатах положения человека. При использовании датасета HAR мы провели три эксперимента: с удалением 10% значений (56 из 561), 20% значений (102 из 561) и 50% значений (280 из 561).

На основе анализа методов (табл. 5.10) получено, что AutoKeras и AutoSklearn по умолчанию не поддерживают многомерные выходы при решении задачи регрессии и требуют значительной переработки данных, поэтому они не будут участвовать в данном сравнении.

В первом эксперименте при поиске 1 потерянного значения в датасете Adult лучший результат по точности показал AutoKeras, однако разработанный метод NNGE имеет значительное превосходство по времени обучения сети и практически сравним по полученной среднеквадратичной ошибке.

Далее в остальных экспериментах мы сравниваем NNGE с DEvol. При 10 и 20% потерянных значений метод NNGE показывает лучшую точность и время обучения нейронной сети. При 50% DEvol имеет лучшую точность, однако время обучения и среднеквадратичная ошибка по-прежнему лучше у NNGE.

5.10. Тест на восстановление потерянных значений

	Время поиска структуры	Ошибка	Точность
dataset Adult, 1 потерянное значение			
AutoKeras	954	**0.092**	**0.869**
DEvol	570	0.38	0.82
AutoSklearn	1121	0.16	0.836
NNGE	**320**	0.1059	0.849
dataset HAR, 10% потерянных значений			
DEvol	196	0.53	0.86
NNGE	**85**	**0.007**	**0.88**
dataset HAR, 20% потерянных значений			
DEvol	195	0.52	0.84
NNGE	**74**	**0.008**	**0.86**
dataset HAR, 50% потерянных значений			
DEvol	191	0.53	**0.51**
NNGE	**49**	**0.015**	0.41

Третий эксперимент заключается в решении задачи *K4* – прогнозировании состояния объектов на основе анализа предыдущих состояний. В качестве исходных данных мы рассматриваем два датасета: Individual household electric power consumption (HEC) и HAR accelerometer. Оба датасета представляют собой временную последовательность, т.е. существуют закономерности в изменении значений. Для анализа будем использовать длину истории $q = 5$, т.е. для прогноза следующей записи будут использоваться пять предыдущих. Многомерность входных векторов означает невозможность полноценного использования методов AutoKeras и AutoSklearn.

Результаты решения задачи прогнозирования *K4* представлены в табл. 5.11.

В тесте 3 (табл. 5.11) получено, что точность обоих методов сравнима и находится в пределах погрешности измерения. Время обучения сети у метода NNGE ниже. Среднеквадратичная ошибка у метода DEvol выше.

5.11. Тест на прогнозирование состояний объектов

	Время поиска структуры	Ошибка	Точность
dataset HEC			
DEvol	2030	83612	0.997
NNGE	**1142**	**16**	**0.999**
dataset HAR accelerometer			
DEvol	1566	32.2	**0.948**
NNGE	**868**	**0.58**	0.929

Четвертый тест заключается в исследовании возможности решения задачи *K2*. Однако при проведении данного эксперимента возникло несколько существенных проблем. Во-первых, метод AutoGAN направлен на генерацию изображений из двух заданных датасетов и применить его для решения задачи генерации численных данных не представляется возможным. С другой стороны, генерация изображений в рамках задачи *K2* – это только одно из возможных направлений, но не основное. В АИС больший интерес представляет генерация многомерных численных или текстовых данных, но не изображений. Поэтому последний тест несет больше теоретический характер. Его результаты представлены в табл. 5.12.

В данном тесте метод AutoGAN имеет явное превосходство по качеству генерируемых изображений. Для NNGE наращивание итераций и дальнейший поиск структуры не дал лучших результатов.

Проиллюстрируем результаты, представленные в табл. 5.9 – 5.12, сводной диаграммой сравнения средней точности, в процентах (рис. 5.8), и среднего времени поиска структуры нейронной сети, в секундах (рис. 5.9), для всех решаемых задач. Из графиков исключены результаты задачи *K*2 и, следовательно, метода AutoGAN. Также рассматривался только первый эксперимент теста восстановления данных (табл. 5.8), так как в остальных не принимали участие другие методы.

5.12. Тест на генерацию подобных объектов

	Время поиска структуры	IS	FID
AutoGAN	От 48 до 72 часов	8.55 ± .10	12.42
NNGE	От 6 до 10 часов	4.7 ± .2	60.6

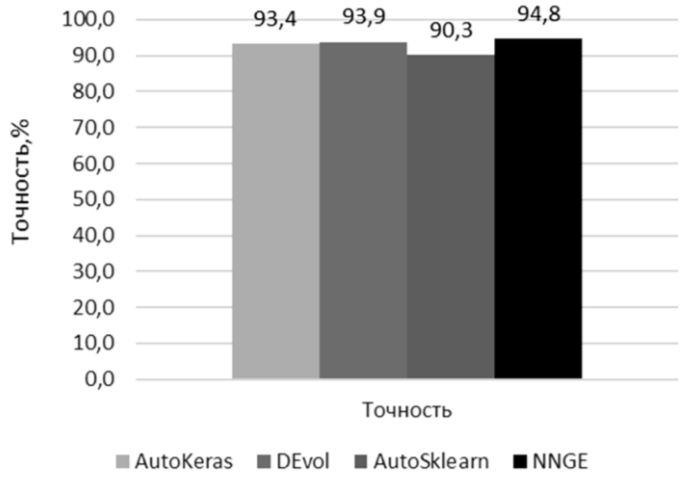


Рис. 5.8. Сравнение точности решения задач генерации данных

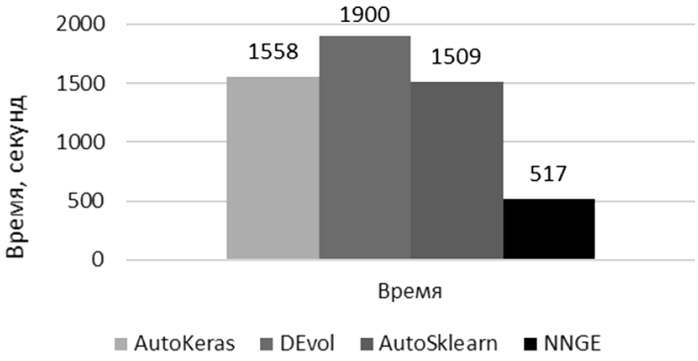


Рис. 5.9. Сравнение времени решения задач генерации данных

Таким образом, средняя точность нейросетевого метода составила 94,8%, а среднее время решения – 517 секунд. Средняя точность всех методов равна 92,5% (нейросетевой метод точнее в среднем на 2,3%). В среднем время решения задач остальными методами равнялось 1655 секундам (нейросетевой метод быстрее в 3,2 раза).

В заключении сравнительного анализа нейросетевого метода генерации данных осуществим оценку сложности его реализации. В ка-

честве второго метода будем использовать DEvol, так как он применялся при решении большинства задач (кроме *K2*). Кроме того, методы AutoGAN, AutoKeras и AutoSklearn – это целые библиотеки программного кода, в отличие от DEvol. Объем и сложность программного кода в них количественно значительно выше.

В таблице 5.13 представлены итоговые результаты сравнения. Использовались следующие метрики: циклометрическая сложность (общая *Sum* (*CC*) и средняя *Average* (*CC*)), метрики Холстеда и Джилба.

Таким образом, имеем следующие результаты. Средняя циклометрическая сложность нейросетевого метода генерации данных чуть ниже, однако общая – выше. Аналогично обстоят дела и с метрикой Холстеда – в нейросетевом методе больше операторов, переменных, что ведет к увеличению объема кода и его сложности, времени реализации. Однако необходимо учитывать то, что в методе DEvol отсутствует решение достаточно сложной задачи *K2*. Если исключить программный код, ответственный за ее решение, то сложность нейросетевого метода снизится приблизительно на 20% (по метрике сложности Холстеда), а время реализации значительно сократится (до 3 раз). Метрика сложности по Джилбу во всех случаях сравнима.

Графики сравнения двух методов представлены на рис. 5.10. За 100% (эталон) взяты значения метрик метода DEvol.

Таким образом, можно сказать, что сложности реализации методов DEvol и NNGE сопоставимы, однако в ряде задач нейросетевой метод генерации данных показывает большую производительность и точность. Кроме того, он отличается большей универсальностью и возможностью решения широкого спектра задач по генерации данных.

5.13. Сравнение DEvol и нейросетевого метода генерации данных

Критерий	Классический метод	Нейросетевой метод	Нейросетевой метод (без решения *K2*)
CC	*Sum* (*CC*) = 119 *Average* (*CC*) = = A(3.97)	*Sum* (*CC*) = 272 *Average* (*CC*) = = A(3.48)	*Sum* (*CC*) = 207 *Average* (*CC*) = = A(3.13)
HAL	*h1*: 20, *h2*: 155, *N1*: 120, *N2*: 230	*h1*: 18, *h2*: 359, *N1*: 347, *N2*: 573	*h1*: 15, *h2*: 216, *N1*: 196, *N2*: 329
J	0.39	0.40	0.43

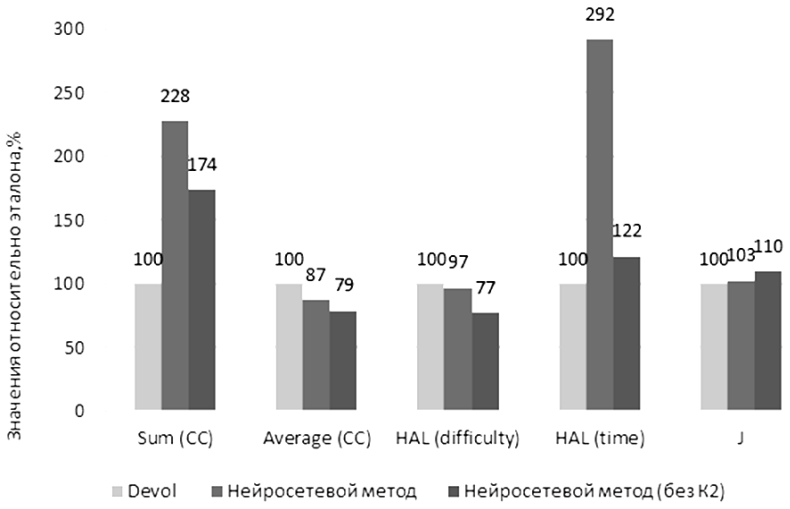


Рис. 5.10. Сравнение эталона (DEvol) с нейросетевым методом генерации данных

Для нейросетевого метода автоматической генерации данных не проводилось его сравнение по метрикам сложности программной реализации с аналитическими подходами, так как для задач генерации они в большинстве ситуаций нереализуемы либо являются значительно более сложными и трудоемкими (необходимость вывода регрессионных функций, получение аналитических решений и т.д.).

Подведем итоги. Полученные результаты подтверждают улучшение точности работы нейронных сетей после применения нейросетевого метода в большинстве случаев. Однако проведенные эксперименты показали необходимость развития и доработки метода. В ходе дальнейшей работы планируется расширить функциональность нейросетевого метода, количество поддерживаемых архитектур нейронных сетей и типов слоев, внедрить интеллектуальные алгоритмы поиска оптимальной структуры. Испытания на ряде датасетов показывают недостаточную гибкость и эффективность метода при генерации сложных структур данных (например, цветных изображений), что может быть исправлено при решении вышеперечисленных задач.

Далее осуществлено сравнение разработанного метода с существующими подходами к автоматическому поиску оптимальных структур нейронных сетей. Проведенные экспериментальные исследования показали, что нейросетевой метод автоматической генерации

данных NNGE превосходит большинство методов (AutoKeras, DEvol, AutoSklearn, AutoGAN) по времени нахождения оптимальной структуры нейронной сети за счет ориентирования на конкретные классы задач генерации, а также обеспечивает сравнимый уровень точности. Метод NNGE является более универсальным при решении задач генерации информации.

Полученные практические результаты подтверждают применимость нейросетевого метода автоматической генерации данных для решения ряда задач в рамках АИС. Изложенные теоретические основы и алгоритмы могут использоваться для разработки более эффективных программных решений в области автоматической генерации данных.

5.4. АПРОБАЦИЯ И ОЦЕНКА ЭФФЕКТИВНОСТИ НЕЙРОСЕТЕВЫХ КОМПОНЕНТОВ ПЕРЕАДРЕСАЦИИ ИНФОРМАЦИИ

Рассмотрим процесс реализации метода автоматической переадресации информации. Разработанное программное обеспечение обозначим нейросетевым каналом переадресации информации.

На первом этапе осуществляется формализация основных объектов и субъектов предметной области: исполнителей и операций. Для операций формируется матрица приоритетов (с привлечением экспертной группы). Разработчики проектируют соответствующую структуру базы данных с необходимым набором таблиц.

На втором этапе разрабатывается программное обеспечение для сбора информации о проведенных исполнителями операциях. Собранная информация сохраняется в базу данных.

На третьем этапе при использовании разработанных программных методов сбора данных осуществляется сохранение необходимой информации (при работе АИС в штатном режиме). Данные включают идентификатор пользователя, идентификатор операции, время ее выполнения и факт успешного завершения. В рамках тестового эксперимента была сформирована тренировочная выборка для 100 пользователей и 5 операций общим объемом 5000 записей (5 попыток выполнить операцию для каждого пользователя). Такой объем можно считать достаточным, если рассматривать каждую операцию как признак: тогда для каждой операции задано 1000 элементов тренировочной выборки, что является достаточным объемом (согласно статье [160] необходимо не менее 50 элементов на признак).

Четвертый этап включает формирование структуры и обучение нейронной сети. Входной слой представлен 6 нейронами: идентификатор, возраст и квалификация (от 0 до 1) пользователя, идентификатор,

количество действий и уровень сложности (от 0 до 1) операции; выходной – двумя для времени операции и вероятности успеха (от 0 до 100) ее выполнения. Также в структуру добавлено 5 скрытых слоев по 200 нейронов в каждом и 2 слоя Dropout. Точность определения факта успешности и времени выполнения операции составила 96%. Полученные результаты можно считать удовлетворительными.

Далее осуществляется интеграция в АИС сервиса на основе фреймворка Flask. Данный сервис, обменивающийся с АИС данными по протоколу HTTP, позволяет интегрировать реализуемый метод в АИС на основе любой платформы, языка программирования или фреймворка. В разработанный сервис подключается обученная нейронная сеть.

В соответствии с разработанным ранее алгоритмом далее АИС получает информацию о новых задачах и операциях, которые передаются в сервис и, следовательно, в нейронную сеть. Последняя оценивает множество пользователей на возможность эффективного выполнения поставленной задачи и формирует подмножество оптимальных исполнителей $U_C \in U$. Для каждого исполнителя из подмножества U_C осуществляется анализ загруженности и расчет времени выполнения операции. На основе полученных значений выбирается исполнитель, способный выполнить операцию за наименьшее время.

В качестве эталона, с которым будем сравнивать эффективность работы нейросетевого метода переадресации, используем программное обеспечение, решающее поставленную задачу определения загруженности и времени выполнения. Однако в данном алгоритмическом обеспечении будем использовать не нейронные сети, а статистическую обработку данных.

Тестирование проводилось на 4500 сгенерированных данных, соответствующих по структуре тренировочному набору, но имеющих небольшие отклонения в значениях. Результаты точности предсказания успешности и времени выполнения операции представлены на рис. 5.11.

Под максимальной точностью в данном эксперименте понималось минимальное среднее отклонение полученных в методе значений успешности и времени выполнения операций от фактических. Нейросетевой метод обеспечил следующую точность: 90,8 и 70,5% по метрикам успешности и времени выполнения. Для классического метода точность составила соответственно 88,5 и 48,8%. Таким образом, нейросетевой метод показывает большую гибкость перед статическим подходом, применяемым в классической реализации задачи переадресации.

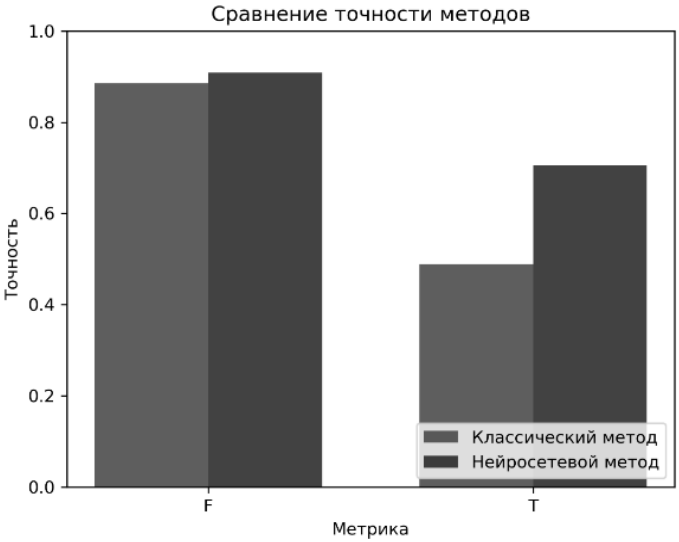


Рис. 5.11. Сравнение точности классического и нейросетевого методов переадресации данных

Далее рассмотрим оценку сложности реализации нейросетевого метода переадресации по усредненной точности $(F+T)/2$, циклометрической сложности, метрикам Холстеда и Джилба (табл. 5.14).

Разработанный метод автоматической переадресации также показывает свою эффективность – обеспечивается более низкая сложность реализации и высокая точность.

5.14. Сравнение классического и нейросетевого метода переадресации данных

Критерий	Классический метод	Нейросетевой метод
P	68.6	80.6
CC	*Sum* (*CC*) = 18 *Average* (*CC*) = A (3.6)	*Sum* (*CC*) = 26 *Average* (*CC*) = A (1.65)
HAL	*h1*: 8 *h2*: 55 *N1*: 38 *N2*: 75	*h1*: 7 *h2*: 36 *N1*: 26 *N2*: 51
J	0.13	0.09

5.5. АПРОБАЦИЯ И ОЦЕНКА ЭФФЕКТИВНОСТИ НЕЙРОСЕТЕВЫХ КОМПОНЕНТОВ КЛАССИФИКАЦИИ И РАСПРЕДЕЛЕНИЯ ИНФОРМАЦИИ

Для реализации нейросетевого метода классификации и распределения данных в АИС будем использовать программное обеспечение на основе Python и библиотеки Keras. Разработка осуществлялась в соответствии с нейросетевой архитектурой и понятием нейросетевых каналов данных. Обозначим реализацию предложенного метода как нейросетевой канал распределения данных. Под этим понятием будем понимать следующее: программный интерфейс, функционирующий на основе одной или нескольких нейронных сетей для классификации данных, осуществляющий анализ поступающей от пользователя информации и распределяющий ее по заданным категориям. Является каналом первой степени, так все нейронные сети используются только для классификации данных. Однако для решения задачи распределения данных в состав нейросетевого канала включен алгоритмический блок проверки условий, изложенных в нейросетевом методе классификации и распределения данных. Рассмотрим его реализацию.

В примере будет анализироваться форма карточки документа с 5 полями: наименование документа; автор документа; контактная информация (адрес) автора; дата создания; описание документа. В ходе экспериментального исследования будет моделироваться ввод данных пользователей в форму и заполнение информации в указанные 5 полей. Нейронная сеть будет классифицировать введенную информацию и проверять ее на соответствие заданной категории.

Для формирования набора исходных данных используем генератор на основе открытых баз русских имен, городов и стран, а также последовательностей из русских слов и цифр. Это позволит получить конструкции, соответствующие реальным данным, но не хранящие персональных данных реальных людей. Текстовые данные будут обработаны с помощью токенизатора для перевода их в численный формат. Максимальная длина предложения ограничена 20 словами.

Модель нейронной сети классификатора имеет следующий вид: входной слой, количество нейронов в котором равно максимальной длине предложения (20 слов), Embedding-слой, слой LSTM и выходной слой с 6 нейронами (количество категорий плюс одна для ошибочных вариантов). Итоговая точность нейронной сети на контрольном наборе после 5 эпох составила 97,8%.

Осуществим итоговую проверку адекватности работы нейронной сети при классификации и распределении данных. Проведем сравне-

ние сложности реализации и точности нейросетевого метода с классическим подходом, в котором использовалось алгоритмическое обеспечение, анализирующее синтаксические конструкции текста.

Для этого был сформирован новый контрольный набор, разбитый на 5 категорий: наименование документа; автор документа; адрес автора; дата создания; описание документа. Далее определено количество ошибок при распознавании категорий (рис. 5.12). Тест подтвердил эффективность нейросетевого метода: ошибки допущены всего в одной категории («Адрес») в размере 2%, остальные категории распознаны полностью правильно.

Для классического метода получены следующие результаты: 16% ошибок в категории «Наименование» и 25% – в «Описание», что существенно хуже показателей нейросетевого метода.

Далее рассмотрим оценку сложности реализации нейросетевого метода классификации и распределения данных по усредненной точности (на основе результатов с рис. 5.12), циклометрической сложности, метрикам Холстеда и Джилба (табл. 5.15).

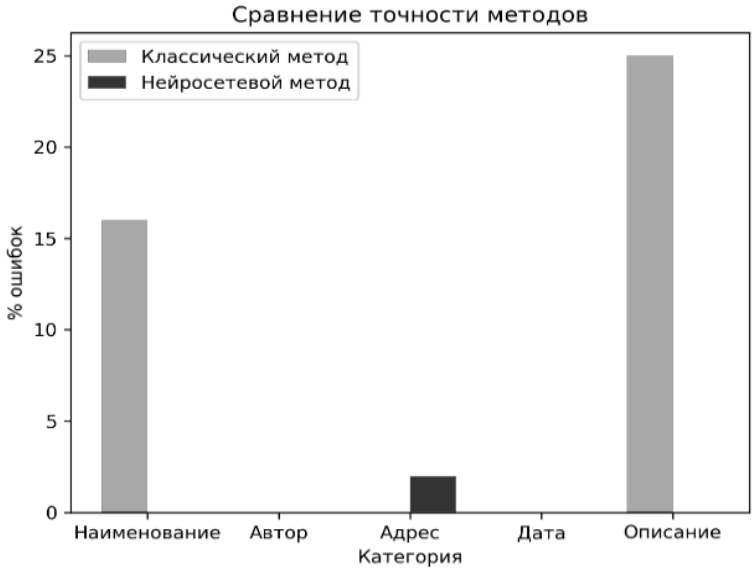


Рис. 5.12. Сравнение точности нейросетевого и классического методов классификации и распределения данных

5.15. Сравнение классического и нейросетевого методов классификации и распределения данных

Критерий	Классический метод	Нейросетевой метод
P	91.8	99.6
CC	*Sum* (*CC*) = 26 *Average* (*CC*) = A (3.71)	*Sum* (*CC*) = 22 *Average* (*CC*) = A (1.37)
HAL	*h1*: 3, *h2*: 15, *N1*: 8, *N2*: 16	*h1*: 5, *h2*: 13, *N1*: 7, *N2*: 14
J	0,8	0.15

Получаем, что сложность по Холстеду нейросетевого метода на 30…40% выше, однако по другим метрикам (цикломатической сложности и Джилба) имеем преимущество нейросетевого метода. Точность его также выше почти на 8%. Стоит заметить, что сложность реализации и классического, и нейросетевого метода классификации и распределения данных относительно невелики, так что их можно считать сравнимыми. Однако полученные результаты по точности классификации подтверждают большую эффективность нейросетевого подхода.

5.6. АПРОБАЦИЯ И ОЦЕНКА ЭФФЕКТИВНОСТИ НЕЙРОСЕТЕВОГО МЕТОДА УПРАВЛЕНИЯ

Для апробации нейросетевого метода управления использовалась тестовая система управления по определению положения информационного объекта в одномерном пространстве (имитация перемещения человека по беговой дорожке с двумя направлениями). В данной системе необходимо корректировать скорость движения пространства в соответствии с положением объекта на ней.

Для системы выбраны следующие параметры: размеры пространства от –1 до 1; скорость движения пространства от –3 до 3. Начальное положение объекта в нуле. На каждой итерации объект делает случайное перемещение в одном из направлений, что приводит к изменению его координаты. Пространство смещается в соответствии с этим сдвигом, возвращая объект в начальное положение. Замеры осуществляются дискретно через равные промежутки времени.

На первом этапе сформулирован эталон – набор исходных данных для обучения нейронной сети. В эталоне на каждом шаге проверяется положение объекта на пространстве, после чего скорость увеличивается (если объект в положительной половине пространства) либо уменьшается. Уменьшение осуществляется с шагом 1.

Для системы управления заданы действия по изменению скорости пространства. Всего задано 7 скоростей (от –3 до 3). Имеется возможность мгновенного переключения скорости, таким образом, количество возможных действий составляет 13 (изменения от –6 до +6).

Классический подход заключается в реализации линейного закона управления – чем дальше объект удален от некоторой нулевой точки, тем выше скорость. Добавлено несколько ключевых положений для возможности свободного перемещения в определенной безопасной зоне. Скорость округляется до десятых, что в теории обеспечивает высокую плавность ее изменения.

Нейросетевой подход реализует частичное нейросетевое управление на основе трех нейронных сетей для анализа положения объекта, его прогнозирования и выбора текущего действия системы.

Первая нейронная сеть используется для прогнозирования движений объекта и устранения тем самым программной задержки. Вторая нейронная сеть используется для распознавания состояния объекта. Всего задано три состояния: ускоряется, сохраняет текущую скорость, замедляется. Третья нейронная сеть принимает решение о действии системы управления в текущих условиях.

Процесс обучения сетей осуществлялся на 100 000 записей. Первая нейронная сеть показывает точность 98,5%, вторая – 99,7%, третья – 83,6. Данные результаты можно считать приемлемыми, так как даже для последней нейронной сети периодическая ошибка при выборе между соседними действиями не будет иметь значительного эффекта (хотя и сказывается на значении точности).

Далее проведен сравнительный эксперимент между эталоном, классическим и нейросетевым методами управления. На рисунке 5.13 представлено изменение положения объекта в пространстве с течением времени. Графики отражают значения координаты объекта в каждый момент времени при использовании всех трех вариантов управления.

Для оценки качества управления используем следующую метрику: просуммируем модули значений отклонения положения объекта от нулевой позиции и найдем их среднее арифметическое. Чем меньше данная величина, тем меньше колебался объект и тем больше он находился в одном положении, несмотря на собственные перемещения. Тогда получим:

– для эталона: 0.168;
– для классического метода: 0.129;
– для нейросетевого метода: 0.096.

Значения данной метрики будут использоваться в качестве точности в сводной таблице при сравнении методов.

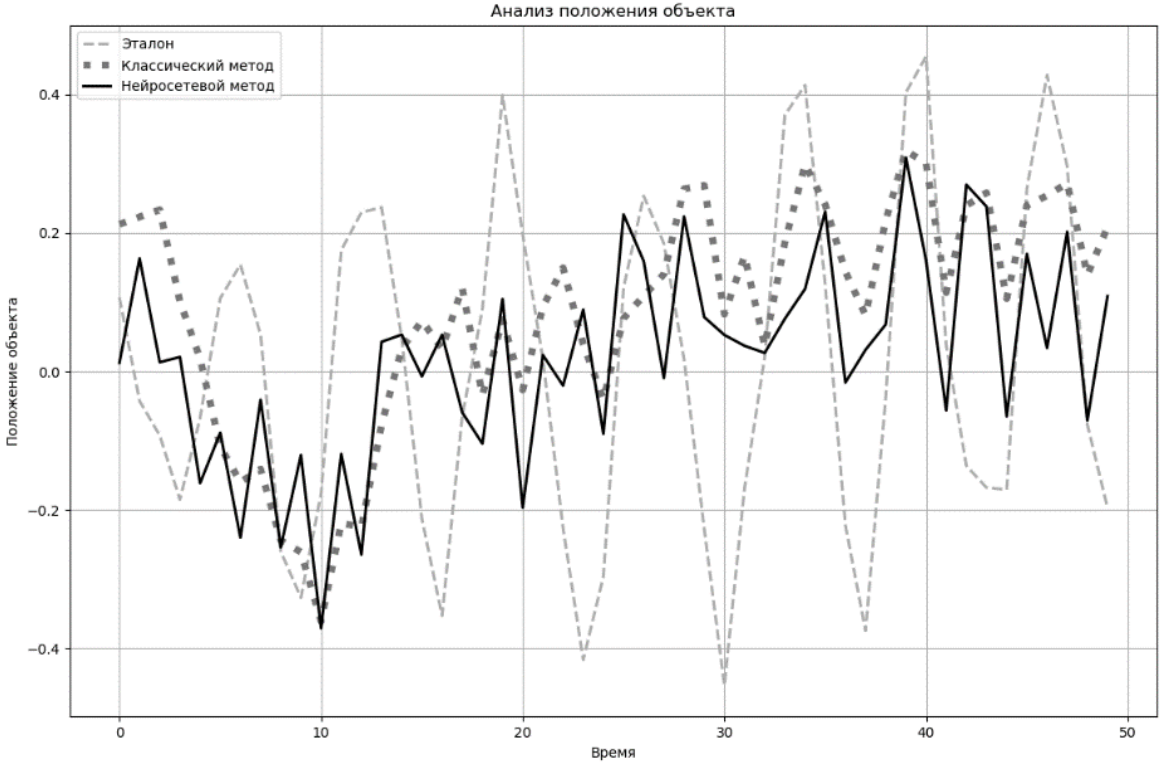


Рис. 5.13. Положение объекта при различных методах управления

5.16. Сравнение классического и нейросетевого методов управления

Критерий	Классический метод	Нейросетевой метод
P	0.129	0.096
CC	*Sum* (*CC*) = 36 *Average* (*CC*) = C (12)	*Sum* (*CC*) = 3 *Average* (*CC*) = A (3.0)
HAL	*h1*: 11 *h2*: 49 *N1*: 49 *N2*: 93	*h1*: 3 *h2*: 15 *N1*: 10 *N2*: 18
J	0.42	0.13

Сравнение сложности реализации нейросетевого метода управления относительно классической реализации представлено в табл. 5.16.

Полученные результаты подтверждают значительное снижение сложности реализации процесса управления после применения нейросетевого метода. Это обусловлено тем, что нет необходимости анализа информационных и разработки алгоритмов, учитывающих все возможные состояния объектов и системы. Нейронные сети самостоятельно аппроксимируют эти зависимости и закономерности, ускоряя процесс разработки. Кроме того, в ряде случаев такой подход приводит к повышению точности работы системы управления за счет более гибкого подхода и возможности прогнозировать состояния системы и объектов.

Таким образом, нейросетевое управление также доказало свою эффективность и перспективность при решении задач управления в АИС, особенно для тех систем, где затруднена формализация законов функционирования объектов.

5.7. ВЫВОДЫ

В главе 5 рассмотрены практическая реализация, апробация и оценка разработанных нейросетевых методов. Для каждого метода представлена структура программного кода, проведены испытания на конкретных задачах и примерах, проанализирована точность и величина ошибки используемых нейронных сетей.

Нейросетевой метод обработки и передачи данных рассмотрен на примере нейросетевого канала передачи информации первой степени. Нейросетевой метод адаптации информационных систем апробирован на примере персонализации интерфейса АИС с помощью нейросетевого канала адаптации. В ходе апробации двух методов получено, что

нейросетевые каналы данных по сравнению с классическими подходами обеспечивают большую точность решения задачи (на 11,8…19,0%), меньшую сложность программной реализации (от 21 до 150% по метрике циклометрической сложности, от 20% по метрике Холстеда), сокращение времени разработки (на 12…32%), сокращение стоимости разработки (от 36 до 63%).

Проведена апробация метода оценки произвольных GAN на нескольких стандартных датасетах, определены возможности и ограничения применения модифицированных оценок IS и FID.

Рассмотрена апробация нейросетевого метода автоматизированной генерации информации на примере решения четырех типов задач: сжатия информации, генерации псевдоподобных объектов, восстановления потерянных данных и прогнозирования состояния объектов. Применение разработанного метода для автоматизации процесса решения задач показывает его значительную эффективность – найденные структуры нейронных сетей имеют большую точность и меньшую ошибку.

Так же проведено сравнение нейросетевого метода генерации с существующими подходами к автоматическому формированию нейронных сетей: AutoKeras, DEvol, AutoSklearn, AutoGAN. В среднем нейросетевой метод показывает на 2,3% большую точность и в 3,2 раза быстрее решает поставленную задачу. По сложности реализации нейросетевой метод сопоставим с конкурентами, однако отличается большей универсальностью и возможностью решения широкого спектра задач по генерации данных.

Апробация нейросетевого метода переадресации информации показала его применимость и эффективность для решения данной задачи: точность определения оптимального исполнителя повысилась в среднем на 15% относительно алгоритмического решения при меньшей сложности реализации по большинству метрик.

Нейросетевой метод классификации и распределения информации был успешно апробирован, показал свою высокую точность. При сравнении с классическими подходами нейросетевой метод оказался более точен (на 8%) при относительно сравнимой сложности реализации.

Нейросетевой метод управления обеспечил увеличение точности (на 34%), а также значительное снижение сложности реализации системы управления за счет автоматизации процесса поиска и анализа закономерностей функционирования информационных объектов.

Таким образом, нейросетевые методы были успешно апробированы, а их эффективность для решения задачи автоматизации проектирования АИС – подтверждена.

Глава 6
АПРОБАЦИЯ МЕТОДОЛОГИИ ПРИ РАЗРАБОТКЕ АДАПТИВНЫХ СИСТЕМ ЭЛЕКТРОННОГО ДОКУМЕНТООБОРОТА

В данной главе рассматривается применение и апробация разработанной методологии и архитектуры, а также нейросетевых методов в предметной области систем электронного документооборота (СЭД). Применение данных подходов позволит осуществить переход от СЭД к адаптивным СЭД, обладающим большими возможностями по персонализации интерфейса, автоматизации при обработке и анализе данных.

6.1. АНАЛИЗ И ФОРМАЛИЗАЦИЯ ПРЕДМЕТНОЙ ОБЛАСТИ АДАПТИВНЫХ СИСТЕМ ЭЛЕКТРОННОГО ДОКУМЕНТООБОРОТА

В соответствии с разработанной методологией структурно-параметрического синтеза АИС на первом этапе необходимо осуществить планирование, включающее системный анализ основных объектов, субъектов и процессов, принадлежащих предметной области СЭД.

Основным объектом в СЭД является документ. Для каждого документа можно выделить множество его состояний, отражающих значения его атрибутов в различные моменты времени.

Субъектами в СЭД являются пользователи системы, осуществляющие множество операций над документами в соответствии с требованиями, заданными внешними или внутренними воздействиями.

Процессы обработки информации включают: классификацию и распределение документов, их поиск, обнаружение дубликатов, переадресацию документов, генерацию типовых документов, адаптацию интерфейса СЭД, поддержку принятия решений в СЭД.

К функциям адаптивности относятся: персонализация интерфейса в зависимости от характеристик оборудования и индивидуальных особенностей пользователей; автоматизация навигации в СЭД на основе предпочтений пользователей.

Сформулируем техническое задание структурно-параметрического синтеза адаптивной СЭД: необходимо разработать программные модули для СЭД, функционирующие на основе технологий машинного обучения, реализующие процессы обработки информации. За счет применения методологии структурно-параметрического синтеза АИС стоимость и сложность реализации модулей должна быть снижена.

На основе проведенного анализа далее реализуем второй этап методологии структурно-параметрического синтеза АИС (формализацию предметной области), в ходе которого разработаем теоретико-графовую модель адаптивной СЭД, формализуем в виде функциональных диаграмм процессы обработки информации, разработаем нейросетевую архитектуру адаптивной СЭД и поставим формализованную задачу ее структурно-параметрического синтеза.

6.1.1. Разработка математической модели предметной области адаптивной системы электронного документооборота

Перед реализацией программного обеспечения адаптивной СЭД необходимо формализовать основные объекты и субъекты предметной области, а также закономерности их взаимодействия. В качестве аппарата для формализации предметной области адаптивной СЭД будем использовать теорию множеств и теорию графов, а также метод формализации информационных потоков на основе моделей многоуровневых графов. Обозначим ее как модель структуры электронного документооборота *S*.

Под структурой электронного документооборота организации S будем понимать формализованное представление предметной области в виде кортежа с множеством объектов, меняющих свои состояния в результате проведения операций множеством пользователей в некоторые моменты времени под влиянием множества внешних и внутренних воздействий:

$$S = (U,\ P,\ RE,\ O,\ T,\ G), \qquad (6.1)$$

где $U = \{u_i \mid i = 1,\ ...,\ I\}$ – множество информационных объектов: документы различных форматов и отдельные фрагменты информации, хранящиеся в СЭД, I – общее количество таких объектов; $P = \{p_q \mid q = 1,\ ...,\ Q\}$ – множество пользователей, Q – общее количество пользователей; T – множество дискретных моментов времени, представленных натуральными числами; RE – множество воздействий на объекты, как внешних, так и внутренних; $O = \{o_l \mid l = 1,\ ...,\ L\}$ – множество операций, выполняемых над объектами, L – общее количество операций; $G = \{G_i(C_i, O) \mid i = 1,\ ...,\ I\}$ – множество структур информационных потоков, каждый из которых отражает жизненный цикл объекта u_i как последовательную смену состояний через осуществление множества операций O (второй уровень масштабирования модели многоуровневых графов).

Рассмотрим элементы этих множеств. Объекту u_i соответствует некоторое множество состояний C_i, каждое из которых определяет содержимое объекта в определенный промежуток его жизненного цикла: $C_i = \{c_{ij} \mid j = 1, \ ..., \ J_i\}$, J_i – количество таких состояний. Каждое состояние определяется как кортеж из множества атрибутов объекта и их значений в заданный промежуток времени (аналогично формулам (3.3), (3.4)).

Каждый пользователь p_q в модели структуры электронного документооборота представляет собой совокупность ряда атрибутов, характеризующую его роль:

$$p_q = \{(\{ap_{qm}\}, \ t) \mid t \in T_{p_q}\}, \tag{6.2}$$

где $\{ap_{qm}\}$ – множество атрибутов пользователя, определяющих доступные ему функции, уровень доступа, личные данные и т.д. Множество атрибутов объединено в кортеж с моментами времени, в которые значения атрибутов являются актуальными; T_{p_q} – множество моментов времени существования пользователя p_q в СЭД.

Воздействия разделяются на внешние EE и внутренние IE: $RE = EE \cup IE$. Внешние воздействия $EE = \{ee_w \mid w = 1, \ ..., \ W_e\}$ включают распоряжения министерств, новые законы и стандарты, заказы от сторонних организаций и прочие воздействия, осуществляемые извне. Внутренние воздействия $IE = \{ie_w \mid w = 1, \ ..., \ W_i\}$ формируются на основе внешнего прямого способа (непосредственный приказ, распоряжение вышестоящих законодательных органов), либо косвенно (в результате изменения или появления новых стандартов и требований) внутри организации.

Воздействие, вне зависимости от того, является оно внешним или внутренним, направлено на получение конкретного результата – документа или некоторого их множества, которые удовлетворяют условиям, поставленным воздействием. К таким условиям относятся ограничения на круг лиц, ответственных за получение результата, требования к оформлению и содержанию документов, затраченному времени и т.д. В формализованном виде воздействие можно представить в виде кортежа:

$$\begin{gathered} ee_{we} = (U^*, \ p_q, \ T^*), \ ee_{we} \to \{ie_w \mid w = 1, \ ..., \ W_i, \ ie_w \in IE\}, \\ ie_w = (U^*, \ P^*, \ O^*, \ T^*), \\ ee_{we} \in EE, \ U^* \in U, \ p_q \in P, \ P^* \in P, \ O^* \in O, \ T^* \in T, \end{gathered} \tag{6.3}$$

где ee_{we} – внешнее воздействие, направленное на получение результата пользователем p_q в виде множества объектов $U*$ за отведенное время $T*$; ie_w – внутреннее воздействие, порожденное воздействием ee_{we} и направленное на получение объектов $U*$ некоторым подмножеством пользователей $P*$ путем выполнения операций $O*$ за время $T*$.

Важным вопросом при реализации СЭД является ограничение доступа пользователей к отдельным модулям, функциям и информации в целом. Разграничение доступа к информации может проводиться различными методами, выбор которых обуславливается поставленными задачами и особенностями предметной области. Неправильно выставленные права доступа нарушают как безопасность системы, так и сохранность данных.

Таким образом, рассмотрена формализация в рамках теоретико-графовой модели основных объектов и субъектов предметной области электронного документооборота, а также различные варианты разграничения доступа к объектам и операциям.

6.1.2. Формализация процессов предметной области адаптивной системы электронного документооборота

Следующим этапом формализации предметной области является анализ процессов, протекающих в предметной области. Для их представления возможно использование математических соотношений (пример таких формул представлен при описании метода формализации на основе модели многоуровневых графов), а также графические схемы в нотации IDEF0 или в виде блок-схемы.

Классификация и распределение информации в СЭД

Классификация документов – одна из задач информационного поиска, заключающаяся в определении отношения документа к одной из нескольких категорий на основании содержания документа или каких-то иных признаков или атрибутов. Отметим, что необходимо разделять задачи классификации и кластеризации документов. В первом случае мы разбиваем множество всех документов по заранее заданным категориям в соответствии с известными критериями, во втором случае подобная группировка заранее не определена и производится на основе поступивших на вход алгоритма документов.

При решении задачи классификации информационных объектов необходимо выбрать систему классификации – совокупность методов

и правил классификации и ее результат, т.е. систему классификационных группировок (классов). Система классификации может быть представлена в виде перечня признаков классификации (фасетов) и их значений и правил образования группировок путем комбинирования признаков-фасетов. Основными методами классификации являются иерархический и фасетный [168]. Для машинного обучения более применимым является фасетный метод, так как каждому типу документов мы сможем сопоставить перечень признаков.

Методика классификации документов на основе фасетного метода заключается в следующем: каждому объекту документооборота $u_i \in U$ соответствует некоторое подмножество признаков классификации f_j, на основе которых осуществляется их группировка в массивы Cat_j:

$$u_i \to \{f_j^i\},\ Cat_j : f_j \to \{u_i \mid f_j \in \{f_j^i\}\},\ f_j^i \in F,\ u_i \in U, \qquad (6.4)$$

где f_j^i – признак f_j классификации, характеризующий документ u_i; u_i – элемент множества документов, каждому из которых ставится в соответствие множество признаков $\{f_j^i\}$; Cat_j – функция формирования классификации по признаку f_j, которая ставит в соответствие каждому признаку f_j набор документов $\{u_i\}$, обладающих данным признаком.

Используя представленную методику, можно разбить множество всех документов по набору признаков, что значительно упростит подготовку информации для последующей ее обработки в методах классификации.

Поиск информации в СЭД

Далее рассмотрим процесс поиска информации в СЭД. Можно выделить две основные задачи поиска:

– поиск по атрибутам объектов;

– поиск похожих объектов.

Первая задача состоит в нахождении соответствий между вводимым пользователем набором данных и значениями атрибутов объектов. Данные соответствия могут быть:

– жесткими: искомые значения и значения атрибутов полностью совпадают, что происходит при поиске по формату файлов, категории, дате и т.д.;

– гибкими: искомое значение либо входит в значение атрибута, либо является его формой, примером такого соответствия является поиск по автору, названию файла, содержимому, т.е. различные символьные конструкции.

Перейдем к проблеме поиска сходных и связанных объектов. Примем следующие определения:

1) два объекта сходны, если найденная численная мера их сходства превышает заданный порог. Можно говорить о сходстве двух объектов, когда их состояния обладают некоторым набором атрибутов с равными значениями [169];

2) два объекта являются связанными, если они состоят в какой-либо связи [170]. Имеется в виду зависимость между объектами, их состояниями, атрибутами или их значениями, их получение или изменение с помощью различных операций.

Задача обнаружения дубликатов документов является частным случаем поиска схожих документов. В данной ситуации выбираются те документы, где величина сходства объектов максимальна.

Маршрутизация информации в СЭД

Маршрутизация (переадресация) документов является одной из ключевых задач в СЭД, так как ее автоматизация позволяет снизить негативное влияние человеческого фактора (неправильная переадресация), повысить производительность труда (ускорение поиска следующего исполнителя), эффективность работы организации в целом (повышение прозрачности и уровня контроля за исполнением операций над документами).

Реализация маршрутизации документов в информационных системах может осуществляться следующим образом: в свободной форме (исполнители не зафиксированы и выбираются произвольно на каждом этапе обработки документа); в жесткой форме (исполнители документа и весь маршрут его следования жестко зафиксирован и не меняется в процессе движения документа).

Анализ существующих подходов к маршрутизации и переадресации документов показал, что автоматизация данных процессов возможна с некоторой точностью при анализе атрибутов документов, что позволяет сформулировать перечень возможных исполнителей на основе содержимого документа [171].

Маршрутизация включает в себя определение оптимального маршрута документа и предполагает, что процесс его доставки должен включать критерии оптимальности, которые связаны с процессом самой передачи. На процесс маршрутизации также накладываются следующие ограничения:

1) максимально допустимое время движения документа в соответствии с существующими нормативными документами;

2) гибкость маршрута движения документа: он может быть изменен в процессе обработки данных в отдельных узлах;

3) ограничение доступа к информации в соответствии с требованиями информационной безопасности.

Задача маршрутизации в СЭД заключается в том, что необходимо перенаправить набор документов от начального источника до конечного в зависимости от каких-либо условий или правил, например на основе атрибутов документов. Рассмотрим пример алгоритма ее решения на основе теории систем массового обслуживания.

При работе с документом пользователю перед отправкой необходимо отметить его срочность. Документы разделяются на два типа: срочный и обычный. Документы будут обрабатываться последовательно для каждого получателя, поэтому необходимо при отправке помечать документ u_j дополнительным необходимым атрибутом d_j, чтобы последовательность документов была распределена по срочности его обработки.

Далее определяется основной атрибут документа – его категория, для чего необходимо решить задачу классификации документов. На данном этапе имеется множество заданных категорий, документов и неизвестная целевая функция (классификатор). Необходимо построить классификатор, который будет определять категорию документов с достаточной точностью.

Следующий шаг алгоритма – поиск и распознавание отправителей и получателей. В результате извлечения содержимого файла получим список ФИО, где каждому элементу будет соответствовать кортеж параметров: положение в тексте (*position*), падеж (*case*) и набор ближайших слов (*nwords*):

$$FIO \rightarrow (\{position\},\ case,\ \{nwords\}). \qquad (6.5)$$

На основе известных ФИО и их параметров можно определить, относится ли фамилия к получателю, отправителю или другим фамилиям, которые используются в документе:

$$F : FIO \rightarrow [P_{out},\ P_{in},\ other], \qquad (6.6)$$

где F – функция классификации ФИО.

На основе полученных атрибутов (срочность, категория, отправитель, получатель) определяется оптимальный маршрут документа. Для реализации этого этапа формируется матрица маршрутизации для

каждого документа. В нее входят отношения вида «получатель – отправитель». По ней можно определить, по какому маршруту перенаправляется документ, является ли это массовой рассылкой или единичной отправкой. Матрица заполняется нулями и единицами, где 1 соответствует наличию отношения «отправитель–получатель», а 0 – его отсутствию.

Конечный этап заключается в передаче документа в соответствии с маршрутом для дальнейшей работы. Пользователи могут за определенное количество времени получать не один документ для обработки, поэтому для организации их работы предлагается матрица приоритетов, которая задает отношения вида «пользователь–приоритет документа», т.е. каждому P_i пользователю соответствует i-я строка с множеством документов, приоритеты которых $\{d_{ij}\}$ заданы в порядке убывания.

Генерация документов в СЭД

Одним из важнейших процессов в электронном документообороте является автоматизация генерации документов. Первым этапом формирования электронной документации является анализ информационных потоков и выбор объектов документооборота, требующих автоматизации. Обозначим их через $U*$. Тогда СЭД должна выдать пользователю на заполнение необходимые формы, а затем сформировать по ним конечные документы $U*$. Существует множество способов реализации данной задачи [172], но одним из эффективных решения является использование шаблонов в качестве основы электронных документов. В качестве таких шаблонов возможно использование документов HTML.

Алгоритм генерации документов в СЭД основан на использовании уже существующих документов формата *.DOC в качестве шаблонов.

Первым этапом является обработка исходного документа. Он сохраняется как документ в формате HTML, после чего происходит проверка на появление ошибок в форматировании; если такие имеются, то происходит их исправление.

На втором этапе HTML-страница открывается в любом текстовом редакторе с поддержкой разметки и происходит добавление по тексту страницы, так называемых «меток» – символьных конструкций, имеющих уникальную структуру, что позволит позже однозначно определить их в тексте.

Третий этап требует написания специальной процедуры на выбранном языке программирования. Сложность процедуры варьируется

в зависимости от имеющихся в языке программирования средств обработки символьных конструкций, но общий смысл ее остается одинаковым: необходимо последовательно считать фрагменты текста шаблона и найти в нем метки. Для этого используемые в тексте метки должны иметь уникальный вид, не похожий ни на одну символьную конструкцию языка разметки HTML.

Следующим шагом будет анализ метки и замена ее на данные из таблиц базы данных, заданные статически (например, значения адресов, телефонов, какие-либо ссылки) либо динамически формируемые из запросов (имя автора, названия, поля документа и т.д.). При этом имеется возможность свободно добавлять новые, специализированные HTML-теги к тексту; в качестве примера можно указать вставку стандартного элемента MS Word «флажок» со значением «снят» или «установлен» в зависимости от значения соответствующего поля в базе данных.

После того как все метки заменены, происходит обратный процесс записи всего текста шаблона в новую HTML-страницу и передача ее пользователю в нужном формате.

Таким образом, пользователь получает исходный документ в первоначальном форматировании, но с уже внесенной информацией из БД. При необходимости он может внести в него дополнительные правки либо распечатать. Стоит заметить, что проведенные исследования могут стать основой для решения более сложных задач по формированию данных в СЭД.

Адаптация интерфейса СЭД

Данный процесс заключается в адаптации компонентов (интерфейса, визуального представления управляющих блоков и данных) под индивидуальные особенности каждого пользователя.

В основе решения задачи адаптации интерфейса СЭД сбор данных о программном и программном обеспечении пользователя, а также его персональных данных (пол, возврат, профессия, должность, образование и т.д.), что может повлиять на его роль в системе. Осуществив обработку собранных данных (статистическим или иным образом), возможно прогнозировать настройки интерфейса, наиболее предпочтительные для пользователя. Кроме этого, это позволяет оптимизировать графическое качество системы в зависимости от мощности оборудования пользователя.

Таким образом, необходимо реализовать программный модуль A, на вход которого поступает информация P о программных и аппаратных характеристиках оборудования каждого пользователя, личные данные, должность, иная сопроводительная информация, на выходе

задается набор параметров интерфейса Представления – тип интерфейса V_{type} и значения его параметров V_{param}:

$$A(P) = (V_{type},\ V_{param})\ . \qquad (6.7)$$

Полученные настройки интерфейса автоматически выставляются в системе.

Процессы поддержки принятия решений в СЭД

В СЭД можно выделить следующие основные подходы к организации процесса принятия решений:

– управление системой только на основе решений человека;

– выдача системой рекомендаций или частичных решений, позволяющих упростить работу пользователя;

– формирование полного решения с необходимостью проверки и подтверждения пользователем;

– автоматическое формирование и выполнение решений с возможностью проверки и остановки пользователем;

– бесконтрольное автоматическое формирование и выполнение решений полностью без участия человека.

Наибольший интерес представляют системы поддержки принятия решений (СППР) последних типов, так как позволяют снизить влияние человеческого фактора, автоматизировать процессы управления, передачи и обработки информации в СЭД.

Рассмотрим формализацию нескольких процессов принятия решений в СЭД.

Одна из задач принятия решения связана с поиском графической информации: дубликатов и схожих изображений. Для человека не составляет труда распознать два одинаковых изображения, даже если они повернуты под разными углами или имеют разные размеры, однако для информационной системы это не является тривиальной задачей. Особенно остро данная задача встает в СЭД, работающих с конструкторской документацией, чертежами и сканированными изображениями.

В контексте поставленной задачи распознавания конструкторской документации имеется ряд существенных трудностей: отсутствие большого набора данных; существование большого количества классов (на тысячу чертежей может приходиться несколько сотен классов), сложная структура изображений (большие пустые пространства, рамки различных размеров и видов, различные масштабы и углы поворота чертежа). Все это делает задачу однозначной классификации чертежей практически невозможной.

Однако можно выявить некоторые признаки изображений и дальше работать уже с признаками, а не графическими данными. Тогда получаем задачу, идентичную поиску сходных объектов (рис. 7.3). Однако в данном случае необходимо формализовать метрики, позволяющие СЭД осуществить принятие решение о сходстве или различии графических файлов.

Пусть для двух изображений заданы значения N признаков $\{y_{1,\,i} \mid i = 1,\ ...,\ N\}$ и $\{y_{2,\,i} \mid i = 1,\ ...,\ N\}$. Самым простым способом сравнения является определение полного (точного) соответствия значения признаков, однако для измененных изображений (яркость, поворот, размеры) эти значения всегда будут различаться.

Более универсальными подходами, отражающими степень сходства изображений, являются методы среднеквадратичного отклонения (*SD*) и схожести значений выходных каналов (*SV*).

Метод *SV* определяет похожесть двух изображений через сумму значений схожести по всем выходным каналам:

$$SV = \sum_{i=0}^{N} sv_i,$$

$$sv_i = \begin{cases} 1, \text{ если } \dfrac{\left| y_{1,\,i} - y_{2,\,i} \right|}{y_{2,\,i}} < \alpha, \\ 0, \text{ в противном случае.} \end{cases} \tag{6.8}$$

Метод *SD* определяет похожесть двух изображений по величине среднеквадратичного отклонения значений по всем выходным каналам:

$$SD = \sum_{i=0}^{N} (y_{1,\,i} - y_{2,\,i})^2, \tag{6.9}$$

где sv_i – значение схожести i-х выходных каналов двух изображений, равное 1 при их отклонении не более чем в α раз, и 0 – в противном случае; $y_{j,\,i}$ – значение i-го признака j-го изображения.

Следующим важным процессом принятия решений в СЭД является прогнозирование действий пользователей. При входе пользователя в СЭД запускается отслеживание его действий. Таким образом формируются последовательности действий для каждого пользователя в рамках каждой сессии. Если в дальнейшем пользователь выполняет несколько действий из начала последовательности, СЭД может выдать

ему рекомендацию (подсказу) либо осуществить переход автоматически. В последнем случае СЭД реализует автоматическую навигацию.

Таким образом, рассмотрены и формализованы основные процессы в предметной области СЭД. Часть из них в дальнейшем в соответствии с методологией будут заменены нейросетевыми компонентами.

6.1.3. Разработка математической модели структуры адаптивной системы электронного документооборота на основе нейросетевой архитектуры

Формализуем структурную модель адаптивной СЭД, выполненную в соответствии с принципами методологии структурно-параметрического синтеза АИС и нейросетевой архитектурой (рис. 6.1).

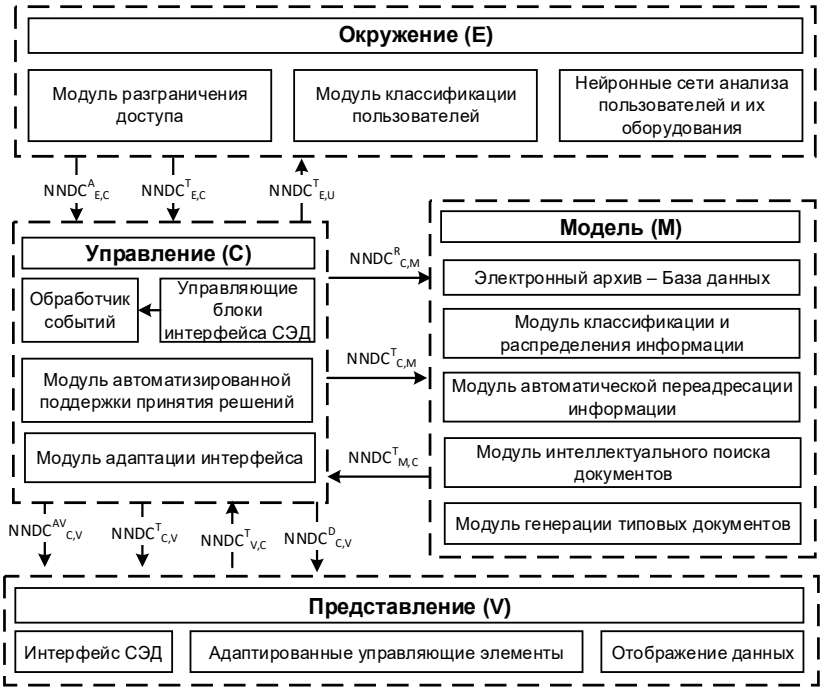


Рис. 6.1. Нейросетевая архитектура адаптивной СЭД

Сущность Окружение в адаптивной СЭД включает модули по анализу и обработке информации о пользователях. Модуль разграничения доступа пользователей к информации реализуется на основе существующих моделей разграничения доступа к информации: дискретном, ролевом или атрибутном представлении. Возможно применение нейронных сетей для автоматизации проверки уровня доступа пользователей. Модуль классификации пользователей реализует функции определения роли пользователя в СЭД. Его работоспособность обеспечивается нейронными сетями, осуществляющими анализ индивидуальных характеристик пользователей, их аппаратного и программного обеспечения. Данные нейронные сети также принимают участие в решении задачи адаптации интерфейса СЭД под конкретного пользователя.

Управление содержит два программных блока (управляющие блоки и обработчик событий), функционирующих по классическим методам реализации информационных систем. Остальные три блока реализуются посредством применения нейронных сетей. Модуль автоматизированной поддержки принятия решений заменяет классические подходы в виде баз знаний, экспертных систем, семантических сетей и т.д. В нем осуществляется принятие решений о сходстве документов (данная функция интегрируется в модуль поиска в СЭД), а также о переходе к следующей операции для пользователя (для обеспечения автоматической навигации по СЭД). Модуль адаптации интерфейса функционирует на базе нейросетевого метода адаптации информационных систем и реализует возможность автоматической настройки параметров СЭД под индивидуальные особенности пользователей и технические характеристики их терминалов.

Модель реализует основные блоки СЭД по хранению и обработке информации. Прежде всего это электронный архив для хранения документов и база данных для размещения информации о документах, пользователях, операциях и процессах в предметной области. Модуль классификации и распределения информации решает задачи по распознаванию типов документов и информации, определению их категории и перемещению в нужные категории, базируется на нейросетевом методе классификации и распознавания информации. Модуль автоматической переадресации используется для передачи информации между пользователями, а также определения нагрузки на пользователей и распределения документов для последующей обработки. Реализуется с помощью нейросетевого метода автоматической переадресации. Модуль интеллектуального поиска осуществляет поиск документов и информации в СЭД, часть операций данного типа осуществляется с при-

менением нейросетевых технологий (например, поиск дубликатов и похожих документов).

Представление содержит основные модули, реализует непосредственно интерфейс СЭД, необходимую визуализацию управляющих элементов и поля форм для ввода информации, визуализацию данных. Интерфейс автоматически персонализируется и подстраивается под тип пользователя с помощью нейросетевого канала адаптации.

Формализуем математическую модель структуры адаптивной СЭД MM_{EDMS} в нотации теории множеств и с помощью графовой модели (рис. 6.2).

$$MM_{EDMS} = (NNA,\ PRM,\ R), \tag{6.10}$$

где NNA – нейросетевая архитектура модулей адаптивной СЭД, включающая множество компонентов и связей между ними; PRM – параметры СЭД, отвечающие за ее функционирование; R – множество оценок эффективности СЭД.

Структура адаптивной СЭД в формализованном виде описывается следующими соотношениями и взаимосвязями между элементами:

$$NNA = (E,\ C,\ M,\ V). \tag{6.11}$$

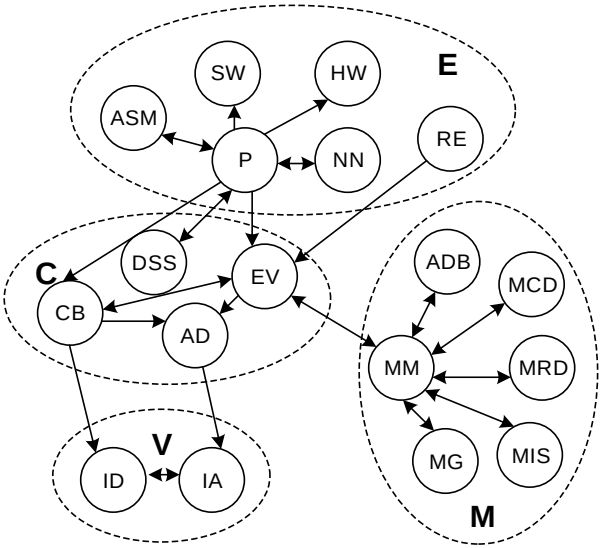


Рис. 6.2. Графовая модель адаптивной СЭД

Окружение включает множество модуль разграничения доступа к информации пользователей ACM, модуль классификации пользователей, основанный на нейронной сети NN_{CP}, информацию о параметрах аппаратного HW и программного SW обеспечения:

$$E = (ACM,\ NN_{CP},\ HW,\ SW). \tag{6.12}$$

Модуль разграничения доступа сопоставляет пользователей P с их уровнями доступа PP:

$$ACM : P \to PP. \tag{6.13}$$

Модуль классификации пользователей осуществляет следующее отображение характеристик пользователя на его категорию PK:

$$NN_{CP} : (P,\ SW,\ HW) \to PK. \tag{6.14}$$

Окружение также формирует ряд воздействий RE, которые можно разделить на два типа: действия пользователя AP (внутренние) и внешние факторы EF:

$$AP \ \to\ \{ie_w\},\ EF \ \to\ \{ee_w\}. \tag{6.15}$$

Управление включает управляющие блоки CB (программные компоненты интерфейса, с которыми взаимодействует пользователь в ходе своих действий), обработчик событий EV, модуль адаптации интерфейса AD, модуль автоматизированной поддержки принятия решений DSS:

$$C = (CB,\ EV,\ AD,\ DSS). \tag{6.16}$$

Обработчик событий EV содержит множество функций $f(re_w)$ – программных методов, осуществляющих ответные действия на воздействия re_w:

$$EV \ =\ \{f(re_w)\}. \tag{6.17}$$

В зависимости от возникающих событий в результате воздействия re_i обработчик EV по результату функций $\{f(re_i)\}$ формирует новые параметры $AD \ = \ \{ad_i\}$, такие, что

$$\sum_i g(ad_i) - g(re_i) \to 0. \tag{6.18}$$

Функции $g(x)$ используются для оценки влияния аргумента x на работоспособность адаптивной СЭД. Соотношение (6.18) отражает

функции адаптивности: значения параметров AD должны нивелировать эффект возникающих событий от воздействий $\{re_i\}$. Модуль адаптации интерфейса посредством реализации нейросетевого метода адаптации NN_{AD} осуществляет связь между программно-аппаратным обеспечением пользователей и подмножеством параметров AD^*:

$$NN_{AD} : (P,\ SW,\ HW) \to AD^*,\ AD^* \in AD. \qquad (6.19)$$

Модуль автоматизированной поддержки принятия решений формирует на основе некоторой входящей информации X решение Y, которое может быть рекомендацией к действиям пользователя, обработкой некоторого события, либо иными данными, позволяющими пользователю сформировать решение. Данный модуль реализуется на основе метода нейросетевого управления:

$$DSS : X \to Y,\ Y = AP,\ EV,\ \ldots. \qquad (6.20)$$

В СЭД используется синтез отдельных фрагментов компонентов Модели: модели информационных объектов MI (хранящей данные и их структуру), функциональных блоков MF (программных модулей, функций и процедур), процессов MP (формирующей специфические процессы движения и обработки информации). Из них формируются пять отдельных блоков:

$$M = (ADB,\ MCD,\ MRD,\ MIS,\ MG). \qquad (6.21)$$

Модуль электронного архива и базы данных ADB содержит все необходимые функции по хранению и выдаче информации с сервера адаптивной СЭД.

Модули классификации и распределения информации MCD, автоматической переадресации информации MRD функционируют на основе соответствующих нейросетевых методов.

Модуль интеллектуального поиска документа MIS решает задачи поиска информации в СЭД, в том числе поиск дубликатов документов и изображений с применением нейронных сетей.

Модуль генерации типовых документов MG осуществляет формирование электронных версий документов.

Все модули Модели обмениваются информацией с другими компонентами СЭД через модуль посредника *MM*.

Представление содержит отображаемые данные ID и адаптивные элементы интерфейса IA:

$$V = (ID,\ IA). \qquad (6.22)$$

Далее рассмотрим системные связи между компонентами модели адаптивной СЭД:

$$D = \left(D_E, D_C, D_M, D_V, D_{OC}, D_{CM}, D_{CV}\right), \tag{6.23}$$

где D – множество ребер графа структуры адаптивной СЭД, являющихся связями между компонентами структуры. Взаимосвязи между компонентами структуры АИС можно разделить на внутримодульные (D_E, D_C, D_M, D_V) и межмодульные (D_{EC}, D_{CM}, D_{CV}). Часть связей реализуется посредством нейросетевых каналов данных.

Параметры регулирования PRM включают множество всех параметров модулей и компонентов СЭД, включая параметры модуля адаптации AD:

$$PRM = \{prm_i\},\ AD \subset PRM. \tag{6.24}$$

Параметры prm_i обозначают атрибуты программных методов и функций, настройки компонентов, являются ограничениями и граничными условиями для процессов и операций.

Для оценки эффективности работы СЭД можно осуществить ее оптимизацию по нескольким направлениям. Будем использовать комплексный критерий оптимизации АИС. В качестве метода поиска оптимального решения FR примем метод изменения ограничений.

В рамках задачи проектирования адаптивной СЭД можно говорить о большей важности экономических затрат над производительностью, качеством или иными другими показателями СЭД. Все остальные метрики комплексного критерия оптимальности АИС будут учитываться как дополнительные условия, что, с одной стороны, позволит найти решение в области значений, удовлетворяющих поставленным дополнительным условиям на производительность, сложность реализации, адаптивность и качество СЭД, а с другой – определить экстремальное значение целевой функции в этой области.

Тогда получим следующую постановку задачи структурно-параметрического синтеза адаптивной СЭД.

Необходимо определить такие структуру NNA^* и параметры PRM^* СЭД, при которых целевая функция R оптимизации экономических затрат достигает экстремума:

$$\begin{gathered}\{NNA^*,\ PRM^*\} = \underset{NNA,\ PRM}{\arg\min}\ (R_V), \\ R_V = V_{hw} + V_{sw} + V_{pers} + V_{data} + V_{rd}\end{gathered} \tag{6.25}$$

при выполнении связей в виде математической модели (6.10) – (6.24) и ограничений на

– максимальную сложность программной реализации СЭД:

$$R_D \le R_D^{\max}, \tag{6.26}$$

– оценку адаптивности СЭД:

$$R_A \ge R_A^{\min}, \tag{6.27}$$

– производительность документооборота:

$$R_P \ge R_P^{\min}, \tag{6.28}$$

– оценку качества работы СЭД:

$$R_Q \ge R_Q^{\min}, \tag{6.29}$$

где $R_D^{\max}$ – оценка сложности программного обеспечения, реализованного на основе алгоритмического (классического) подхода; $R_A^{\min}$ – минимальная оценка адаптивности либо оценка используемого ранее программного обеспечения; $R_P^{\min}$ – минимально возможная производительность СЭД, зависящая от максимально допустимого времени обработки и передачи информации; $R_Q^{\min}$ – минимально допустимая оценка качества программного обеспечения либо оценка используемого ранее программного обеспечения.

После формализации предметной области, протекающих в ней процессов, разработки теоретико-множественной модели структуры адаптивной СЭД и постановки задачи ее синтеза начинается этап разработки. В качестве примеров адаптивных СЭД будем рассматривать СЭД конструкторской документации, а также отдельные модули СЭД научно-образовательных учреждений. Их специфика полностью соответствует тем моделям, процессам и задачам, что были рассмотрены ранее.

6.2. РЕАЛИЗАЦИЯ НЕЙРОСЕТЕВЫХ КОМПОНЕНТОВ И АПРОБАЦИЯ НЕЙРОСЕТЕВЫХ МЕТОДОВ В АДАПТИВНОЙ СЭД

В соответствии с разработанной методологией после постановки задачи начинается этап реализации нейросетевых компонентов, в ходе которого реализуется ряд модулей СЭД, функционирующих на основе

нейронных сетей. Для их формирования необходимо применение программных инструментов в виде нейросетевых каналов данных, математического и алгоритмического обеспечения нейросетевых методов.

Рассмотрим решение конкретных прикладных задач в предметной области СЭД на примере СЭД конструкторской документации.

6.2.1. Классификация документов СЭД на основе нейросетевого метода классификации и распределения данных

На первом этапе решения задачи классификации документов в СЭД перечислим основные и дополнительные признаки документов (рис. 6.3).

Основные и специфические признаки документов позволяют осуществить классификацию документов, причем как аналитически, так и с использованием современным методов машинного обучения. Таким образом, мы переходим от первой задачи (построения системы классификации) ко второй – подготовительной обработке документов.

Машинное обучение является эффективным инструментом при реализации алгоритмов классификации, маршрутизации, обработки и поиска документов, однако определяющее значение в этих процессах имеет качество исходных данных. Именно поэтому проведение подготовки исходных документов, их предварительная обработка позволяют значительно повысить точность результатов, получаемых в ходе применения машинного обучения.

Рис. 6.3. Структурная схема признаков классификации документов

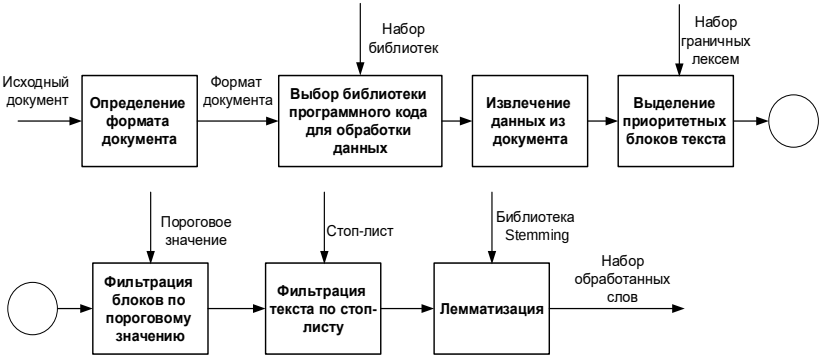


Рис. 6.4. Алгоритм представительной обработки документов

Учитывая специфику предметной области и приведенные на рис. 6.3 признаки классификации документов, сформулирована методика предварительной обработки документов, направленная на повышение точности работы методов машинного обучения. Представим ее в виде алгоритма (рис. 6.4).

Представленная методика позволяет значительно ускорить выполнение процедуры машинного обучения и ее точность для осуществления классификации документов. Эффективность работы классификаторов до применения разработанной методики и после представлена на рис. 6.5, 6.6. В качестве выбранной технологии машинного обучения использовался наивный байесовский классификатор.

Рис. 6.5. Влияние предварительной обработки данных на точность классификатора документов

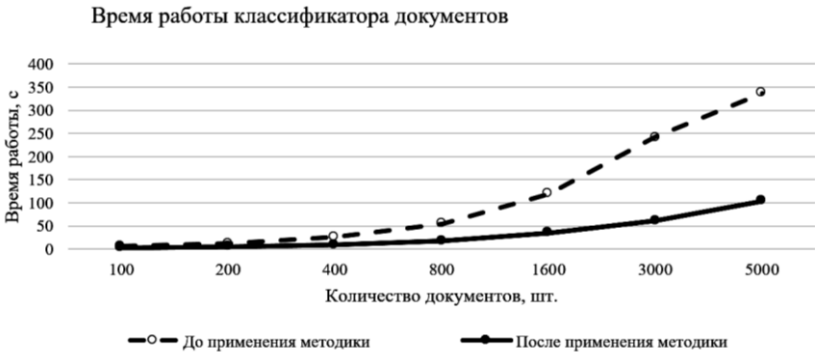


Рис. 6.6. Влияние предварительной обработки данных на время работы классификатора документов

В среднем точность классификации документов повысилась с 80 до 87%, а время работы классификатора снизилось в 3,2 раза за счет выделения приоритетных блоков документов и сокращения общего объема анализируемого текста.

Далее было проведено исследование эффективности представленной методики при классификации документов с помощью нейронных сетей (библиотека Tensorflow). Несмотря на то что такого же значительного сокращения времени обучения не произошло (показатель улучшился в диапазоне 1,33–1,5 раза), точность классификации возросла в среднем с 73 до 89%.

Полученные результаты были интегрированы в нейросетевой метод классификации и распределения данных, что позволило повысить эффективность решения задачи классификации документов, сократить время и сложность ее решения.

Далее проведена апробация нейросетевого метода классификации и распределения данных на примере модуля СЭД учебно-методических документов научно-образовательного учреждения. В качестве основного признака классификации использовалось «Наименование документа» (т.е. его категория: служебная записка, заявление, рабочая программа, лекционный материал и т.д.), а для решения задачи использовались различные методы машинного обучения.

В итоге получены экспериментальные данные, представленные на рис. 6.7 и 6.8. При обучении использовался набор из 3000 документов.

Полученные результаты подтверждают оправданность применения нейросетевого метода для решения задачи классификации документов.

Рис. 6.7. Сравнение точности классификации документов при различных методах машинного обучения

Рис. 6.8. Сравнение времени обучение классификатора документов при различных методах машинного обучения

Достигнуты положительные результаты как по времени обучения классификатора (вплоть до трехкратного улучшения показателя), так и по точности его работы (прирост от 5 до 20%).

6.2.2. Реализация нейросетевого метода адаптации для персонализации интерфейса в СЭД

Для решения задачи адаптации интерфейса СЭД под индивидуальные особенности пользователей в системе реализуется нейросетевой метод адаптации. Для тестирования метода использовалась реализованная СЭД на базе фреймворка Laravel, включающая регистрацию

пользователей и работу с электронным архивом документов. Таким образом, рассматривается простейшая СЭД, разработанная в рамках нейросетевой архитектуры, но изначально не обладающая специализированной функциональностью по адаптации интерфейса.

Разработанная тестовая СЭД состоит из нескольких компонентов: страница регистрации с вводом информации о пользователе, страница с настройками интерфейса, главная страница с основным меню и возможностью просмотра сведений о документах (рис. 6.9). Тестовая СЭД использовалась только для сбора информации о пользователях и выбранных ими параметров интерфейса. Собранные данные сохранялись в формате JSON и передавались на сервер.

Анализ предметной области позволил сформировать перечень выходных параметров, основанный на личной информации о пользователе, используемом им программном и аппаратном обеспечении (табл. 6.1). Сбор параметров осуществляется при регистрации и в процессе работы с помощью встроенных в СЭД средств мониторинга. В качестве выходных параметров выбраны настройки интерфейса. Представленный перечень в дальнейших исследованиях может быть расширен.

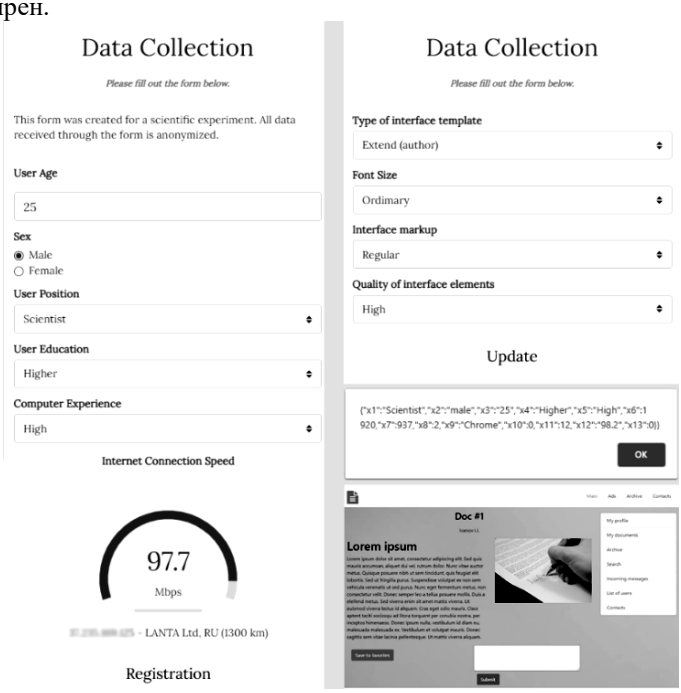


Рис. 6.9. Примеры работы тестовой СЭД

6.1. Описание входных и выходных параметров

Обозначение	Описание	Область определения
x_1	Роль пользователя	{гость, студент, преподаватель, ученый, сотрудник редакции}
x_2	Пол пользователя	{мужской, женский}
x_3	Возраст пользователя	[18; 80+] (лет)
x_4	Образование пользователя	{отсутствует, среднее, высшее, высшее с ученой степенью}
x_5	Опыт работы за компьютером	{небольшой, средний, большой}
x_6	Ширина окна браузера	[720; 3840] (пикселей)
x_7	Высота окна браузера	[480; 1600] (пикселей)
x_8	Тип клиента	{мобильный, планшетный, настольный компьютер}
x_9	Браузер	{Safari, Firefox, Chrome (Android), Safari (iOS), Chrome, Opera, Microsoft Explorer }
x_{10}	Язык браузера	{русский, английский}
x_{11}	Количество ядер процессора	[1; 8] (шт.)
x_{12}	Скорость подключения к серверу	[0; 100] (Мбит/с)
x_{13}	Наличие графического процессора	{нет, да}
y_1	Тип шаблона интерфейса, определяющий количество функциональных элементов	{облегченный, обычный, расширенный (авторский), расширенный (сотрудник)}
y_2	Размер шрифта	{небольшой, обычный, крупный}
y_3	Разметка интерфейса	{мобильная, обычная}
y_4	Качество элементов интерфейса и изображений	{низкое, обычное, высокое}

На основе представленных в табл. 6.1 параметров осуществлен сбор необходимых данных о пользователях через СЭД для редакции научного журнала. Общий объем собранных данных составил 7900 записей (включает анонимные данные гостей сайта журнала, сотрудников организации, авторов, рецензентов и т.д.). Исходные данные разделены на обучающий и тестовый набор в соотношении 80/20.

Дальнейшая работа заключалась в определении оптимальной структуры нейронной сети, для чего проводился ряд экспериментов, в результате которых получена следующая структура сети: 13 входных нейронов, три скрытых плотных слоя, четыре выхода с 4, 3, 2 и 3 нейронами соответственно.

Подготовив и обучив нейронную сеть, на следующем этапе осуществляется разработка API на основе REST API языка Python, который позволяет принимать извне входные параметры X и отправлять спрогнозированные нейронной сетью выходные параметры Y обратно в формате JSON.

Для тестирования разработанного программного обеспечения использовалась следующая схема эксперимента:

1) Пользователь регистрируется в тестовой СЭД, вводя свои личные данные. Остальные входные параметры считываются самой системой средствами библиотек JavaScript.

2) Данные компонуются в JSON-объект и отправляются на сервер через API адаптации.

3) Данные преобразуются в массив из формата JSON и поступают как входные параметры в загруженную обученную нейронную сеть.

4) Полученные выходные параметры компонуются в JSON-объект и отправляются обратно в тестовую СЭД.

5) JSON-объект распаковывается, значения выходных параметров устанавливаются в качестве рекомендованных параметров интерфейса и записываются в базу данных. Параметры обрабатываются шаблонизатором (например, в фреймворке Laravel используется Blade).

6) Пользователь входит в адаптированную под него тестовую СЭД.

7) При повторном заходе в систему параметры интерфейса загружаются из базы данных.

8) Если пользователь переключается между компьютером и мобильным устройством, параметр y_3 («Разметка интерфейса») проверяется каждый раз, когда страница обновляется и изменяется независимо от настроек, сохраненных в базе данных.

Для проверки работоспособности изложенного подхода проведено два эксперимента, входные и выходные данные которых представлены в табл. 6.2.

6.2. Результаты экспериментальных исследований по адаптации СЭД

№	Входные данные	Выходные данные
1	{ x_1 = гость, x_2 = мужской, x_3 = 22, x_4 = среднее, x_5 = большой, x_6 = 1280, x_7 = 800, x_8 = настольный компьютер, x_9 = Safari, x_{10} = английский, x_{11} = 2, x_{12} = 4.5, x_{13} = нет}	{ y_1 = облегченный, y_2 = обычный, y_3 = обычная, y_4 = низкое}
2	{ x_1 = преподаватель, x_2 = мужской, x_3 = 27, x_4 = высшее со степенью, x_5 = большой, x_6 = 1920, x_7 = 1080, x_8 = настольный компьютер, x_9 = Chrome, x_{10} = английский, x_{11} = 4, x_{12} = 50.2, x_{13} = да}	{ y_1 = расширенный (авторский), y_2 = обычный, y_3 = обычная, y_4 = высокое}

Сгенерированный на основе выходных данных нейронной сети интерфейс представлен на рис. 6.10 и 6.11 соответственно для каждого эксперимента.

Рис. 6.10. Результаты эксперимента № 1 по адаптации СЭД

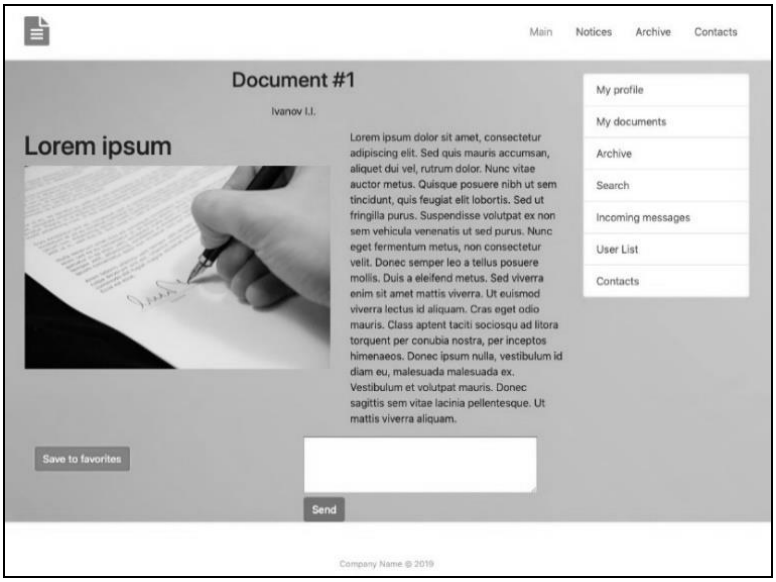


Рис. 6.11. Результаты эксперимента № 2 по адаптации СЭД

Таким образом, подтверждена работоспособность нейросетевого метода адаптации параметров интерфейса в СЭД.

6.2.3. Автоматизация поддержки принятия решений в СЭД с применением нейросетевых технологий

В процессе поддержки принятия решений в СЭД конструкторской документации выделим две подзадачи:

1) поиск похожих чертежей (изображений);

2) автоматизация принятия решения о следующем действии пользователя (автоматическая навигация).

Для распознавания изображений различной степени сложности широко используются сверточные нейронные сети. При этом можно сформировать собственную структуру сети либо использовать уже готовые модели обученных сетей, например VGG16 или ResNet, показывающие достаточно высокую точность на различных типах изображений [173 – 177]. В контексте поставленной задачи распознавания конструкторской документации имеется ряд обозначенных ранее проблем существенных трудностей: отсутствие большого набора данных для обучения и тестирования; существование большого количества классов (по факту – на тысячу чертежей может приходиться несколько

сотен классов); сложная структура изображений (большие пустые пространства, рамки различных размеров и видов, различные масштабы и углы поворота чертежа). Все это делает задачу классификации чертежей практически невозможной.

Однако в качестве возможного решения предлагается использовать нейросетевые методы классификации и распределения данных и управления, в основе которых использование обученной сверточной нейронной сети (например, VGG16). Для этой сети необходимо получить совокупность значений выходных каналов, отражающих степень принадлежности к каждому из 1000 классов. Далее, сравнивая эти наборы значений у различных чертежей, возможно осуществить автоматическое принятие решений, насколько они являются похожими или разными. В ходе экспериментальных исследований рассматривались различные варианты сравнения значений – полное соответствие, соответствие числа десятичных разрядов, среднеквадратичное отклонение, анализ количества схожих признаков. Более универсальными подходами, отражающими степень сходства изображений, оказались методы среднеквадратичного отклонения (SD) и схожести значений выходных каналов (SV).

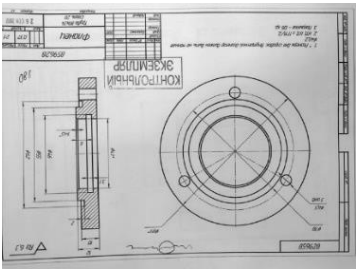
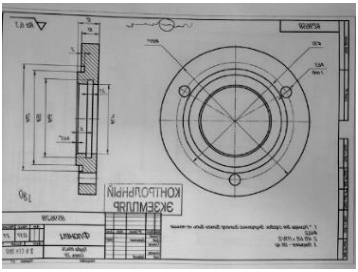
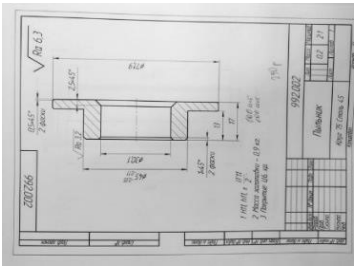
Используя представленные метрики, осуществлен эксперимент на малой выборке чертежей. На вход нейронной сети с архитектурой VGG16 поступают несколько изображений, автоматически масштабируемые до размеров 224×224 пикселей. Обученная сеть для каждого изображения возвращает массив из 1000 значений, которые затем обрабатываются по представленным методикам.

Результаты экспериментов при распознавании похожих чертежей представлены в табл. 6.3 на примере трех изображений: исходное (эталон), производное от него (с другой цветовой гаммой) и полностью отличающееся. Далее указаны значения среднеквадратичного отклонения значений выходных каналов всех изображений от эталонного, а также два столбца со значениями количества сходных признаков при двух различных вариантах параметра α. Из полученных данных можно сделать следующие выводы: предлагаемый подход с использованием обученной нейронной сети и методов анализа значений выходных каналов позволяет успешно оценить степень похожести изображений.

Таким образом, используя нейросетевой метод классификации и распознавания, можно автоматизировать процесс принятия решений о похожести или различии чертежей. Пользователь при загрузке нового чертежа получает подсказку о наличии или отсутствии дубликатов, после чего принимает решение о загрузке документа в архив. При этом отпадает необходимость вручную проверять все записи, что приводит к значительному сокращению времени и ликвидирует возможность дублирования чертежей в базе данных. Дальнейшая работа в этом

направлении будет связана с совершенствованием методики оценки похожести чертежей для повышения ее универсальности и точности. Так же предполагается проведение исследований в области улучшения быстродействия алгоритма, так как с ростом базы данных чертежей время сравнения будет постоянно возрастать. Решить данный вопрос возможно за счет модернизации нейронной сети или оценочных методов.

6.3. Результаты исследований по распознаванию похожих чертежей

Наименование	Изображение	SD	SV, α = 0.1	SV, α = 0.2
Эталон		0	1000	1000
Похожее изображение		0.0004	77	173
Другое изображение		0.6731	0	0

Далее рассмотрим задачу прогнозирования действий пользователей для автоматизации навигации по СЭД с помощью нейронных сетей. Для обучения тестовой нейронной сети подготовлена небольшая тренировочная выборка деятельности нескольких пользователей. Итоговая точность нейронной сети на контрольном наборе составила 79%. Точность может быть улучшена путем увеличения

При входе пользователя в систему запускается модуль прогнозирования действий пользователей. Текущее действие отслеживается и отправляется на вход нейронной сети, которая возвращает следующее возможное действие на основе набора типовых взаимодействий вида «пользователь–СЭД–документ». Примеры возможных взаимодействий представлены в табл. 6.4.

Таким образом, если текущим действием является «Завершение загрузки документа», то в качестве возможных следующих операций в СЭД могут быть выбраны: переход на главную страницу, загрузка следующего документа либо заполнение сведений о связанных с документом заказов. Каждая из перечисленных операций приводит к соответствующим действиям с документами. Использование заранее заданных сценариев позволит избежать совершения нелогичных действий.

6.4. Примеры взаимодействий вида «пользователь–СЭД–документ»

Пользователь	СЭД	Документ
Загрузка документа	Поиск похожих в базе	Добавление нового документа
		Выдача похожих документов
Вход в систему	Главная страница	Выдача загруженного документа
	Архив	Выдача последних документов
	Поиск	
	Добавление документа	Добавление нового документа
Завершение загрузки документа	Переход на главную страницу	Добавление нового документа
	Загрузка следующего документа	
	Заполнение сведений о заказах	Редактирование карточки документа

В соответствии с предлагаемым алгоритмом, если нейронная сеть предполагает процент вероятности следующего действия выше 95% (значение получено на основе экспертной оценки), то спрогнозированное действие выполняется автоматически, иначе пользователь получает подсказку. В процессе работы системы поддержки принятия решений нейронная сеть должна постоянно анализировать новые данные, что позволит повысить ее точность.

6.2.4. Применение нейросетевых методов при маршрутизации документов в СЭД

Рассмотрим решение задачи маршрутизации документов между пользователями в СЭД. Данная задача решалась в несколько этапов.

На первом этапе необходимо осуществить анализ и классификацию документов. Обученная нейронная сеть классификации документов показала точность почти 94%.

Далее в соответствии с формализацией процесса маршрутизации необходимо классифицировать пользователей в тексте документа, чтобы определить отправителя и получателя. Вторая нейронная сеть определяла по типу и позиции лексемы ее принадлежность к ФИО пользователей, а также категорию (исполнитель, отправитель, получатель, пр.). Для проверки точности второй нейронной сети использовалось специальное тестовое программное обеспечение (нейросетевой маршрутизатор) (рис. 6.12).

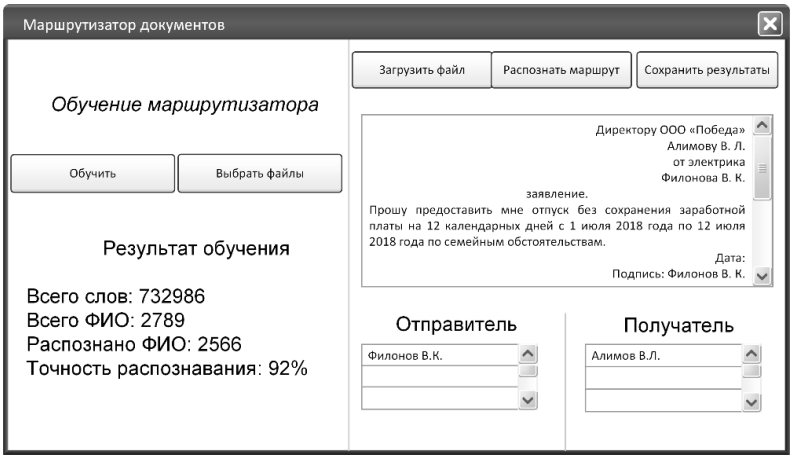


Рис. 6.12. Пример работы программного обеспечения нейросетевого маршрутизатора

Разработанный маршрутизатор позволяет осуществить первичное обучение на заданном множестве файлов, получить результаты обучения в виде количества обработанных слов, найденных и правильно распознанных среди них ФИО пользователей, рассчитать точность распознавания. В правой части окна маршрутизатора реализован поиск ФИО пользователей, распределение их по отправителям и получателям, что позволяет распознать дальнейший маршрут документа. Таким образом, разработанное на основе представленного алгоритма программное обеспечение позволяет сначала обнаружить ФИО пользователя системы документооборота, после чего определить его признаки и на их основе рассортировать пользователей по категориям «отправитель» и «получатель». Итоговая точность распознавания категории пользователей составила 92%.

Завершает решение задачи применение нейросетевого метода автоматической переадресации для формирования матрицы приоритетов операций и матрицы компетенций исполнителей.

Таким образом, маршрутизация документов осуществляется следующим образом: определяется категория документа, распознаются ключевые пользователи, участвующие в работе с ним (отправители и получатели), затем оценивается их загруженность и компетентность. В случае, если все требования, позволяющие осуществить операцию, выполняются, документ отправляется исполнителю.

6.3. КОНСТРУИРОВАНИЕ И ОПТИМИЗАЦИЯ АДАПТИВНОЙ СИСТЕМЫ ЭЛЕКТРОННОГО ДОКУМЕНТООБОРОТА

В качестве примера адаптивной СЭД будем рассматривать СЭД конструкторской документации. Тогда на первой итерации этапа Конструирования необходимо осуществить получение структуры и параметров модулей СЭД, при которых функция оптимальности (экономические затраты) достигает экстремума, а все ограничения выполнены.

Осуществим расчет метрик для классической и адаптивной реализации СЭД. Для классической реализации СЭД будет использоваться структура на основе распространенного шаблона проектирования MVC (рис. 6.13). Список модулей соответствует нейросетевой архитектуре СЭД (рис. 6.1), однако для каждого модуля используются алгоритмические методы решения задач анализа, обработки и передачи информации. Далее осуществим расчет основных метрик комплексного критерия оптимальности АИС для классической СЭД.

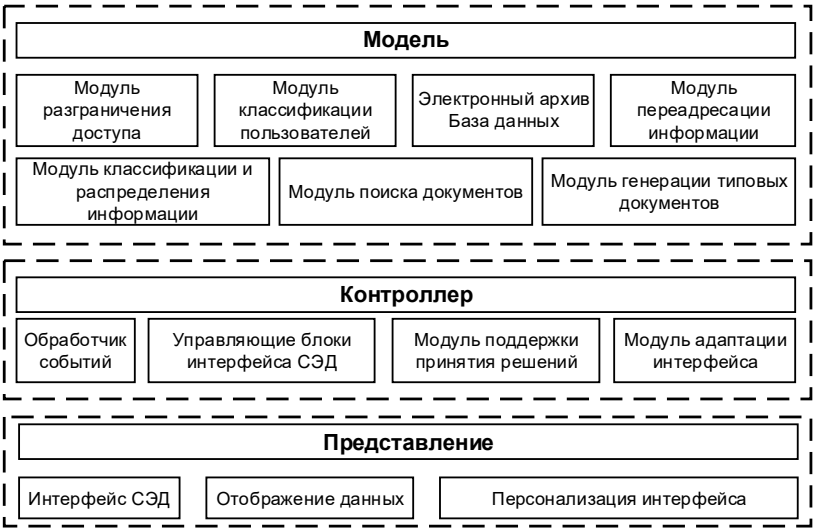


Рис. 6.13. Архитектура классической СЭД на базе шаблона MVC

Рассмотрим расчет экономических затрат R_V на реализацию СЭД (табл. 6.5). Затраты на аппаратное и программное обеспечение примем равными. Из-за использования нейронных сетей для анализа и обработки данных затраты на НИР, а также на обработку и передачу информации сокращаются.

6.5. Сравнение экономических затрат на реализацию СЭД

Компонент экономического критерия	Классическая СЭД	Адаптивная СЭД
Затраты на аппаратное обеспечение	214 461	214 461
Затраты на программное обеспечение	0	0
Затраты на зарплату персонала	482 478	386 562
Стоимость хранения информации	1500	1500
Стоимость обработки информации	33 333	20 000
Стоимость передачи информации	8204	5650
Затраты на проведение НИР	218 400	93 600
Общие затраты	958 377	721 774

6.6. Оценка сложности программной реализации СЭД

Метрика	Классическая СЭД	Адаптивная СЭД
COCOMO	0,1753	0,098
Джилба	0,25	0,22
Холстеда	0,61	0,49
Цикломатическая сложность	0,53	0,25
Общая оценка сложности	0,391	0,264

Расчет сложности реализации программного обеспечения классической и адаптивной СЭД осуществлялся по следующим метрикам: COCOMO, Джилба, Холстеда и цикломатической сложности (усредненной для каждого программного блока). Результаты измерений приведены в табл. 6.6. Так как при нейросетевой архитектуре алгоритмические блоки сокращаются либо заменяются на использование нейронных сетей, то наблюдается значительное сокращение сложности программной реализации. Это также приводит к снижению затрат на разработку.

Расчет адаптивности R_A для СЭД осуществлялся по набору критериев адаптивности и эргономических критериев (табл. 6.7). Экспертная оценка показала значительный рост по некоторым метрикам адаптивности за счет реализации новых функций в СЭД (например, автоматической адаптации интерфейса системы под потребности пользователя).

6.7. Оценка адаптивности СЭД

Критерий адаптивности	Классическая СЭД	Адаптивная СЭД
Время доступа к системе	0,9	0,9
Функциональность	0,9	0,9
Гибкость	0,5	0,8
Стабильность	0,9	0,9
Доступность	0,7	0,9
Качество поддержки	0,8	0,8
Доступность руководства пользователя	0,7	0,9
Загруженность интерфейса	0,8	0,9
Контроль пользователя над операциями	0,7	0,9

Продолжение табл. 6.7

Критерий адаптивности	Классическая СЭД	Адаптивная СЭД
Адаптивность	0,4	0,8
Управление ошибками	0,7	0,9
Согласованность	0,7	0,9
Значение идентификаторов (кодовых имен)	0,8	0,8
Совместимость	0,7	0,8
Общая оценка адаптивности	0,728	0,864

Качество работы классической СЭД R_Q определялось также экспертной оценкой по набору метрик (табл. 6.8). Наблюдается некоторое улучшение метрик при применении нейросетевого подхода.

6.8. Оценка качества СЭД

Критерий качества	Классическая СЭД	Адаптивная СЭД
Надежность	0,9	0,9
Безотказность	0,8	0,9
Долговечность	0,9	0,9
Ремонтопригодность	0,9	0,9
Достоверность	0,9	0,9
Эффективность	0,7	0,8
Целостность	1,0	1,0
Сложность	0,8	0,9
Структурированность	0,8	0,8
Адаптивность	0,5	0,9
Лабильность	0,5	0,8
Интегрируемость	0,8	0,8
Делимость	0,7	0,8
Валидность	0,9	0,9
Общая оценка качества	0,793	0,871

Наконец, осуществим расчет производительности R_P СЭД.

Для классической реализации производительность программного обеспечения СЭД составила: $P_{SW} = 0.93$, аппаратного: $P_{HW} = 0.9$. В итоге $R_P = 0.915$.

Для адаптивной СЭД имеет аналогичную производительность аппаратного обеспечения $P_{HW} = 0.9$. Производительность программного обеспечения адаптивной СЭД составила $P_{SW} = 0.95$. Прирост получен за счет более быстрого решения ряда задач анализа и обработки информации. В итоге для адаптивной СЭД $R_P = 0.925$.

Таким образом, в ходе решения задачи структурно-параметрического синтеза получены результаты (табл. 6.9), доказывающие эффективность применения разработанной методологии проектирования АИС.

Таким образом, наблюдается снижение экономических затрат при реализации адаптивной СЭД (тем самым достигается минимизация целевой функции), а также улучшение всех остальных метрик (что обеспечивает выполнение ограничений в поставленной задаче).

На рисунке 6.14 представлен общий положительный эффект от перехода от классического подхода к разработке СЭД к методологии на основе нейросетевой архитектуры.

6.9. Сравнение классической и адаптивной СЭД по метрикам комплексного критерия оптимальности

Метрика	Классическая СЭД	Адаптивная СЭД
Экономические затраты на разработку R_V	958 377	721 774
Сложность программного обеспечения R_D	0.391	0.264
Оценка адаптивности R_A	0.728	0.864
Оценка качества R_Q	0.793	0.871
Производительность R_P	0.915	0.925

Рис. 6.14. Положительный эффект от использования методологии

Основным эффектом, который достигнут в ходе применения методологии можно считать снижение стоимости (на 24,7%) и сложности (на 32,5%) реализации СЭД. Также значительно повысилась адаптивность системы (на 13,6%). Данные показатели подтверждают достижение поставленной цели исследования.

После успешного решения задачи структурно-параметрического синтеза и разработки адаптивной СЭД начинается этап ее внедрения, итогового тестирования и эксплуатации.

6.4. ВЫВОДЫ

В рамках данной главы рассматривается апробация методологии структурно-параметрического синтеза АИС на примере предметной области систем электронного документооборота.

Проводится подробное изложение применения методологии для разработки адаптивных СЭД с описанием каждого этапа. Рассматривается формализация объектов и субъектов предметной области, процессов, протекающих в ней. Используя метод формализации информационных потоков на основе моделей многоуровневых графов, разработана математическая модель структуры электронного доку-

ментооборота. Рассмотрено взаимодействие документов, пользователей, структура жизненного цикла документов, типовые операции над документами. С помощью функциональных моделей представлены процессы обработки информации: классификация документов, маршрутизация, поиск, генерация документов, поддержка принятия решений в СЭД.

Далее приводится нейросетевая архитектура адаптивной СЭД, описание сущностей и элементов с учетом специфики предметной области. Представлена математическая модель адаптивной СЭД, выполненная с применением теорий множеств и графов. Поставлена задача структурно-параметрического синтеза адаптивной СЭД.

Для проверки эффективности разработанной методологии проектирования АИС осуществлено решение поставленной задачи структурно-параметрического синтеза адаптивной СЭД. Приводится расчет метрик комплексного критерия оптимальности АИС для двух СЭД: классической, разработанной по методологии RAD и шаблону MVC, и адаптивной, проектирование которой осуществлялось в соответствии с предлагаемой методологией. В результате применения методологии достигнуто снижение стоимости (на 24,7%) и сложности (на 32,5%) реализации СЭД, повышена адаптивность системы (на 13,6%). Также наблюдается улучшение ее качества и незначительный прирост производительности.

ЗАКЛЮЧЕНИЕ

В рамках данной работы проведены исследования в области автоматизации структурно-параметрического синтеза АИС. Рассмотрено состояние вопроса разработки информационных систем: существующие методологии и архитектуры, методы реализации функций адаптивности и автоматизации разработки. Исследованы подходы к математическому моделированию процессов анализа, обработки и передачи данных, формализации структуры АИС и критерии ее оптимизации. Проведенный анализ показал, что для автоматизации структурно-параметрического синтеза АИС необходимо разработать новую методологию и архитектуру, основанную на применении методов машинного обучения и направленную на автоматизацию операций взаимодействия с информацией, что позволит снизить стоимость, время и сложность разработки АИС.

Разработана методология структурно-параметрического синтеза АИС на основе нейросетевой архитектуры, сформулированы принципы и структура методологии, используемые нейросетевые методы и формализованы основные этапы. Методология основана на существующих подходах в области проектирования информационных систем (RAD), но ее научная новизна заключается в выделении в отдельный этап реализации нейросетевых компонентов и ориентации на использование нейросетевых технологий и методов для решения задач автоматизации процессов анализа, обработки, генерации и передачи данных.

Представлены, формализованы и теоретически обоснованы автоматизированные нейросетевые методы обработки, передачи, переадресации, классификации и распределения, генерации информации, а также метод нейросетевого управления. Методы апробированы, доказана их применимость и эффективность, выраженная в снижении сложности программной реализации и повышении точности решения поставленных перед ними задач.

Рассмотрено применение методологии нейросетевой архитектуры и нейросетевых методов в рамках предметной области адаптивных систем электронного документооборота.

СПИСОК ЛИТЕРАТУРЫ

1. Макаренко, С. И. Интеллектуальные информационные системы / С. И. Макаренко. – 2009. – 206 с.

2. Федорова, О. В. Применение методологий SADT и ARIS для моделирования и управления бизнес-процессами информационных систем / О. В. Федорова, А. А. Мамаева, Е. А. Якунина // Вестник Воронежского государственного университета инженерных технологий. – 2018. – Т. 80, № 1(75). – С. 105 – 109.

3. Сидихменова, Е. И. Применение CASE-технологий при проектировании информационных систем / Е. И. Сидихменова, С. А. Ланец // Научно-техническое и экономическое сотрудничество стран АТР в XXI веке. – 2013. – Т. 1. – С. 288 – 294.

4. Чайников, С. И. Принципы организации вычислений на базе граф-модели предметной области / С. И. Чайников, А. С. Солодовников // Bonics of Intelligense: Sci. Mas. – 2012. – № 2. – С. 72 – 75.

5. Когаловский, М. Р. Системы доступа к данным, основанные на онтологиях / М. Р. Когаловский // Программирование. – 2012. – Т. 38, № 4. – С. 55 – 77.

6. Qureshi, M. R. J. Agile software development methodology for medium and large projects / M. R. J. Qureshi // IET software. – 2012. – V. 6, No. 4. – P. 358 – 363.

7. Улыбин, В. В. Архитектура информационных систем : учебное пособие / В. В. Улыбин. – Ульяновск : УлГТУ, 2019. – 192 с.

8. Ларина, И. Б. Методология RAD-разработки информационных систем / И. Б. Ларина, Н. А. Курапина, Г. В. Хачатурянц // Научные исследования: теория, методика и практика. – 2018. – С. 231 – 232.

9. Белоусова, С. А. Анализ подходов к созданию пользовательского интерфейса / С. А. Белоусова, Ю. И. Рогозов // Известия Южного федерального университета. Технические науки. – 2014. – № 6(155).

10. Верлань, А. Ф. Об организации адаптивного пользовательского интерфейса в автоматизированных системах / А. Ф. Верлань, М. Ф. Сопель, Ю. О. Фуртат // Известия ЮФУ. Технические науки. – 2014. – № 1(150). – С. 100 – 110.

11. Bastien, J. M. C. Evaluating a user interface with ergonomic criteria / J. M. C. Bastien, D. L. Scapin // International Journal of Human-Computer Interaction, 1995. – V. 7, No. 2. – P. 105 – 121.

12. Мезенков, А. А. Адаптация пользовательского интерфейса информационной системы к характеристикам пользователя / А. А. Ме-

зенков, С. В. Шибанов // Труды Международного симпозиума «Надежность и качество». – 2012. – Т. 1.

13. Применение адаптивного сенсорного интерфейса в приложениях информационной безопасности / К. Н. Жернова и др. // Вопросы кибербезопасности. – 2020. – № 1(35).

14. Первунинский, С. М. Разработка системы автоматизированной адаптации интерфейсов под потребности пользователей / С. М. Первунинский, А. А. Букша // Молодий вчений. – 2018. – № 11(2). – С. 1053 – 1056.

15. Яковлев, Ю. С. О развитии адаптивного человеко-машинного интерфейса и критериях его оценки в учебных системах / Ю. С. Яковлев, Л. И. Курзанцева // Образовательные технологии и общество. – 2013. – Т. 16, № 1. – С. 547 – 563.

16. Ghaibi, N. A tool support for the adaptation of user interfaces based on a business rules management system / N. Ghaibi, O. Dâassi, L. J. B. Ayed // Proceedings of the 29th Australian Conference on Computer-Human Interaction. – ACM, 2017. – P. 162 – 169.

17. Toward the adaptation of component-based architectures by model transformation: behind smart user interfaces / J. Criado et al. // Software: Practice and Experience. – 2015. – V. 45, No. 12. – P. 1677 – 1718.

18. A recommender system for component-based applications using machine learning techniques / A. J. Fernández-García et al. // Knowledge-Based Systems. – 2019. – V. 164. – P. 68 – 84.

19. Ham, N. Machine learning and dynamic user interfaces in a context aware nurse application environment / N. Ham, A. Dirin, T. H. Laine // Journal of Ambient Intelligence and Humanized Computing. – 2017. – V. 8, No. 2. – P. 259 – 271.

20. Asgar, T. S. Formalizing requirements in ERP software implementations / T. S. Asgar, T. M. King // Lecture Notes on Software Engineering. – 2016. – V. 4, No. 1. – P. 34.

21. Grif, M. G. Functional-structural theory based techniques for human-machine systems optimal design / M. G. Grif, S. A. Kochetov, N. D. Ganelina // 2016 13th International Scientific-Technical Conference on Actual Problems of Electronics Instrument Engineering (APEIE). – IEEE, 2016. – V. 2. – P. 494 – 497.

22. Personalization and adaptation in e-learning systems / A. Klašnja-Milićević et al. // E-Learning Systems. – Springer, Cham, 2017. – P. 21 – 25.

23. Software engineering for self-adaptive systems: A second research roadmap / R. De Lemos et al. // Software Engineering for Self-Adaptive Systems II. – Springer, Berlin, Heidelberg, 2013. – P. 1 – 32.

24. Vijayasarathy, L. R. Choice of software development methodologies: Do organizational, project, and team characteristics matter? / L. R. Vijayasarathy, C. W. Butler // IEEE software. – 2015. – V. 33, No. 5. – P. 86 – 94.

25. Herbsleb, J. D. Global software development / J. D. Herbsleb, D. Moitra // IEEE software. – 2001. – V. 18, No. 2. – P. 16 – 20.

26. Cliff, D. The global financial markets: An ultra-large-scale systems perspective / D. Cliff, L. Northrop // Monterey workshop. – Springer, Berlin, Heidelberg, 2012. – P. 29 – 70.

27. Мартынов, О. О. Автоматизация процессов сбора и обработки информации. Повышение эффективности принятия управленческих решений : монография / О. О. Мартынов. – М. : Янус-К, 2013. – 96 с.

28. Terletskyi, D. O. Mathematical foundations for designing and development of intelligent systems of information analysis / D. O. Terletskyi, O. I. Provotar // arXiv preprint arXiv:1510.04183. – 2015.

29. Эволюционная процедура структурного и параметрического синтеза имитационных моделей систем документооборота / В. А. Ломазов и др. // Научные ведомости Белгородского государственного университета. Сер. Экономика. Информатика. – 2013. – Т. 28, № 22-1(165).

30. Vasundara, M. Recent developments on machining fixture layout design, analysis, and optimization using finite element method and evolutionary techniques / M. Vasundara, K. P. Padmanaban // The International Journal of Advanced Manufacturing Technology. – 2014. – V. 70, No. 1-4. – P. 79 – 96.

31. Ильин, В. Н. Технология автоматизации структурно-параметрического синтеза на основе метода морфологического ящика / В. Н. Ильин, А. В. Лепехин // Эл. журнал «Труды МАИ». – 2012 г. – URL : http://www. mai. ru/upload/iblock/0a8/0a88fa19b392492a3f875e1ea1a7f1f4. pdf. – 2011.

32. Rakov, D. L. The structural analysis of new technical systems based on a morphological approach under uncertainty conditions / D. L. Rakov, A. V. Sinyev // Journal of Machinery Manufacture and Reliability. – 2015. – V. 44, No. 7. – P. 650 – 657.

33. Прошин, И. А. Структурно-параметрический синтез математических моделей объектов исследования по экспериментальным данным / И. А. Прошин, Д. И. Прошин, Р. Д. Прошина // Вестник Астраханского государственного технического университета. Сер. Морская техника и технология. – 2009. – № 1.

34. Жихарев, А. Г. Современные способы представления знаний: проблемы, перспективы развития / А. Г. Жихарев, Р. А. Маматов // Научные ведомости Белгородского государственного университета. Сер. Экономика. Информатика. – 2011. – Т. 19, № 13-1(108).

35. Третьякова, Н. В. Особенности обработки информации о материальных потоках с помощью математического моделирования / Н. В. Третьякова // Информационные ресурсы России. – 2017. – № 4. – С. 37 – 41.

36. Могилев, А. В. Технологии обработки текстовой информации. Технологии обработки графической, аудио-, видеоинформации и мультимедиа / А. В. Могилев. – БХВ-Петербург, 2010.

37. Постников, А. А. Формализация процессов обработки заявок в информационных системах / А. А. Постников, А. Ю. Крупский // Интернет-журнал Науковедение. – 2011. – № 3(8). – С. 1 – 11.

38. Kleinrock, L. Queueing systems. Volume I: theory. / L. Kleinrock // New York : Wiley Interscience, 1975. – 417 p.

39. Приходько, М. А. Оптимизация процесса обработки информации в узлах распределенной мультиагентной системе обработки разнородной информации / М. А. Приходько // Горный информационно-аналитический бюллетень (научно-технический журнал). – 2011. – № S6. – С. 257 – 262.

40. Leite, J. B. Development of a self-healing strategy with multiagent systems for distribution networks / J. B. Leite, J. R. S. Mantovani // IEEE Transactions on Smart Grid. – 2016. – V. 8, No. 5. – P. 2198 – 2206.

41. Гузаиров, М. Б. Оценка живучести аппаратно-программных комплексов по среднему значению показателя целевой эффективности / М. Б. Гузаиров, В. Е. Гвоздев, А. С. Давлиева, В. В. Тесленко // Информационные и математические технологии в науке и управлении. – 2018. – № 1(9). – С. 106 – 113.

42. Зяблов, Д. В. Применение микросервисной архитектуры при разработке корпоративных веб-приложений / Д. В. Зяблов, А. А. Кот // Электронный научный журнал. – 2017. – С. 17.

43. Обухов, А. Д. Постановка задачи структурно-параметрического синтеза системы электронного документооборота научно-образовательного учреждения / А. Д. Обухов // Вестник ТГТУ. – 2016. – № 2. – С. 217 – 232.

44. Системный анализ и формализация структуры адаптивных тренажерных комплексов эргатических систем / М. Н. Краснянский, Д. Л. Дедов, А. Д. Обухов, С. Ю. Алексеев // Вестник компьютерных и информационных технологий. – 2019. – № 4. – С. 45 – 52.

45. Перегудов, Ф. И. Основы системного анализа / Ф. И. Перегудов, Ф. П. Тарасенко. – Томск : Изд-во НТЛ, 1997. – 396 с.

46. Ожерельева, Т. А. Структурный анализ систем управления / Т. А. Ожерельева // Государственный советник. – 2015. – № 1(9).

47. Салимханова, С. А. Функциональный анализ и проектирование корпоративных информационных систем / С. А. Салимханова, Б. Ш. Дадаева // Ответственный редактор. – 2019. – С. 22.

48. Клячкин, В. Н. Статистические методы анализа данных / В. Н. Клячкин, Ю. Е. Кувайскова, В. А. Алексеева. – Ульяновск : УлГТУ, 2016. – 123 с.

49. Краснянский, М. Н. Методика классификации и обработки документов в системе управления электронным документооборотом научно-образовательного учреждения / М. Н. Краснянский, А. Д. Обухов // Вопросы современной науки и практики. Ун-т им. В. И. Вернадского. – 2018. – № 2(68). – С. 203 – 216.

50. Mathematical Model of Information Processing in Electronic Document Management System / A. D. Obukhov, M. N. Krasnyansky, D. L. Dedov, S. V. Karpushkin // International Review of Automatic Control. – 2018. – V. 11, No. 6. – P. 336 – 345.

51. Круковский, М. Ю. Автоматно-графовая формальная модель композитного документооборота / М. Ю. Круковский // ММС. – 2006. – № 2. – С. 87 – 95.

52. Маслобоев, А. В. Формальные спецификации активных программных компонентов мультиагентной виртуальной бизнес-среды развития инноваций / А. В. Маслобоев // Научно-технический вестник информационных технологий, механики и оптики. – 2010. – № 3(67). – С. 93 – 102.

53. Проектирование информационных систем управления документооборотом научно-образовательных учреждений: монография / М. Н. Краснянский, С. В. Карпушкин, А. В. Остроух, А. Д. Обухов и др. – Тамбов : Изд-во ФГБОУ ВПО «ТГТУ», 2015. – 216 с.

54. Formulation of the Problem of Structural and Parametric Synthesis of Electronic Document Management System of Research and Education Institution / M. N. Krasnyanskiy, A. V. Ostroukh, S. V. Karpushkin, A. D. Obukhov // Global Journal of Pure and Applied Mathematics. ISSN 0973-1768. – 2016. – V. 12, No. 3. – P. 2395 – 2409.

55. Болотов, Д. Н. Методические подходы к определению величины экономического эффекта от применения оптимизационных моделей для сокращения затрат на выполнение международных банковских переводов / Д. Н. Болотов // Вопросы современной экономики. – 2013. – № 3. – С. 58 – 72.

56. Этингоф, Е. В. Оценка затрат на информационные системы / Е. В. Этингоф // Управление экономическими системами: электронный научный журнал. – 2013. – № 12(60). – С. 1 – 15.

57. Дьячковская, А. Н. Оценка экономической эффективности информационных систем управления в вузе / А. Н. Дьячковская, М. А. Иванова // Научное сообщество студентов XXI столетия. Экономические науки. – 2018. – С. 76 – 80.

58. Ермаков, А. В. Оценка экономической эффективности использования мультисервисной информационной системы вуза / А. В. Ермаков // Сетевой научно-практический журнал. Сер. Экономические исследования. – 2015. – № 3. – С. 86 – 94.

59. Чуркин, Г. М. Использование процессорного подхода при формировании критериев оценки свойств автоматизированной системы управления технологическим процессом / Г. М. Чуркин, Н. О. Пинюгин // Вестник Саратовского государственного технического университета. – 2016. – Т. 1, № 1(82).

60. Егоров, А. В. Количественные показатели оценки информационных систем / А. В. Егоров, А. С. Грищенко, В. Ф. Кузнецова // Наука и Мир. – 2016. – Т. 1, № 1. – С. 48 – 50.

61. Липаев, В. В. Качество крупномасштабных программных средств / В. В. Липаев. – Directmedia, 2015. – 231 с.

62. Липаев, В. В. Выбор и оценивание характеристик качества программных средств. Методы и стандарты / В. В. Липаев. – М., 2001. – 228 с.

63. Ажмухамедов, И. М. Комплексный критерий оценки качества информационных систем / И. М. Ажмухамедов, О. М. Князева // Актуальные проблемы гуманитарных и естественных наук. – 2017. – № 4-6. – С. 14 – 17.

64. Бубарева, О. А. Оценка качества информационных систем с распределенной архитектурой / О. А. Бубарева // Южно-Сибирский научный вестник. – 2017. – № 4. – С. 263 – 266.

65. Бочкарев, А. М. Критерии оценки системы информационного обеспечения производственной деятельности промышленных предприятий / А. М. Бочкарев // Вестник УГНТУ. Наука, образование, экономика. Сер. Экономика. – 2019. – № 1(27).

66. Рыков, А. А. Многокритериальная оценка качества информационных систем при неопределенности / А. А. Рыков, А. С. Рыков // Проблемы управления. – 2004. – № 2. С. 31 – 39.

67. Алексеев, С. Ю. Влияние формы представления элементов технической системы в программном обеспечении на сложность его алгоритмического обеспечения / С. Ю. Алексеев // Прикаспийский журнал: управление и высокие технологии. – 2018. – № 4. – С. 45 – 56.

68. Kuznetsov, M. A. Analysis of complexity metrics of a software code for obfuscating transformations of an executable code / M. A. Kuznetsov, V. O. Surkov // IOP Conference Series: Materials Science and Engineering. – 2016. – V. 155, No. 1. – P. 012008.

69. Романов, В. Ю. Анализ объектно-ориентированных метрик для проектирования архитектуры программного обеспечения / В. Ю. Романов // International Journal of Open Information Technologies. – 2014. – V. 2, No. 3.

70. Маевский, Д. А. Оценка количества дефектов программного обеспечения на основе метрик сложности / Д. А. Маевский // Electrotechnic and Computer Systems. – 2012. – № 7(83). – С. 113 – 120.

71. Найханова, Л. В. Расчет сложности программного продукта методом функциональных точек / Л. В. Найханова, С. В. Дамбаева, М. А. Пыкин // Научные исследования. – 2017. – Т. 1, № 6(17). – С. 12 – 16.

72. Оценка сроков проекта с использованием модификации метода PERT / D. Y. Blagovisny et al. // Theoretical & Applied Science. – 2019. – No. 12. – P. 331 – 335.

73. Тютюнников, Н. Н. Оценка трудозатрат на создание программных средств для стадии разработки по модели COCOMO II / Н. Н. Тютюнников // Современные тенденции в экономике и управлении: новый взгляд. – 2014. – № 25. – С. 69 – 75.

74. Садовский, И. Д. Применение модели COCOMO II для оценки разработки программного обеспечения в Windows проектах / И. Д. Садовский // Экономика и бизнес: теория и практика. – 2016. – № 10. – С. 102 – 106.

75. COCOMO [Электронный ресурс]. – URL : https://ru.wikipedia.org/wiki/COCOMO

76. Stephens R. Essential Algorithms: A Practical Approach to Computer Algorithms Using Python and C. – Wiley, 2019.

77. Weyuker, E. J. Evaluating software complexity measures / E. J. Weyuker // IEEE transactions on Software Engineering. – 1988. – V. 14, No. 9. – P. 1357 – 1365.

78. Savchenko, D. Code quality measurement: Case study / D. Savchenko, T. Hynninen, O. Taipale // 2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO). – IEEE, 2018. – P. 1455 – 1459.

79. Ланемец, В. В. Оценка результатов решения задач по программированию с использованием метрики программного кода / В. В. Ланемец, П. Н. Воробкалов // Известия Волгоградского государственного технического университета. – 2011. – № 9. – С. 122 – 125.

80. Семахин, А. М. Методы верификации и оценки качества программного обеспечения : учебное пособие / А. М. Семахин. – Курган : Изд-во КГУ, 2018. – 150 с.

81. Звездин, С. В. Проблемы измерения качества программного кода / С. В. Звездин // Вестник Южно-Уральского государственного университета. Сер. Компьютерные технологии, управление, радиоэлектроника. – 2010. – № 2(178).

82. К вопросу о метриках трудоемкости разработки мобильных приложений / И. В. Евдокимов и др. // Фундаментальные исследования. – 2017. – № 9-1. – С. 54 – 58.

83. Ракитский, А. А. Использование вычислительной способности как характеристики для оценки и сравнения суперкомпьютеров / А. А. Ракитский // Вестник СибГУТИ. – 2013. – № 4. – С. 67 – 84.

84. Душкин, А. В. Аппаратное обеспечение компьютерных систем. Гл. II / А. В. Душкин, А. С. Кравченко // Информатика и информационные технологии в профессиональной деятельности. – 2016. – С. 53 – 99.

85. Model-based compatibility checking of system modifications / A. Poetzsch-Heffter et al. // International Symposium On Leveraging Applications of Formal Methods, Verification and Validation. – Springer, Berlin, Heidelberg, 2012. – P. 97 – 111.

86. Mikhail, K. Using Statistical Analysis to Fine-Tune the Results of Knapsack-Based Computational Platform Benchmarking / K. Mikhail, B. Georgii, K. Natalia // 2019 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus). – IEEE, 2019. – P. 1816 – 1820.

87. Performance Evaluation of Docker Container and Virtual Machine / A. M. Potdar et al. // Procedia Computer Science. – 2020. – V. 171. – P. 1419 – 1428.

88. Futuremark [Электронный ресурс]. – URL : http://www.futuremark.com/

89. Jeon, H. A performance measurement framework of cloud storage services / H. Jeon, Y. G. Min, K. K. Seo // Indian Journal of Science and Technology. – 2015. – V. 8. – P. 105 – 111.

90. Маковий, К. А. Использование генетического алгоритма для выбора серверных ресурсов при внедрении инфраструктуры виртуальных рабочих столов в вузе / К. А. Маковий, Ю. В. Хицкова, А. И. Шашкин // Актуальные проблемы прикладной математики, информатики и механики. – 2017. – С. 257 – 263.

91. Smith H. Data center storage: cost-effective strategies, implementation, and management. – CRC press, 2016.

92. Бражнев, С. М. Модель параметрической настройки индекса производительности информационных систем / С. М. Бражнев, И. П. Шепеть, И. С. Кареев // Инновационные направления развития в образовании, экономике, технике и технологиях. – 2019. – С. 130 – 134.

93. Беккалиева, Н. К. Оценка эффективности экономических затрат на развитие информационной системы в системе экономической безопасности компании (бизнеса) / Н. К. Беккалиева // Экономика и современный менеджмент: теория, методология, практика. – 2018. – С. 167 – 171.

94. Краснянский, М. Н. Математическая модель обработки информации в системе управления электронным документооборотом / М. Н. Краснянский, А. Д. Обухов, И. Л. Коробова // Вестник Тамбовского государственного технического университета. – 2018. – Т. 24, № 3. – С. 382 – 399.

95. Скрипкин, К. Экономическая эффективность информационных систем в России / К. Скрипкин. – Litres, 2017.

96. Ханаева, Г. А. Экономическая эффективность информационной системы / Г. А. Ханаева // Инновации и инвестиции. – 2020. – № 5. – С. 140 – 143.

97. Азаров, А. Е. Разработка проекта информационной системы для организации видеонаблюдения / А. Е. Азаров, Р. И. Баженов // Постулат. – 2018. – № 12. – С. 1 – 11.

98. Структурно-параметрический синтез системы поддержки принятия решений при проектировании и эксплуатации тепло- и массообменного оборудования / Е. Н. Малыгин и др. // Вестник Тамбовского государственного технического университета. – 2019. – Т. 25, № 3. – С. 350 – 359.

99. Анализ эффективности применения искусственных нейронных сетей для решения задач распознавания, сжатия и прогнозирования / А. А. Талалаев и др. // Искусственный интеллект и принятие решений. – 2008. – Т. 2. – С. 24 – 33.

100. Прогнозирование бизнес-процессов на основе нейронных сетей / Р. Н. Гайнуллин и др. // Вестник Казанского технологического университета. – 2017. – Т. 20, № 3. – С. 121 – 124.

101. Тетерин, Д. А. Обзор применения искусственных нейронных сетей в управлении социальными и экономическими системами / Д. А. Тетерин, Р. Ш. Хабибулин, С. В. Гудин // Научные ведомости Белгородского государственного университета. Сер. Экономика. Информатика. – 2018. – Т. 45, № 3. – С. 574 – 583.

102. Флах, П. Машинное обучение. Наука и искусство построения алгоритмов, которые извлекают знания из данных / П. Флах. – М. : ДМК Пресс, 2015. – 400 с.

103. A Bayesian classification approach using class-specific features for text categorization / B. Tang et al. // IEEE Transactions on Knowledge and Data Engineering. – 2016. – V. 28, No. 6. – P. 1602 – 1606.

104. Adeniyi, D. A. Automated web usage data mining and recommendation system using K-Nearest Neighbor (KNN) classification method / D. A. Adeniyi, Z. Wei, Y. Yongquan // Applied Computing and Informatics. – 2016. – V. 12, No. 1. – P. 90 – 108.

105. Jun, S. Document clustering method using dimension reduction and support vector clustering to overcome sparseness / S. Jun, S. S. Park, D. S. Jang // Expert Systems with Applications. – 2014. – V. 41, No. 7. – P. 3204 – 3212.

106. Pliakos, K. Global multi-output decision trees for interaction prediction / K. Pliakos, P. Geurts, C. Vens // Machine Learning. – 2018. – P. 1 – 25.

107. Гладких, Т. В. Обнаружение аномалий–обучение без учителя / Т. В. Гладких, Т. Гнот, В. Сольский // Вимірювальна та обчислювальна техніка в технологічних процесах. – 2016. – № 1. – С. 148 – 158.

108. TensorFlow: A System for Large-Scale Machine Learning / M. Abadi et al. // OSDI. – 2016. – V. 16. – P. 265 – 283.

109. Van Veen, F. & Leijnen, S. (2019). The Neural Network Zoo. – URL : http://www.asimovinstitute.org/neural-network-zoo

110. Попова, Е. С. Использование искусственных нейронных сетей для решения задачи классификации текста / Е. С. Попова, В. Г. Спицын, Ю. А. Иванова // Труды Международной конференции по компьютерной графики и зрению «Графикон». – 2019. – № 29. – С. 270 – 273.

111. Du, M., Li, F., Zheng, G., & Srikumar, V. (2017, October). Deeplog: Anomaly detection and diagnosis from system logs through deep learning. In Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security (pp. 1285 – 1298).

112. Gupta, A., Jain, A., Yadav, S., & Taneja, H. (2018). Literature survey on detection of web attacks using machine learning. International Journal of Scientific Research Engineering & Information Technology, 3, 1845 – 1853.

113. Qian, L., Yu, J., Zhu, G., Mei, F., Lu, W., Ge, B., ... & Chen, H. (2019, December). A Duplication Reduction Approach for Unstructured Data Using Machine Learning Method. In 2019 International Conference on Intelligent Computing, Automation and Systems (ICICAS) (pp. 515 – 519). IEEE.

114. The algorithm of document routing in the electronic document management system using machine learning methods / A. D. Obukhov et al. // Proceedings of the 18th International Multidisciplinary Scientific Geo Conference. – 2018. – V. 2.1. – P. 765 – 772.

115. Cybenko, G. V. Approximation by Superpositions of a Sigmoidal function // Mathematics of Control Signals and Systems. – 1989. – V. 2, No. 4. – P. 303 – 314.

116. Hecht-Nielsen, R. Replicator neural networks for universal optimal source coding / R. Hecht-Nielsen // Science. – 1995. – V. 269, No. 5232. – P. 1860 – 1863.

117. Kolmogorov, A. N. On the representation of continuous functions of many variables by superposition of continuous functions of one variable and addition / A. N. Kolmogorov // Doklady Akademii Nauk. – Russian Academy of Sciences, 1957. – V. 114, No. 5. – P. 953 – 956.

118. MaeSTrO: A mobile app for style transfer orchestration using neural networks / M. Reimann et al. // 2018 International Conference on Cyberworlds (CW). – IEEE, 2018. – P. 9 – 16.

119. Акинин, М. В. Нейросетевые системы искусственного интеллекта в задачах обработки изображений / М. В. Акинин, М. Б. Никифоров, А. И. Таганов. – 2015. – 154 с.

120. Forsyth, P. A. I. Neural Networks for Audio: a Survey and Case Study / P. A. I. Forsyth. – 2018. – 67 p.

121. Моисеева, Е. Д. Проблемы обработки видеопотока нейросетями RESNET / Е. Д. Моисеева // Избранные вопросы науки XXI века. – 2019. – С. 136 – 139.

122. Nagarajaiah, S. Modeling and harnessing sparse and low-rank data structure: a new paradigm for structural dynamics, identification, damage detection, and health monitoring / S. Nagarajaiah, Y. Yang // Structural Control and Health Monitoring. – 2017. – V. 24, No. 1. – P. e1851.

123. Jain N., Porwal R. Automated Test Data Generation Applying Heuristic Approaches—A Survey // Software Engineering. – Springer, Singapore, 2019. – P. 699 – 708.

124. Data synthesis based on generative adversarial networks / N. Park et al. // Proceedings of the VLDB Endowment. – 2018. – V. 11, No. 10. – P. 1071 – 1083.

125. Infogan: Interpretable representation learning by information maximizing generative adversarial nets / X. Chen et al. // Advances in neural information processing systems. – 2016. – P. 2172 – 2180.

126. Beltramelli, T. Pix2code: Generating code from a graphical user interface screenshot / T. Beltramelli // Proceedings of the ACM SIGCHI Symposium on Engineering Interactive Computing Systems. – ACM, 2018. – P. 3.

127. Gheyas, I. A. A neural network-based framework for the reconstruction of incomplete data sets / I. A. Gheyas, L. S. Smith // Neurocomputing. – 2010. – V. 73, No. 16-18. – P. 3039 – 3065.

128. Сташкова, О. В. Использование искусственных нейронных сетей для восстановления пропусков в массиве исходных данных / О. В. Сташкова, О. В. Шестопал // Известия высших учебных заведений. Северо-Кавказский регион. Технические науки. – 2017. – № 1(193).

129. Tay C. K. L., Shim K. J. A Cloud-Based Data Gathering and Processing System for Intelligent Demand Forecasting // 2018 IEEE International Conference on Big Data (Big Data). – IEEE, 2018. – P. 5451 – 5453.

130. Jin H., Song Q., Hu X. Auto-keras: An efficient neural architecture search system // Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. – 2019. – P. 1946 – 1956.

131. Regularized evolution for image classifier architecture search / E. Real et al. // Proceedings of the aaai conference on artificial intelligence. – 2019. – V. 33. – P. 4780 – 4789.

132. Automated Machine Learning Overview / R. Budjač et al. // Research Papers Faculty of Materials Science and Technology Slovak University of Technology. – 2019. – V. 27, No. 45. – P. 107 – 112.

133. Efficient and robust automated machine learning / M. Feurer et al. // Advances in neural information processing systems. – 2015. – P. 2962 – 2970.

134. Autogan: Neural architecture search for generative adversarial networks / X. Gong et al. // Proceedings of the IEEE International Conference on Computer Vision. – 2019. – P. 3224 – 3234.

135. Ha, D. World models / D. Ha, J. Schmidhuber // arXiv preprint arXiv:1803.10122. – 2018.

136. Горкун, О. П. Оценка качества работы алгоритма машинного обучения / О. П. Горкун // Актуальные проблемы и пути развития энергетики, техники и технологий. – 2019. – С. 103 – 107.

137. Borji, A. Pros and cons of gan evaluation measures // Computer Vision and Image Understanding / A. Borji. – 2019. – V. 179. – P. 41 – 65.

138. Improved techniques for training gans / T. Salimans et al. // Advances in neural information processing systems. – 2016. – P. 2234 – 2242.

139. Rethinking the inception architecture for computer vision / C. Szegedy et al. // Proceedings of the IEEE conference on computer vision and pattern recognition. – 2016. – P. 2818 – 2826.

140. Kwitt, R. Image similarity measurement by Kullback-Leibler divergences between complex wavelet subband statistics for texture retrieval /

R. Kwitt, A. Uhl // 2008 15th IEEE International Conference on Image Processing. – IEEE, 2008. – P. 933 – 936.

141. Gans trained by a two time-scale update rule converge to a local nash equilibrium / M. Heusel et al. //Advances in neural information processing systems. – 2017. – P. 6626 – 6637.

142. Barratt, S. A note on the inception score / S. Barratt, R. Sharma // arXiv preprint arXiv:1801.01973. – 2018.

143. Обухов, А. Д. Нейросетевая архитектура информационных систем / А. Д. Обухов, М. Н. Краснянский // Вестник Удмуртского университета. Математика. Механика. Компьютерные науки. – 2019. – Т. 29, вып. 3. – С. 438 – 455.

144. Object-oriented model for life cycle management of electrical instrumentation control projects / J. Zhou et al. // Automation in Construction. – 2015. – V. 49. – P. 142 – 151.

145. Hierarchical spatio-temporal visual analysis of cluster evolution in electrocorticography data / S. Murugesan et al. // Proceedings of the 7th ACM International Conference on Bioinformatics, Computational Biology, and Health Informatics. – ACM, 2016. – P. 630 – 639.

146. Kirikova, M. Viable systems model based information flows / M. Kirikova, M. Pudane // New Trends in Databases and Information Systems. – Springer, Cham, 2014. – P. 97 – 104.

147. Remote working and collaboration in agile teams / A. Deshpande et al. – 2016.

148. Flexible work practices: analysis from a pragmatist perspective / J. Brandl et al. // Historical Social Research/Historische Sozialforschung. – 2019. – V. 44, No. 1(167). – P. 73 – 91.

149. Terzi, D. S. Big data analytics for network anomaly detection from netflow data / D. S. Terzi, R. Terzi, S. Sagiroglu // 2017 International Conference on Computer Science and Engineering (UBMK). – IEEE, 2017. – P. 592 – 597.

150. Arcos-Garcia, A. Evaluation of deep neural networks for traffic sign detection systems / A. Arcos-Garcia, J. A. Alvarez-Garcia, L. M. Soria-Morillo // Neurocomputing. – 2018. – V. 316. – P. 332 – 344.

151. Stroud, R. O. Application of Cyclomatic Complexity in Enterprise Architecture Frameworks / R. O. Stroud, A. Ertas, S. Mengel // IEEE Systems Journal. – 2019. – V. 13, No. 3. – P. 2166 – 2176.

152. Pan, W. Measuring software stability based on complex networks in software / W. Pan, C. Chai // Cluster Computing. – 2019. – V. 22, No. 2. – P. 2589 – 2598.

153. Stasiak, A. Software Metrics for Similarity Determination of Complex Software Systems / A. Stasiak, J. Chudzikiewicz, Z. Zieliński //

KKIO Software Engineering Conference. – Springer, Cham, 2018. – P. 175 – 191.

154. Hovorushchenko, T. Ontology-Based Intelligent Agent for Determination of Sufficiency of Metric Information in the Software Requirements / T. Hovorushchenko, O. Pavlova, D. Medzatyi // International Scientific Conference "Intellectual Systems of Decision Making and Problem of Computational Intelligence". – Springer, Cham, 2019. – P. 447 – 460.

155. Manikavelan, D. Software quality analysis based on cost and error using fuzzy combined COCOMO model / D. Manikavelan, R. Ponnusamy // Journal of Ambient Intelligence and Humanized Computing. – 2020. – P. 1 – 11.

156. Fadhil, A. A. Software Cost Estimation Based on Dolphin Algorithm / A. A. Fadhil, R. G. H. Alsarraj, A. M. Altaie // IEEE Access. – 2020. – V. 8. – P. 75279 – 75287.

157. Applicability of the software cost model COCOMO II to HPC projects / J. Miller et al. // International Journal of Computational Science and Engineering. – 2018. – V. 17, No. 3. – P. 283 – 296.

158. Du K. L. Elements of Computational Learning Theory / K. L. Du, M. N. S. Swamy // Neural Networks and Statistical Learning. – Springer, London, 2019. – P. 65 – 79.

159. Fronza, I. An Approach to Evaluate the Complexity of Block-Based Software Product / I. Fronza, L. Corral, C. Pahl // Informatics in Education. – 2020. – V. 19, No. 1. – P. 15 – 32.

160. Alwosheel, A. Is your dataset big enough? Sample size requirements when using artificial neural networks for discrete choice analysis / A. Alwosheel, S. van Cranenburgh, C. G. Chorus // Journal of choice modelling. – 2018. – V. 28. – P. 167 – 182.

161. A public domain dataset for human activity recognition using smartphones / D. Anguita et al. // Esann. – 2013.

162. Classification of epileptic EEG recordings using signal transforms and convolutional neural networks / R. San-Segundo et al. // Computers in biology and medicine. – 2019. – V. 109. – P. 148 – 158.

163. Chujai, P. Time series analysis of household electric consumption with ARIMA and ARMA models / P. Chujai, N. Kerdprasop, K. Kerdprasop // Proceedings of the International MultiConference of Engineers and Computer Scientists. – 2013. – V. 1. – P. 295 – 300.

164. Kohavi, R. Scaling up the accuracy of naive-bayes classifiers: A decision-tree hybrid / R. Kohavi // Kdd. – 1996. – V. 96. – P. 202 – 207.

165. Brock, A. Large scale gan training for high fidelity natural image synthesis / A. Brock, J. Donahue, K. Simonyan // arXiv preprint arXiv:1809.11096. – 2018.

166. Label-removed generative adversarial networks incorporating with K-Means / C. Wang et al. // Neurocomputing. – 2019. – V. 361. – P. 126 – 136.

167. Chong, M. J. Effectively Unbiased FID and Inception Score and where to find them / M. J. Chong, D. Forsyth // arXiv preprint arXiv:1911.07023. – 2019.

168. Юнусова, Л. Р. Построение классификатора / Л. Р. Юнусова, А. Р. Магсумова // Научный журнал. – 2020. – № 2(47). – С. 17 – 19.

169. Герасименко, Е. М. Интеллектуальный анализ данных. Алгоритмы data mining / Е. М. Герасименко. – Таганрог : Изд-во Южного федерального университета. – 2017. – 82 с.

170. Курако, Е. А. Системы объектно-связанного документооборота и организация их взаимодействия с бизнес-процессами / Е. А. Курако, В. Л. Орлов // Проблемы управления. – 2017. – № 2. – С. 42 – 49.

171. Churikov, N. Experience of using machine learning methods in document processing system / N. Churikov // Процессы управления и устойчивость. – 2017. – V. 4, No. 1. – P. 548 – 554.

172. Белых, Е. А. Подход к проектированию языка подстановок для генерации электронных документов, содержащих сложные таблицы / Е. А. Белых, Ю. В. Гольчевский // Вестник Удмуртского университета. Математика. Механика. Компьютерные науки. – 2019. – Т. 29, № 3. – С. 422 – 437.

173. A. Soudani, W. Barhoumi. Flotation froth image recognition with convolutional neural networks, Minerals Engineering. – 2019. – V. 132. – P. 183 – 190.

174. Y. Fu, C. Aldrich. An image-based segmentation recommender using crowdsourcing and transfer learning for skin lesion extraction, Expert Systems with Applications. – 2019. – V. 118. – P. 400 – 410.

175. M. Wang, X. Liu, H. Jin. A generative image fusion approach based on supervised deep convolution network driven by weighted gradient flow, Image and Vision Computing. – 2019. – V. 86. – P. 1 – 16.

176. S. Liu, G. Tian, Y. Xu. A novel scene classification model combining ResNet based transfer learning and data augmentation with a filter, Neurocomputing. – 2019. – V. 338. – P. 191 – 206.

177. Z. Wu, C. Shen, A. Hengel. Wider or Deeper: Revisiting the ResNet Model for Visual Recognition, Pattern Recognition. – 2019. – V. 90. – P. 119 – 133.

178. Иванов, А. В. Исследование комплексных адаптивных алгоритмов обработки информации для навигационных систем подвижных наземных объектов / А. В. Иванов, С. П. Москвитин, В. О. Сурков // Вестник Тамбовского государственного технического университета. – 2019. – Т. 25, № 3. – С. 395 – 405.

ОГЛАВЛЕНИЕ

Научное издание

ОБУХОВ Артем Дмитриевич
КРАСНЯНСКИЙ Михаил Николаевич

СТРУКТУРНО-ПАРАМЕТРИЧЕСКИЙ СИНТЕЗ АДАПТИВНЫХ ИНФОРМАЦИОННЫХ СИСТЕМ НА ОСНОВЕ НЕЙРОСЕТЕВЫХ МЕТОДОВ И АРХИТЕКТУРЫ

Монография

Редактор Л. В. К о м б а р о в а
Инженер по компьютерному макетированию И. В. Е в с е е в а

ISBN 978-5-8265-2353-7

Подписано в печать 30.04.2021.
Выход в свет 25.06.2021.
Формат 60×84/16. 13,95 усл. печ. л.
Тираж 400 экз. (1-й з-д 50). Заказ № 19А

Издательский центр ФГБОУ ВО «ТГТУ»
392000, г. Тамбов, ул. Советская, д. 106, к. 14.
Тел. 8(4752) 63-81-08.
E-mail: izdatelstvo@tstu.ru

Отпечатано в Типографии ФГБОУ ВО «ТГТУ»
392008, г. Тамбов, ул. Мичуринская, д. 112А.
Тел. 8(4752) 63-07-46.
E-mail: tipo_tstu68@mail.ru