Международный университет информационных технологий

УДК 004.912

На правах рукописи

МЕРЕМБАЕВ ТИМУР ЖУМАКАНОВИЧ

Разработка программно-аппаратных инструментариев для мониторинга технических систем

Специальность 6D070400 - «Вычислительная техника и программное обеспечение»

Диссертация на соискание учёной степени доктора философии (PhD)

Научный руководитель:
д.т.н., профессор Амиргалиев Е.Н.
Зарубежный руководитель:
Senior Full Professor Ружанский М.
Гент Университет, Бельгия

Республика Казахстан
Алматы - 2022

СОДЕРЖАНИЕ

ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ

ОДУ - Обыкновенные дифференциальные уравнения

RNN - Recurrent neural network (Рекуррентные нейронные сети)

LSTM - Long short-term memory (Длинная цепь элементов краткосрочной памяти)

ВИЭ - Возобновляемый источник энергии

CNN - Convolutional neural network (Свёрточная нейронная сеть)

AE - Autoencoder (Автокодировщик)

DL - Deep learning (Глубокое обучение)

ARIMA - Autoregressive integrated moving average (Интегрированная модель авторегрессии - скользящего среднего) IoT - Internet of Things (Интернет вещей)

SQL - Structured Query Language

NoSQL - not only SQL, не только SQL

API - Application Programming Interface

JDBC - Java Database Connectivity

HDFS - Hadoop Distributed File System

RSME - Root mean square error (Среднеквадратичное отклонение)

MAE - Mean absolute error (Средняя абсолютная ошибка)

MAPE - Mean absolute percentage error (Средняя абсолютная ошибка в процентах)

ВВЕДЕНИЕ

Идея работы заключается в создании системы на основе искусственного интеллекта для уменьшения риска появления аварийности или предсказания предаварийного состояния в технических устройствах, в частности на биогазовой установке. В работе рассматривается создание комплексного подхода на основе алгоритмов машинного обучения для технических устройств.

Предсказание аварийности и устойчивой работы технического комплекса является важной задачей в проблеме поддерживания качественной работы различного вида оборудования. Решение данной проблемы логическим образом является экономически выгодным по сравнению с проблемой полной замены, глубокого ремонта технического комплекса или простоя оборудования. Выявление предаварийности технического комплекса заключается в обнаружении отклонения в режимах работы основных компонентов устройства, которые в свою очередь на влияют на КПД работы технического устройства в целом. Современные исследования показали, что мониторинг характеристик компонентов технического устройства по прежнему остается важнейшей задачей для принятия обоснованных решений в области эксплуатации и технического комплекса.

Проблема безопасности для разных типов технических устройств изучается, и получены результаты, которые позволяют определить основные проблемы безопасности, в частности на производстве биогаз. Основными проблемами выделены следующие пункты: защитные меры биогазовых проектов недостаточны, методы мониторинга являются относительно отсталыми, запись данных в реальном времени и статистический анализ не могут быть выполнены, время от времени происходят утечки биогаза и взрывы. Вопросы безопасности при использовании биогаза являются одной из важнейших тем в биогазовой промышленности. Для обеспечения безопасности работников и надлежащего управления биогазовыми проектами необходима система мониторинга в режиме реального времени и в течение длительного времени в различных условиях. Важным компонентом для данной цели является большое количество данных о параметрах работы различных узлов установки.

Одной из проблем анализа данных являются нерегулярные данные, в диссертационной работе предлагается использовать способ построения модели со скрытыми переменными, описывающей генерацию временного ряда на основе дифференциальных уравнений. Данный подход основан на теории обыкновенных дифференциальных уравнений, который позволяет разработать новую архитектуру нейронных сетей. В диссертационной работе основной идей является кастомизация новой архитектуры нейронных сетей в задачах экстраполяции временного ряда и применения в практических задачах предсказания аварийности (задача регрессии) и классификации (классификация аварийности на основе исторических данных). Применение новой архитектуры имеет ряд преимуществ по сравнению существующими подходами машинного обучения:

– Эффективность. В связи с использованием adjoin метода, обучение нейронных дифференциальных уравнений проходит быстрее, так как расчет и хранение градиента можно оптимизировать.

– Гибкость времени работы. В нейронных ОДУ можно находить баланс между численной точностью и вычислительными затратами, что быстрее по сравнению с численными решениями ОДУ и уравнений частных производных.

– Количество параметров. По сравнению с другими архитектурами глубокого обучения, нейросетевые ОДУ используют меньшее количество параметров.

Целью работы является разработка интеллектуальной системы для анализа и оценки аварийности технических систем, построенной на основе современных технологий искусственного интеллекта.

Для достижения поставленной цели необходимо было решить следующие **задачи**:

1. Разработать комплексную платформу по повышению эффективности работы биогазовой установки на основе нейронных дифференциальных уравнений для повышения отказоустойчивости оборудования 5-20%.
2. Построить архитектуру нейронных дифференциальных уравнений для расчета оценки риска аварийности технического устройства с использованием обыкновенного дифференциального уравнения для экстраполяции временных рядов и оценки плотности.
3. Создать автоматизированный комплекс по сбору и обработке данных с биогазовой установки.
4. Определить критерии «оптимального» и предаварийного режима работы и обслуживания объектов, минимизация роли «человеческого фактора».
5. Разработать систему мониторинга и «быстрого» оповещения о внештатных (аварийных) ситуациях в работе объекта.

Объектом исследования являются показания датчиков (временной ряд), события об авариях или отклонения работы технического устройства и методы анализа временных рядов.

Предметом исследования является техническое устройство, в частности биогазовая установка.

Научная новизна: Научная новизна заключается в разработке и в получении следующих выводов:

1. Разработана математическая модель на основе нейронных дифференциальных уравнений для предсказания значений в будущем.
2. Доказана устойчивость нейронных дифференциальных уравнений.
3. Построена комплексная система для мониторинга работоспособности узлов и агрегатов биогазовой установки и выявления предаварийного состояния технических устройств.

Теоретическое и практическое значение работы. Теоретическая значимость данной работы заключается в математическом доказательстве применения математического аппарата ОДУ для создания новой архитектуры нейронных сетей, также в выводе свойства достаточности существования

решения для нейросетевых ОДУ, т.е. отсутствие пересечения интегральных кривых. Практическая значимость работы заключается в апробации нейросетевых ОДУ для задачи предсказания аварийности на биогазовой установке.

Во введении дано обоснование актуальности выбранной темы диссертационной работы. Сформулированы цель, объект, предмет и задачи научно-исследовательской работы. Описаны результаты проведенных исследований, показаны их научная новизна и практическая значимость.

В первом разделе диссертационной работы рассматривается проблема предаварийного состояния технических устройств. Представлено описание объекта исследования, датчики сбора информации об биогазовой установке. Выполнен анализ собранных данных, визуализация на интерактивных графиках. Дается описание математического моделирования состояния объектов биогазовой установки, в частности плоского гелеоколлектора, который является устройством для обогрева биогазовой установки.

Второй раздел содержит описание нового семейства моделей глубоких нейронных сетей, основанных на дифференциальных уравнениях (ОДУ) или нейросетевых ОДУ. В начале раздела формально определено, что такое временной ряд, и что значит делать выводы по временным рядам. Предоставлен обзор классических моделей прогнозирования временных рядов, таких как ARIMA, LSTM и Prophet. Для данных моделей показано, какие слабые стороны они имеют в случае нелинейной динамики происхождения временных данных.

Третий раздел содержит введение в дифференциальные уравнения, которые являются строительным блоком для нейросетевых ОДУ. Математический аппарат дифференциальных уравнений используется для параметризации нейронной сети. В данном разделе описана аппроксимация решения ОДУ с помощью математического аппарата дифференциальных уравнений, решение является в виде «черного ящика». Также описано масштабируемое обратное распространение через решение ОДУ без использования внутренних операций, что позволяет проводить сквозное обучение. Данный подход позволяет построить модели скрытых переменных с непрерывным временем, в которых скрытое состояние прогрессирует как непрерывная функция во времени. В разделе показаны свойства достаточного существования решения для нейросетевых ОДУ, т.е. отсутствие пересечения интегральных кривых.

Четвертый раздел содержит описание предложенной и разработанной архитектуры решения начиная с цифровизации биогазовой установки и заканчивая визуализацией данных на дашборде. В рамках описания архитектуры представлены open source решения для различных этапов сбора, хранения и обработки данных. Используя собранный набор данных с биогазовой установки, было показано, что модели на основе ОДУ являются мощным инструментом для нерегулярных, разреженных и зашумленных временных рядов и превосходят рекуррентные нейронные сети в своих экстраполированных и интерполяционных свойствах.

В заключениях каждой главы изложены основные результаты работы, выводы диссертации и будущие шаги исследования данного направления.

Апробация работы. Основные результаты по теме диссертации изложены в 10 работах, в 3 статьях в журналах, рекомендованных Комитетом по обеспечению качества в сфере науки и высшего образования МНВО РК, в 4х статьях в международных научных изданиях, входящих в базу данных Scopus и в 3х материалах международных зарубежных конференциях: 14-й международной конференции Electronics Computer and Computation (ICECCO), IEEE. Казахстан, СДУ, Каскелен, 2018 г.; Международной конференции «Вычислительные и информационные технологии в науке, технике и образовании», 2020 г.; 13-й международный симпозиум “Intelligent Systems 2018” (INTELS’18), 22-24 Октября, 2018, Санкт-Петербург.

Связь с государственными программами. Рассматриваемые задачи диссертации имеют большое практическое значение и непосредственно связаны с задачами цифровизации производства. Вопросам цифровизации уделяется значительное внимание в выступлении президента Токаева К.К. и в правительственных документах: Стратегии «Казахстан-2050», Государственной программе «Цифровой Казахстан», Послании Главы государства «Единство народа и системные реформы - прочная основа процветания страны» от 1 сентября 2021 года, Концепции перехода Казахстана к «зеленой» экономике, Государственной программе индустриально-инновационного развития Республики Казахстан на 2020-2025 годы, Законе РК «Об энергосбережении», где особое место отводится мерам по вовлечению энергобаланса ВИЭ; Киотскому протоколу по чистому развитию.

Диссертационная работа выполнялась в рамках научно-исследовательских работ по коммерциализации МОН РК 0365-18-ГК – «Производство и реализация биогаза, биоудобрений на базе разработки и построения модульного автоматизированного биогазового комплекса с цифровыми технологиями управления и функционирования» Института информационных и вычислительных технологий Комитета науки Министерства науки и высшего образования Республики Казахстан (ИИВТ МНВО РК), апробация технических, теоретических методов и гипотез выполнена на техническом комплексе – биогазовой установке.

Также в рамках диссертации были выполнены работы по направлению создания датчиков для сбора данных с технических устройств (солнечного коллектора), был получен патент по полезную модель 5591 «Система дистанционного мониторинга солнечных коллекторов» в рамках программно-целевого финансирования исследования BR05236693-ОТ-20 «Математические и компьютерные модели, программно-аппаратные инструментарии и опытно-экспериментальные разработки по созданию сети комбинированных эффективных двухконтурных гелиоколлекторов с термосифонной циркуляцией и мониторинг их функционирования». Результаты исследований по данной диссертации включены в отчеты проектов за 2019 год и включены в итоговый отчет за 2020 год.

Объем и структура работы. Диссертация состоит из введения, четырех глав, заключения, списка использованных источников и четырех приложений. Полный объём диссертации составляет 92 страницы с 48 рисунками и 4 таблицами. Список литературы содержит 102 наименования.

1 ПРОБЛЕМА ВЫЯВЛЕНИЯ ПРЕДАВАРИЙНОГО СОСТОЯНИЯ ТЕХНИЧЕСКОГО УСТРОЙСТВА

1.1 Виды аварийного состояния технического устройства

Предсказание аварийности и устойчивой работы технического устройства является важной задачей в проблеме поддерживания качественной работы различного вида оборудования. Решение данной проблемы логическим образом является экономически выгодным, так как остановка технологического процесса вне цикла ремонтных работ или аварии являются более затратными, чем замена оборудования. Выявления предаварийности технического комплекса заключаются в обнаружении отклонения в режимах работы основных компонентов устройства, которые в свою очередь влияют на КПД технического устройства в целом. Современные исследования показали, что мониторинг характеристик компонентов технического устройства по-прежнему остается важнейшей задачей для принятия обоснованных решений в области эксплуатации технического устройства.

В последние годы использование данных промышленного оборудования в единой интернет платформе позволяет оптимизировать, мониторить и управлять техническими устройствами в удобной форме [1]. Методы анализа, основанные на исторических данных, содержащих эксплуатационную и имитационную историю технического устройства, используются для улучшения понимания сложных нелинейных устройств. Основные операции по техническому обслуживанию основаны на принципах максимизации прибыльности, снижения затрат и увеличения жизненного цикла устройства, а также повышения надежности и доступности. В исследованиях показано, что внедрение подобных цифровых платформ для аналитики и оптимизации может иметь значительную экономию. Примерами такой экономической оптимизации является нефтегазовый и авиационный секторы, например, такие компании как Southwest [2], и Quantas [3] ежегодно экономят миллионы долларов в расходах на топливо за счет партнерства с производителями двигателей и постоянного улучшения компонентов системы. С другой стороны, в секторах нефти и газа экономия, связанная с внедрением цифровых платформ, составляет порядка миллиардов в год и ориентирована на исключения внеплановых простоев [4]. Возможность разработки и внедрения цифровой платформы по мониторингу состояния, диагностике и прогнозированию аварийности лежит в основе трансформации традиционной промышленной среды в ее цифровую эру с огромными перспективами доступности и надежности оборудования.

На данный момент существуют различные методики и подходы по техническому обслуживанию [5-8], которые определяют стратегию технического обслуживания оборудования и перехода от классических методов «сбоя и исправления» к методологии «прогнозирования и предотвращения». При прогнозном техническом обслуживании условия эксплуатации оборудования постоянно контролируются для выявления необходимости проведения технического обслуживания в режиме реального времени. Создание и внедрение

методов автоматического мониторинга состояния, диагностики и прогнозирования позволяет значительно снизить, как экономические потери, вызванные поломкой системы, так и затраты, связанные с ненужным ремонтом, заменой компонентов и простоем устройств.

Большинство исследований по обнаружению аномалий временных рядов протестированы на популярных эталонных наборах данных. Результат статей хорошо работает для этих общедоступных наборов данных. К сожалению, когда идет апробация на реальных данных и с реальными бизнес-задачами, новые подходы не могут достичь высоких значений по бизнес-показателям, производительности или эффективности. Большинство общедоступных наборов данных имеют несколько недостатков, таких как: тривиальность, нереалистичная (симулированная) аномалия и отсутствие недостающих данных. Из-за этих недостатков мы считаем, что для бизнеса необходимо обеспечить сравнение алгоритмов на реальных данных.

Временной ряд - это последовательность данных, полученных на многих устройствах, они могут представлять собой регулярные или нерегулярные интервалы времени и храниться в последовательном порядке во времени; например, измерение производительности скважины с течением времени из системы управления. На основе данных временных рядов извлекаются полезные статистические характеристики (например, тренд, закономерность, аномалия и изменчивость). Созданная модель (прогнозная или классификационная) на основе данных временных рядов используется для оперативной поддержки и принятия решений.

В данной работе апробация технических и теоретических методов и гипотез будет выполнена на техническом устройстве – биогазовой установке.

Проблема безопасности для данных типов технических устройств рассмотрена в нескольких современных работах. Исследования касательно безопасности и аварийности изучены автором Морена, объектом исследования рассматриваются аварии на биогазовых станциях. Авторы выделяют следующие пункты: защитные меры биогазовых проектов недостаточны, методы мониторинга являются относительно отсталыми, запись данных в реальном времени и статистический анализ не могут быть выполнены, время от времени происходят утечки биогаза и взрывы. В работе показано более высокое увеличение аварийности на биогазовых станциях, чем увеличение производства энергии на биогазовых станциях. Оценка риска аварий на биогазовых станциях описана в статье [9], где риски серьезным образом повлияли на продвижение биогазовых проектов. Вопросы безопасности при использовании биогаза являются одной из важнейших тем в биогазовой промышленности [10]. Для обеспечения безопасности пользователей и, одновременно, для надлежащего управления биогазовыми проектами необходима система мониторинга в режиме реального времени и в течение длительного времени в помещении [11]. Важным компонентом для данной цели является большое количество данных о параметрах [12].

На рисунке 1.1 показана схема биогазового комплекса. Основной контроль безопасности будет осуществляться на основе данных из следующих компонентов технического устройства: 1, 2, 4, 7.

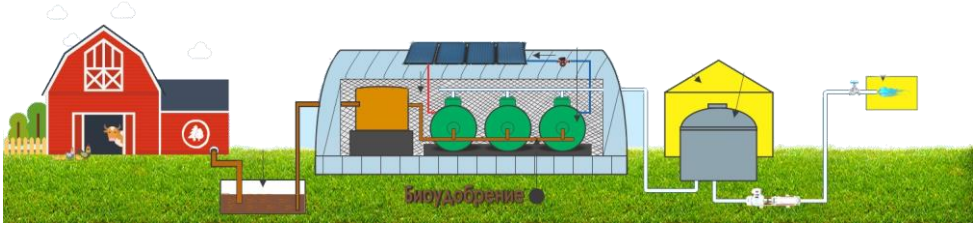

Рисунок 1.1 - Схема биогазового комплекса. 1 – Гелиоколлектор. 2 – Реакторный блок. 3 – Гомогенизатор. 4 – Газгольдер. 5 – Ангар для хранения биогаза. 6 – Ангар для реакторного блока. 7 – Выход биометана и биоудобрения.

Если рассматривать данную проблему в международных масштабах, то по мнению экспертов рост числа аварий в работе биогазовых станций начался на рубеже XXI века, в период активного внедрения современных экологически чистых методов выработки электроэнергии. В результате постоянной эксплуатации технического устройства были случаи аварий на технических устройствах, соответственно каждый раз остро поднимался вопрос предсказывания таких случаев. Одним из примеров является немецкая комиссия по безопасности объектов, которая решила сосредоточить свой контроль только на биогазовых станциях. За 2001-2006 годы комиссия по безопасности объектов провела в общей сложности 115 инспекций [13], и выявила большое количество аварий, которые можно было сократить или предотвратить, используя предсказывающие системы.

Такие аналитические системы на основе искусственного интеллекта для мониторинга сложных технических систем имеют растущий спрос и важны для индустрии в Казахстане, так как в связи со стратегией развития Казахстана будет идти обширная цифровизация производства. Также в связи с внедрением технологии возобновляемых источников в экономику Казахстана, данная система также имеет важное направление развития науки и техники.

В настоящее время один из эффективных способов мониторинга состояния компонентов технического устройства является обнаружение аномальных сигналов, таких как вибрация, давление и электрический ток и другое [14], оцифрованных с технических устройств и собранных индустриальными датчиками. Из таких косвенных, смешанных и зашумленных сигналов часто трудно извлечь полезную информацию, поэтому существует потребность в инструментах анализа данных для извлечения конкретных признаков из этих сложных сигналов.

1.2 Анализ существующих схем биогазовой установки

В Казахстане произошли структурные изменения в сельскохозяйственной деятельности. Одним из толчков к изменениям является переход на новые

рыночные условия деятельности, переход с бывших колхозов и совхозов на мелкие частные фермерские хозяйства, которые стали играть определяющую роль в сельском хозяйстве. Большинство этих хозяйств удалены от элементарного комфорта, в частности от источников электроэнергии и расположены в горных и предгорных районах.

В основном энергообеспечение быта сельских потребителей осуществляется за счет привозного топлива. Стоимость привозного топлива напрямую связана с доставкой, что представляет собой определенные трудности. Жители отдаленных местностей из-за дороговизны, недоступности электроэнергии и привозного топлива вынуждены незаконно вырубать ценные породы лесонасаждений, используя их в качестве дров. В связи с этим фактом, одним из решений является использование возобновляемых источников энергии для отдаленных сельских потребителей, в том числе биогаза, получающегося в процессе анаэробной переработки органических отходов сельского хозяйства.

В настоящее время разрабатываются и внедряются биогазовые установки, которые успешно применяются в фермерских хозяйствах. Одной из проблем на данный момент является подогрев сбраживаемой биомассы с применением электроэнергии, что снижает эффективность их использования в фермерских хозяйствах, расположенных в регионах, где отсутствует доступ к энергоснабжению.

Поэтому разработка новых автономных биоэнергетических установок является актуальной проблемой, решение которой привнесет значительный вклад в направление использования возобновляемых источников энергии с обеспечением экологической безопасности в труднодоступных регионах. Социальный и экономический эффект неоспорим. Выполнение данного проекта позволит разработать новые системы автономного энергообеспечения малых населенных пунктов, создаст альтернативу традиционным источникам энергии. Новые подходы к энергообеспечению позволят значительно поднять уровень комфортности и жизнедеятельности людей.

За рубежом такие системы приветствуются, государство помогает населению дотациями и безвозмездными ссудами, зная, что разовые вложения окупятся энергетической безопасностью малых городов и населенных пунктов. Кроме того, проблема обеспечения топливом, особенно в зимний период на порядок снизится.

Авторы [15] описывают использование биомассы в качестве возобновляемого и устойчивого биотопливного сырья, сравнивают различные источники биомассы с точки зрения рентабельности, требований к земле, свойств сырья, экологических выгод и более, а также объясняют процессы и технологии, связанные с преобразованием биомассы в топливные продукты. Также объясняют состояния источников энергии, находящихся на разных стадиях разработки, включая метан, метанол, водород, электричество и бутанол. Также рассматриваются возможные области для новых исследований, которые могут способствовать будущим прорывам на альтернативных видах топлива, что биомасса, полученная из фотосинтеза, является одним из очевидных источников

топлива, и микробные преобразования биомассы имеют большие перспективы. Предоставляется текущее состояние каждого подхода и факторы, ограничивающие его развитие. Уделяется особое внимание доступности и потенциалу ресурсов биомассы, развитию институтов и рынков, технико-экономическим улучшениям и успешным примерам из Европы и развивающихся стран.

Авторы [16, 17] рассказывают о газификации и пиролизе, освещаются некоторые новые результаты по производству синтез-газа методом тепловой плазмы. Предоставляют всесторонний обзор и углубленную техническую информацию обо всех возможных биоэнергетических ресурсах. Интегрируют текущее состояние художественного охвата от сырья к рентабельным процессам конверсии для биотоплива, экономического анализа и экологической политики. В работе [18] авторы в свой книге предоставляют актуальный учет не древесных, лесных остатков, сельскохозяйственной биомассы (натуральных волокон) и энергетических культур вместе с обработкой, свойствами и их применениями. Критически изучают все аспекты биомассы и биоэнергетики. Своевременный сбор исследований среди таких вопросов, как глобальное потепление, высокий уровень потребления энергии и высокая цена на ископаемое топливо. Авторы в исследовании [19] изучают цепочку поставок биомассы для биоэнергетики и био очистки, которая является основным источником для биогазовой установки, похожее направление было изучено в исследовательской работе [20].

Методы по очистке и последующего метанового брожения известны с конца прошлого столетия, первые такие опытные испытания были 1895 году в английском городе Экзетер. Кроме отчистки отходов данный септик производил биогаз, который использовался для освещения улиц. Анаэробный метод обработки отходов долгое время применялся в основном для стабилизации осадков водоочистных станций и отходов животноводства. С началом энергетического кризиса 1970х годов, метод имел внимание, как со стороны ученных, так и индустрии.

Анализ технологических процессов подготовка субстрата, закачивание в биореактор, подогрев до рабочей температуры с интервальным перемешиванием, подогрев биореактора с солнечными коллекторами, осуществление процесса брожения с попутным получением биогаза и получение биоудобрения. Анализ технологических процессов будет осуществлен контрольными системами.

Анализ технологических процессов осуществлен контрольными системами из микропроцессорных блоков с разработанным программным обеспечением. Подогрев реактора осуществляется с газовыми печами с контроллером на базе ядра ATMEL. Температурный режим подогрева полностью контролируется датчиками.

Перемешивание реактора осуществляется с волновыми редукторами. Электродвигатели контролируются с частотным преобразователем, подключенным к часовому механизму от микропроцессорного блока. На рисунке 1.2 показан разрез биореактора с редукторами и датчиками, которые производят

сбор данных о биореакторе. Биогазовая установка состоит из нескольких таких биореакторов, которые позволяют увеличить производство, на рисунке 1.3 показан план расположения нескольких реакторов в комплексе. Основные технологические процессы, такие как брожение с попутным получением биогаза и получение биоудобрения мониторятся с помощью датчиков температуры, давления, кислотности среды, также имеется контроллер и беспроводная передача показаний.

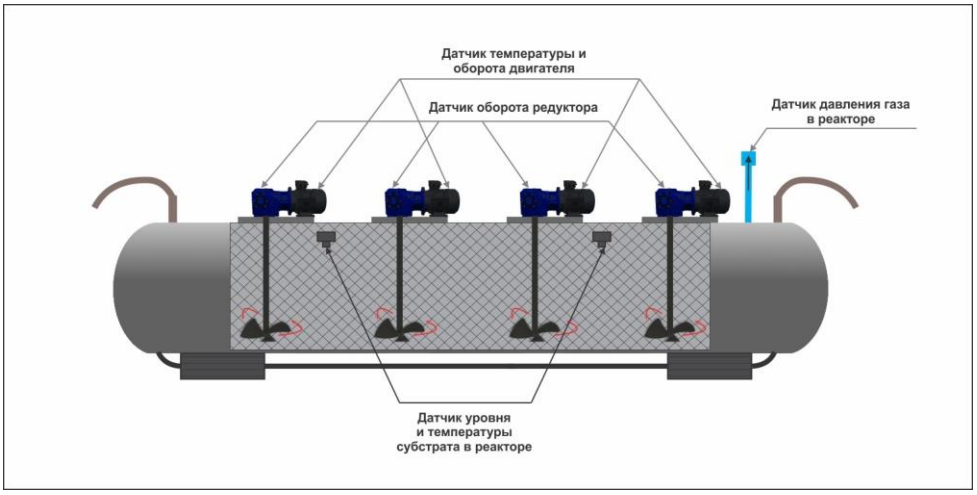

Рисунок 1.2 - Схема биореактора с редукторами и датчиками сбора данных.

Биоустановка, содержащая метан с водяной рубашкой, теплоизоляцией, мешалкой, загрузочным и выгрузочным патрубками, трубопроводы подачи биогаза и газгольдер, текущая биогазовая установка имеет особенность в виде подключенных плоских гелиоколлекторов для подогрева смеси в реакторах, электроводонагревателем и осевым реактивно-центробежным тепловым двигателем и двухроторным генератором на постоянных магнитах. Электрический генератор расположен со стороны днища двигателя биогазовой горелки, которая соединена с трубопроводом для подачи биогаза из газгольдера. В периоды пасмурных дней полученный газ используется в горелке для обогрева сбраживаемой биомассы до необходимой температуры и обеспечения непрерывной работы системы.

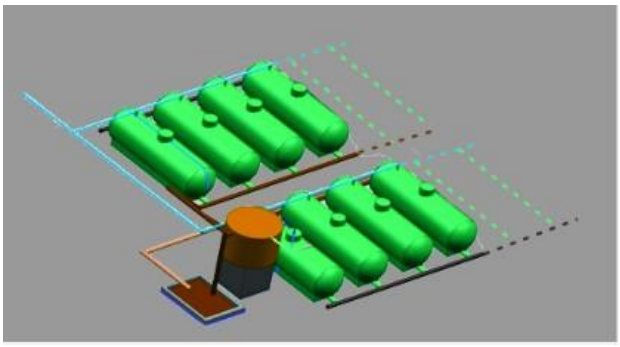

Рисунок 1.3 - Общая иллюстрация комплекса с несколькими биореакторами.

Системы очистки биогаза от примесей углекислого газа, аммиака и сероводорода осуществляются с помощью повышения давления биогаза до 75 атмосфер, с последующим стадийным сжижением аммиака в 8 – 10 атмосфер, 18 - 20 атмосфер сероводорода и 70 - 73 атмосфер углекислого газа. При компримированности (сжатый) газа на низких температурах составляющие биогаза разделяются с температурным режимом и коэффициентом сжатия. Низкая температура осуществляется введением систему повышения давления жидкого азота с температурой охлаждения – 178 градусов Цельсия. Охлажденные с жидким азотом составляющие биогаза разделяются и дают степень очистки метана от примеси до 95 – 98 %.

Проектируется система контроля за технологическими процессами в биореакторе на одноплатном микрокомпьютере типа «Arduino».

Заявляемые результаты биогазовой установки заключаются в создании в биореакторе безопасного и низко энергозатратного режима работы, который достигается с помощью применения искусственного интеллекта, новых подходов мониторинга технических узлов установки, что позволяет поддерживать работу физико-химических параметров среды в разных секциях биореактора и их частях, биомассы в которых находится на разных стадиях брожения.

1.3 Решение проблем выявления предаварийного состояния

Для целей данной работы обработка событий связана с обнаружением аномалий в данных датчиков технического устройства. Алгоритмы обнаружения аномалий давно изучаются в статистике и информатике. Обзоры общих и обычных алгоритмов обнаружения событий и обзоры алгоритмов машинного обучения для обнаружения аномалий представлены в [21-23]. В последние годы для решения проблемы использовались технологии, где анализировались все исторические данные и применялось глубокое обучение. Обычно используемые методы глубокого обучения включают многослойные персептроны [24], автоэнкодеры (AE) [25], и детальное описание данного подхода представлено в работе [26], сверточные нейронные сети (CNN) [27] и рекуррентные нейронные сети (RNN) [28]. Эти методы, как правило, могут хорошо работать для большинства задач классификации, однако они хорошо работают только с

сигналами, которые генерируются при тех же условиях работы, что и их тренировочные данные. Кроме того, методы глубокого обучения всегда терпят неудачу при недостаточном количестве данных. Учитывая, что данные, собранные в лабораторных условиях, все еще слишком просты, чтобы представлять сложную промышленную среду. Поэтому исследователи стремятся улучшить производительность методов DL при выводе сигналов при различных условиях работы, таких как разные скорости и нагрузки.

Обычно временные ряды обрабатываются рекуррентными нейросетевыми архитектурами вида encoder – decoder [29], которые по обозреваемой части временного ряда считают скрытое представление (кодировка), а на закрытой части предсказывают значения на основе полученного скрытого состояния (декодировка).

Для предсказания временных рядов обычно используют ARIMA, RNN, LSTM. Проблема алгоритма ARIMA состоит в невозможности уловить сложные паттерны во времени. Обучение RNN долго из-за затухающих и взрывающихся градиентов. LSMT решат эту проблему, но у этой архитектуры малый effective content size. Согласно работе [30] авторы рассмотрели LSTM, данная архитектура нейронной сети смотрит только на последние 200 токенов, но для задачи требуется рассмотреть больше.

Решить данную проблему можно также с помощью трансформера [31]. Трансформер смотрит на всю историю временного ряда, тем самым улавливая паттерны, удаленные во времени. Однако у трансформера есть две проблемы: dot-product self attention нечувствителен к локальному контексту; и ограничения по памяти: затраты пропорциональны квадрату длины последовательности.

Авторы предыдущих работ предложили две идеи для решения выше описанных проблем: causal convolutional self attention и LogSparse Transformer. Первая идея состоит в замене Queries и Keys на свертки, как используется в Convolution Neural Network. Расчет выполняется не для одной точки, а для нескольких точек по выбранному окну. Авторы утверждают, что такой подход может эффективно выявлять не сами значения, а тренд кривой. Вторая идея - LogSparse Transformer, данный подход состоит в использовании только части данных для расчета. Это позволит снизить потребление памяти до $O(L(logL)^2)$. При одинаковом количестве времени это позволит использовать больше исторических данных для обучения модели. Авторы выполнили сравнительный анализ со следующими алгоритмами: ARIMA, matrix factorization method - TRMF, autoregressive DeepAR (3-layers LSTM) и RNNbased state space model DeepState. Валидация была выполнена для двух датасетов: electricity-c, traffic-c; предсказывание выполнено на 7 дней вперед rolling окном и на 7 дней вперед в будущее. Предложенный подход показал лучшую точность по выбранным метрикам, но авторы в исследовании не предоставили сравнение с обычным трансформером для экстраполяции временного ряда, также не указали параметры трансформера результатов.

1.4 Моделирование аварийности технического устройства в разрезе плоского геоколлектора

Солнечная система состоит из нескольких частей, для упрощения процесса описания и моделирования мы описываем каждую часть системы в отдельности. На рисунке 1.4 показана солнечная система нагрева горячей воды с теплообменником в накопительном баке. Змеевик теплообменника установлен внутри накопительного бака и позволяет нагревать воду для потребления: для отопления и горячего водоснабжения [32], и также детальные результаты схожего подхода представлены в [33].

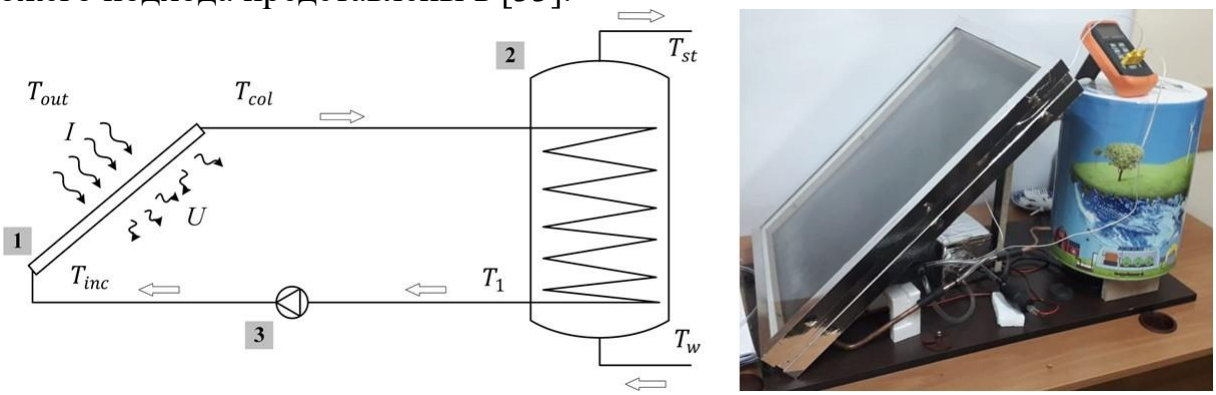

Рисунок 1.4 - Схема солнечной системы горячего водоснабжения и построенная солнечная система горячего водоснабжения. 1 – солнечный плоский коллектор; 2 – накопительный бак с теплообменником; 3 – насос.

В центральном объекте исследования есть солнечный плоский коллектор, для объекта описали математическую модель солнечного плоского коллектора, представленную на рисунке 1.5 и провели анализ энергии и эксергии. Математическая модель коллектора определяется как функция: $T_{col} = f(T_{inc},v_p,I,T_{out},U)$, где T_{col} – температура жидкости на выходе из коллектора, T_{inc} - температура жидкости на входе в коллектор,, I - объемный расход в цикле коллектора, T_{out} - температура окружающей среды коллектора, U – коэффициент тепловых потерь коллектора.

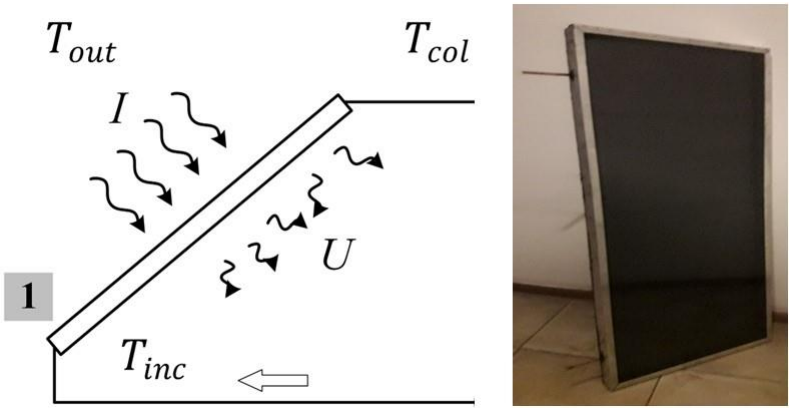

Рисунок 1.5 - Принципиальная схема солнечного плоского коллектора.

На основании солнечной энергии, поглощаемой пластиной коллектора, коэффициента тепловых потерь коллектора и тепла, поглощаемого жидкостью, уравнение энергетического баланса может быть задано как [34]:

$$\frac{T_{col}(t)}{dt} = \frac{A\eta}{C} I(t) - \frac{UA}{C}(T_{avg}(t) - T_{out}(t)) + \frac{v_{col}}{V_{col}}(T_1(t) - T_{col}(t)), \quad (1.1)$$

где $C = \rho_{col} c_{col} V_{col}$ - уравнение для расчета общей теплоемкости жидкости, $T_{avg}(t) = \frac{T_1(t)+T_{col}(t)}{2}$ - средняя температура жидкости в коллекторе.

Входные значения солнечного плоского коллектора описаны в таблице 1. Мы использовали эти параметры коллектора, чтобы сделать анализ коэффициента тепловых потерь.

Таблица 1 - Входные значения солнечного плоского коллектора.

Параметры	Значения	Ед. измерения
Расстояние от плиты до крышки	30	мм
Пластина излучения	0.95	
Коэффициент теплопередачи ветра	10	$W/m^{2\circ}C$
Наклон коллекторов	45	угол наклона
Стеклянная эмиссия	0.885	
Площадь	2	$m2$

Чтобы упростить расчет дифференциальных уравнений для расчета потерь тепла, было предложено использовать общие коэффициенты потерь для солнечного коллектора, аналогичные расчеты были выполнены в [35]. Рассмотрим солнечный плоский коллектор для системы с одним стеклянным покрытием. Расчет коэффициента потерь для верхней поверхности осуществляется по формуле:

$$U_t = \left(\frac{1}{h_{c,p-c} + h_{r,p-c}} + \frac{1}{h_w + h_{r,c-a}} \right)^{-1}, \quad (1.2)$$

Коэффициент конвекции между поглощенным и стеклянным покрытием $h_{c,p-c}$ рассчитывается с использованием параметров Нуссельта, Рэлея и Прандтля. Скорость теплообмена между двумя пластинами, наклоненными под определенным углом к горизонту, имеет очевидное значение при работе плоских коллекторов. Данные конвективного теплообмена обычно коррелируют в

терминах двух или трех безразмерных параметров: числа Нуссельта Nu, числа Рэлея Ra и числа Прандтля Pr.

$$Nu = \frac{hL}{k}, \tag{1.3}$$

Зависимость параметров Nu и Ra рассчитывалась в [36].

$$Nu = 1+1.44\left[1-\frac{1708(sin1.8\beta)^1.6}{Racos\beta}\right]\left[1-\frac{1708}{Racos\beta}\right]^{+}+\left[\left(\frac{Racos\beta}{5830}\right)^{1/3}-1\right]^{+} \tag{1.4}$$

,

$$Ra = \frac{g\beta'\Delta TL^3}{v\alpha}, \tag{1.5}$$

Зависимость разности температур между принимающей поверхностью и Ra, рассчитанная по формуле (1.5). Из этой формулы можно выстроить взаимосвязь между числом Ra и разностью температур между принимающей поверхностью и поглощенной, представлено на рисунке 1.6. Из этого графика можно отметить, что при большой разнице между поверхностями число Ra является минимальным, что влияет на вычисление числа Nu.

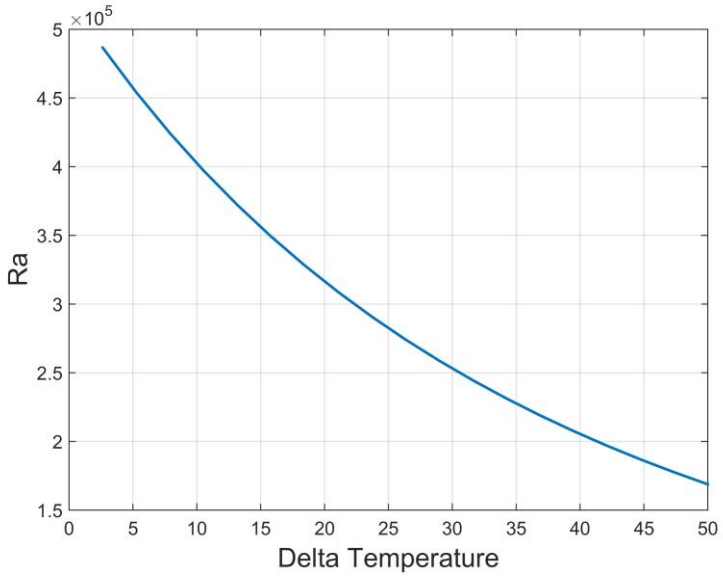

Рисунок 1.6 - Зависимость перепада температур от Ra.

Соотношение между Ra и Nu рассчитывается по формуле (1.4). На рисунке 1.7 и 1.8 отображается в логарифмическом формате. Значения Ra генерировались в диапазоне $5.96e + 03 - 2.44e + 05$ и для 5 различных углов наклона коллектора.

На рисунке 1.9 показаны теплопроводность, Nu и коэффициент теплопередачи между поглощенным и стеклянным покрытием. Максимальные значения Nu и теплопроводности дали максимальное значение коэффициента теплопередачи, это необходимо учитывать при проектировании солнечного

плоского коллектора. Наша главная цель - собрать максимум солнечного излучения и нагреть воду как можно быстрее.

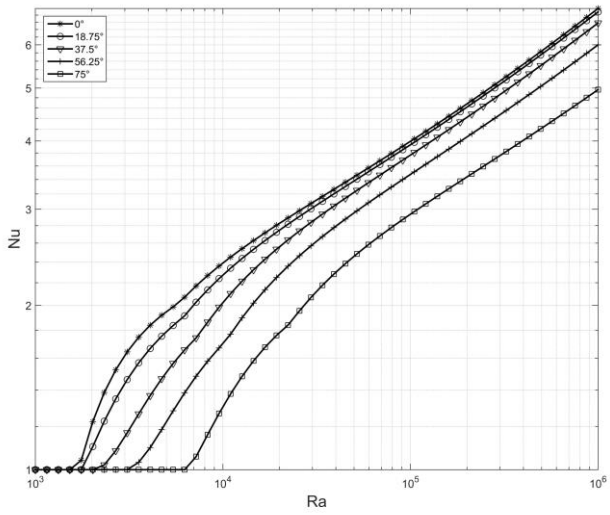

Рисунок 1.7 - Соотношение Ra и Nu в разных масштабах.

После расчета коэффициента теплопроводности необходимо рассчитать коэффициент излучения от стеклянного покрытия до поглотителя.

$$h_{r,p-c} = \frac{\sigma(T_p^2 + T_c^2)(T_p - T_c)}{\frac{1}{\epsilon_p} + \frac{1}{\epsilon_c} - 1}, \tag{1.6}$$

Расчет коэффициента излучения между поверхностью стекла и воздухом рассчитывается по формуле:

$$h_{r,c-a} = \epsilon_c \sigma(T_c^2 + T_a^2)(T_c + T_a), \tag{1.7}$$

Расчет температуры поверхности стекла рассчитывается по формуле:

$$T_c = T_p - \frac{U_t(T_p - T_a)}{h_{c,p-c} + h_{r,p-c}}, \tag{1.8}$$

Процесс расчета коэффициента потерь для верхней поверхности является итеративным и зависит от температуры поверхности стекла, на рисунке 1.10 приведена блок-схема этого расчета.

Во время расчета мы получаем следующий результат для максимального коэффициента потерь в таблице 2.

Результатом расчета является максимальный коэффициент потерь от пластины коллектора в окружающую среду, W/m^2C, равный 7.528.

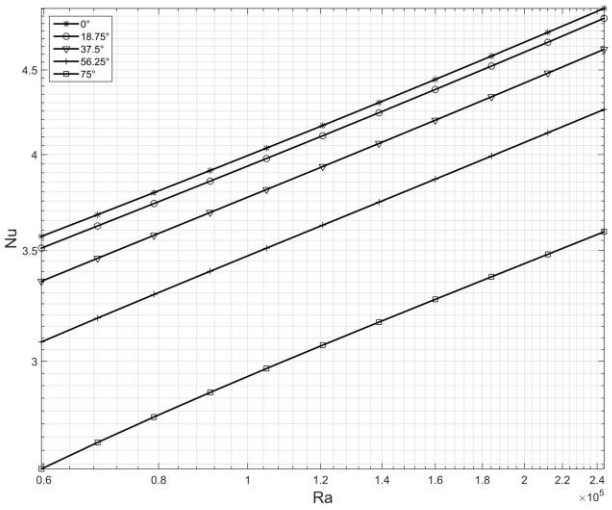

Рисунок 1.8 - Увеличенный масштаб соотношения Ra и Nu в разных масштабах.

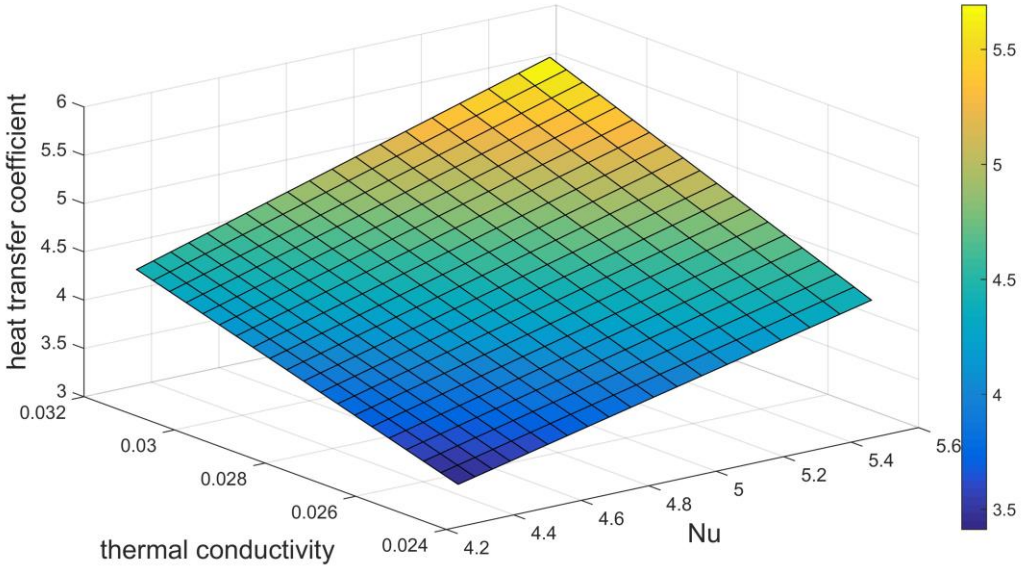

Рисунок 1.9 - Зависимость теплопроводности Nu от коэффициента теплопередачи.

На основании уравнения (1.2) получен диапазон коэффициента потерь для верхней поверхности (одной поверхности стекла), рассчитанный для разных показателей температуры окружающей среды [40; 20; -10], температура пластин имеет диапазон от 0 до 200 °C с коэффициентом теплопередачи ветра 10 W/m^2C.

На рисунке 1.11 мы получили зависимость между средней температурой пластины и коэффициентом потерь сверху. Можно заметить, что температура и коэффициент потерь увеличивались почти линейно, кроме от 0 до 60 °C. Наш рассчитанный коэффициент равен 7.5286, на рисунке это значение эквивалентно средней температуре плиты около 120 °C. Этот факт может быть возможен в летнее время.

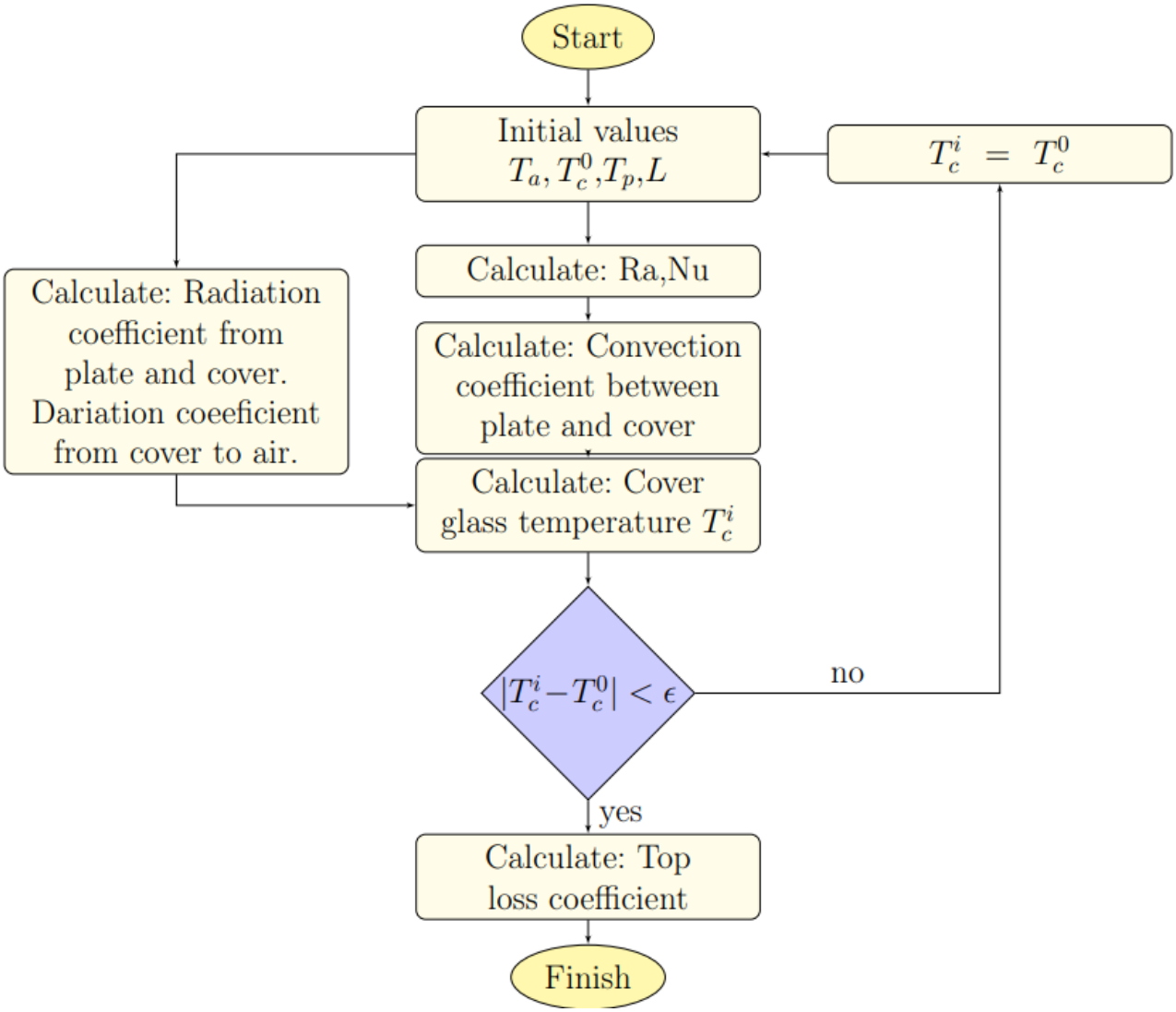

Рисунок 1.10 - Блок-схема расчета максимального коэффициента потерь.

Расчет коэффициента потерь для нижней поверхности коллектора рассчитывается по формуле:

$$U_b = \frac{k}{L}, \tag{1.9}$$

,где k - теплопроводность изоляционного материала (материал - 0.04 Вт/(мК). L - толщина изоляционного материала.

Коэффициент тепловых потерь для боковых границ обычно имеет небольшие значения и часто не рассчитывается для общего коэффициента тепловых потерь. Если необходимо рассчитать точные значения тепловых потерь, расчет будет выполнен по формуле [37]:

$$U_e = \frac{(UA)_{edge}}{A}, \tag{1.10}$$

,где U рассчитывается по формуле (1.11), но для боковой изоляции. Толщина коллектора: 0.1 м; толщина изоляции: 0.01 м. Общий коэффициент потерь будет рассчитываться путем суммирования всех коэффициентов:

$$U = U_t + U_b + U_e, \tag{1.11}$$

Таблица 2. - Результат расчета максимального коэффициента потерь.

Входные данные	Значения	Вычисленные данные	Значения
Температура окружающей среды,°C	40	Ra	5.28*e*+05
Температура поверхности, °C	50	Nu	5.4279
Температура абсорбента, °C	100	Коэффициент конвекции между плитой и крышкой, $W/m^2{\circ}C$	4.8829
Расстояние между покрытием и абсорбентом, m	0.03	Коэффициент излучения от плиты до крышки, $W/m^2{\circ}C$	8.6584
		Коэффициент излучения для покрытия воздуха, $W/m^2{\circ}C$	6.9555
		Температура покровного стекла, °C	66.6

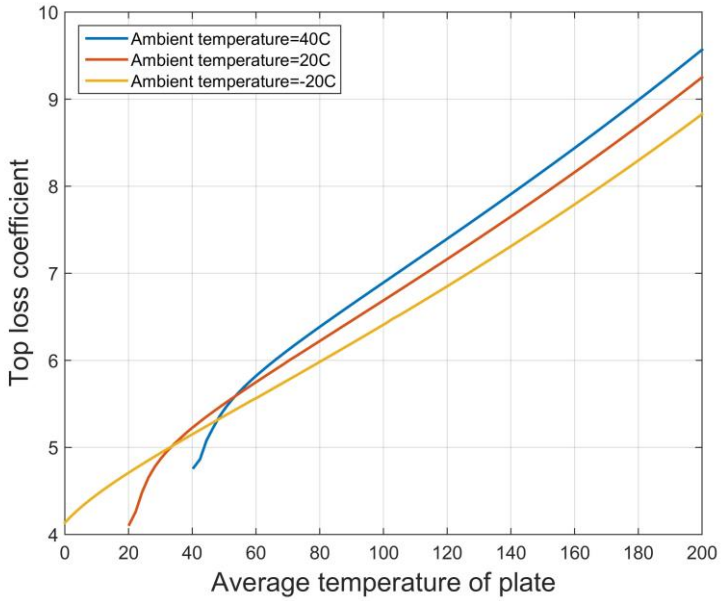

Рисунок 1.11 - Зависимость между средней температурой плиты и коэффициентом потерь сверху.

MatLab / Simulink провела симуляцию модели, а использованная модель была взята из исследования [38]. Созданная модель солнечной системы горячего водоснабжения содержит следующие узлы: плоский коллектор, накопительный бак и насосы. С помощью математической модели солнечного плоского коллектора (1.1) была реализована блок-схема в Simulink. Блок схема установки показана на рисунке 1.10. Результатом моделирования процесса работы установки является температура жидкости на выходе из коллектора [39].

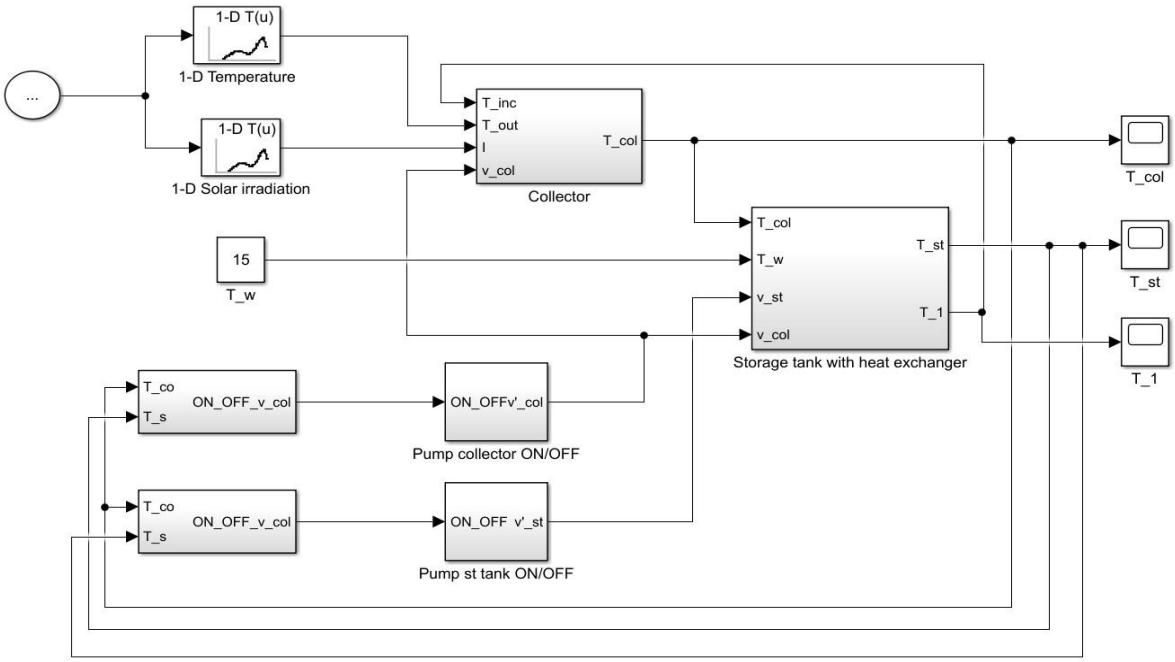

Рисунок 1.12 - Общая блок-схема солнечной системы горячего водоснабжения.

Мы отобрали данные за один год по Алматинской области. Входные данные: солнечное излучение и температура окружающей среды. Результат моделирования показан на рисунке 1.13. На основании полученных результатов можно отметить, что для региона система солнечного нагрева горячей воды не будет обеспечивать приемлемую температуру в плоском коллекторе и резервуаре-хранилище, период с декабря по март. Лучший результат (высокая температура) системы будет обеспечен с июня до середины октября.

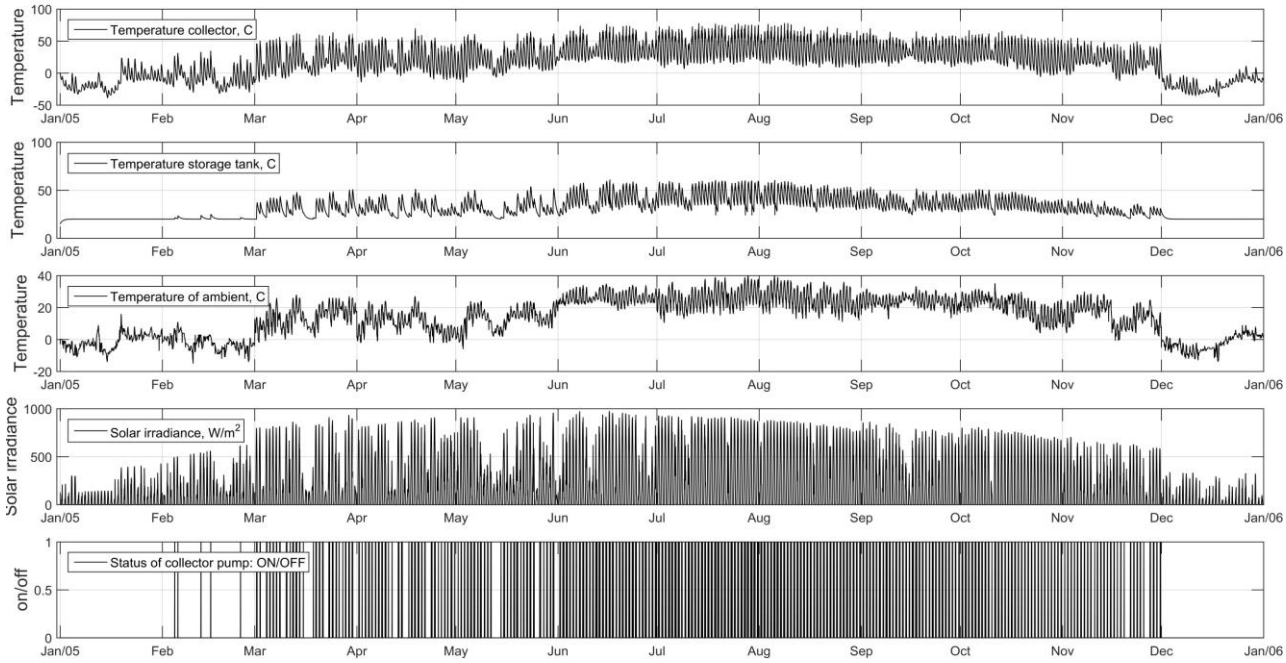

Рисунок 1.13 - Результат моделирования за один год.

Графики на рисунке 1.13 показывают высокую корреляцию между солнечным излучением и температурой в коллекторе и резервуаре. Если плотность солнечного излучения выше, то температура является более монотонной, а стандартное отклонение температуры в хранилище близко к среднему значению, например, с июля по август.

Рабочий процесс насоса в цикле коллектора зависит от достигнутой температуры в цикле коллектора, мы использовали такие же предельные значения для активации насоса, как в исследовании. На рисунке 1.14 и 1.15 показано, что в летнее время насос перешел на регулярную работу для передачи достигнутой температуры из коллектора в резервуар-хранилище. В зимнее время насос явно не работает.

Для подробного анализа результатов моделирования мы смоделировали данные моделирования до 3 дней (1 июля - 3 июля). Результат моделирования, отображенный на рисунках 1.14 и 1.15, показал максимальную температуру воды в резервуаре-хранилище 56,9 °C в Алматинской области и в коллекторе - 60.5 °C. На рисунках отображена информация о температуре в коллекторе, резервуаре и режиме работы насоса. Можно заметить явную корреляцию между температурой окружающей среды и температурой нагреваемой воды. Если температура в коллекторе достигает 40 °C, в этом случае насос активизируется и приступает передавать нагретую жидкость в теплообменник внутри накопительного резервуара, а взамен получает охлажденную жидкость, которая подается в цикл коллектора для нагрева. Также на графике можно заметить снижение температуры в цикле, одной из причин является изменения погодных условий (снижение температуры окружающей среды, снижение солнечной активности).

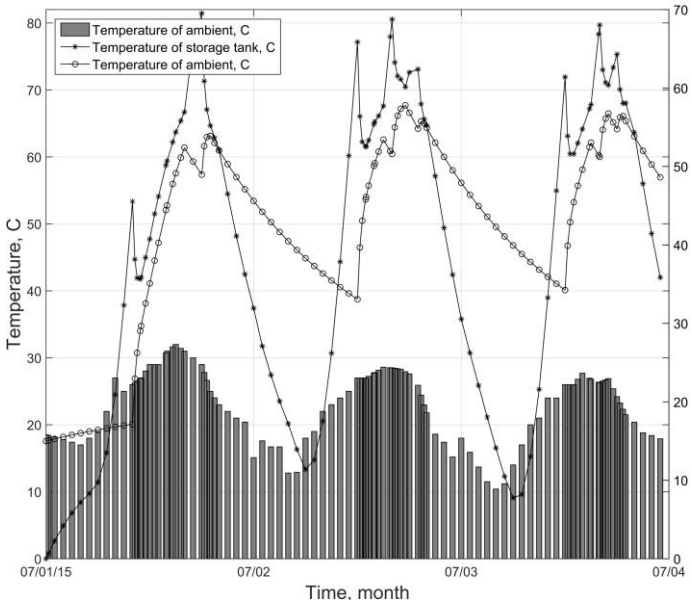

Рисунок 1.14 - Температура окружающей среды и температура в коллекторе и резервуаре.

Перед выполнением моделирования нагрева воды необходимо проанализировать технические характеристики солнечного плоского

коллектора, такие как тепловые потери коллектора. Этот анализ занимает больше времени, но позволяет имитировать более точные результаты. После расчета значений можно приступить к анализу энергии. Основной целью имитационной модели является определение предельных значений установки и выявление слабых точек установки для улучшения. Разработанная имитационная модель позволяет моделировать работу солнечной системы горячего водоснабжения в различных климатических условиях (температура и солнечная радиация) и различных индивидуальных параметров системы для повышения эффективности системы в различных климатических условиях Казахстана.

Результат моделирования показал, что в дневное время температура воды в коллекторе составляет 70 °C, и в течение этого периода ее можно использовать дома. Основная проблема - хранение нагретой воды, так как ночью нагретая вода охлаждается. Другая задача, которую мы можем определить из анализа моделирования, состоит в том, чтобы найти способ продления периода работы системы с высокой эффективностью (апрель-октябрь), возможно, это будет инженерное, техническое решение или программирование контроллера. Этот момент необходимо исследовать и улучшить.

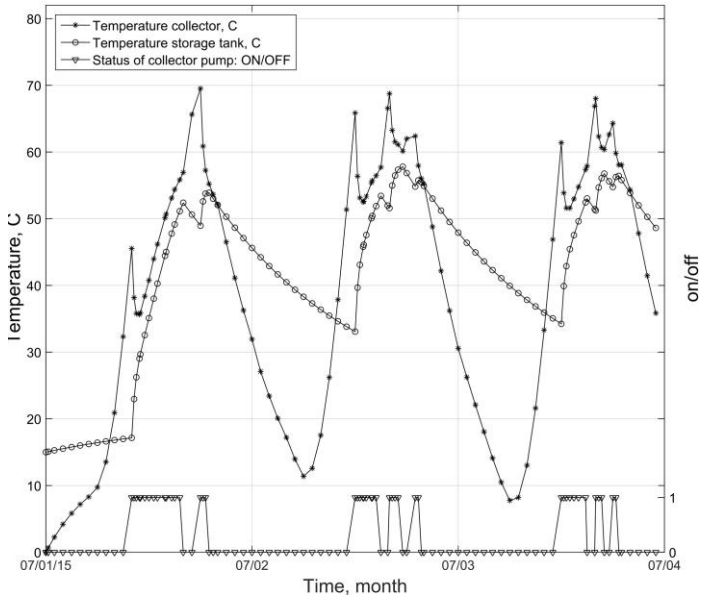

Рисунок 1.15 - Температура в коллекторе и резервуаре с режимом работы насоса.

1.5 Цифровизация биогазовой установки

В современное время сбор, хранение и анализ данных имеет стратегическое преимущество, которое позволит повышать КПД установки от управления установкой до прогнозирования эффективности технического устройства. Индустриальные системы имеют собственные контроллеры для сбора и управления установкой, но данные решения являются закрытыми и практически не имеют возможности масштабирования для других типов технических устройств.

Простые контролеры в основном имеют ограниченный функционал, также имеются функции для сбора данных из базы, коллектора, датчиквна крыше. На данный момент популярны модульные контроллеры, которые можно запрограммировать и интегрировать в единую систему Arduino, Rasberry Pi с возможностью сбора и хранения данных в облаках, также с возможностью управления установкой через Интернет [40-42]. Преимущество таких контроллеров в том, что они могут быть модернизированы и масштабированы в зависимости от поставленной задачи.

Общей особенностью контроллеров для биогазовой установки является предоставление информации о работе поддерживаемой системы. Базовые контроллеры обеспечивают показания только 3–4 датчиков температуры, pH, давления и других показаний в баках. Более сложные устройства позволяют использовать до 5–7 датчиков температуры [43]. Ценная информация об эффективности биогазовой установки может быть обеспечена использованием датчиков расчета выхода возможной вырабатываемой электроэнергии.

Для управления, сигнализации об аварийных режимах и мониторинга применяется контролер на основе Arduino Mega с набором датчиков и внешних устройств: монохромный дисплей, карта памяти, модуль связи по Ethernet, блок силовых реле, часы реального времени, 3 датчика температуры, звуковой сигнал [44]. Схема работает от БП на 5 В. [45]. Контролер подключен к беспроводной точке доступа, обеспечивающей связь по WIFI и Ethernet с любым браузером на устройствах пользователя. В Arduino реализован простой WEB сервер для удаленного мониторинга.

Большое количество измерительных каналов позволяет добавить дополнительные датчики разных параметров на входе и выходе установки, а также расходомеров. Использование этих датчиков позволяет реализовать тепло счетчик, который можно использовать при анализе эффективности работы технического устройства. Доступ к Интернету дает возможность для мониторинга многих современных систем через Интернет – технология умный дом и IoT. Прототип контроллера представляет собой простое программное обеспечение веб-сервера, которое позволяет контролировать работу установки на компьютере (или мобильных устройствах, таких как смартфоны) в домашней сети. В целях безопасности эти данные должны быть защищены. Уведомление о событиях через SMS или уведомление через чаты (WhatsApp, Telegram) в прототипе системы не было реализовано, хотя это полезная функция. Однако возможно дальнейшее расширение системы с использованием дополнительного модуля связи GSM. Для обслуживания относительно небольших солнечных установок это представляется невыгодным, поскольку требует дополнительных сборов за передачу данных. Уведомление по электронной почте с использованием встроенного модуля Ethernet представляется лучшим решением для такой системы.

Результаты измерений обрабатываются, используются для индикации и сохраняются. Также, по этим измерениям формируются управляющие воздействия на агрегаты установки и сигнализацию аварийного режима. На

рисунке 1.16 и 1.17 показана схема контролера со снятой крышкой. Видны основные компоненты схемы, подводящие провода и блок питания [46].

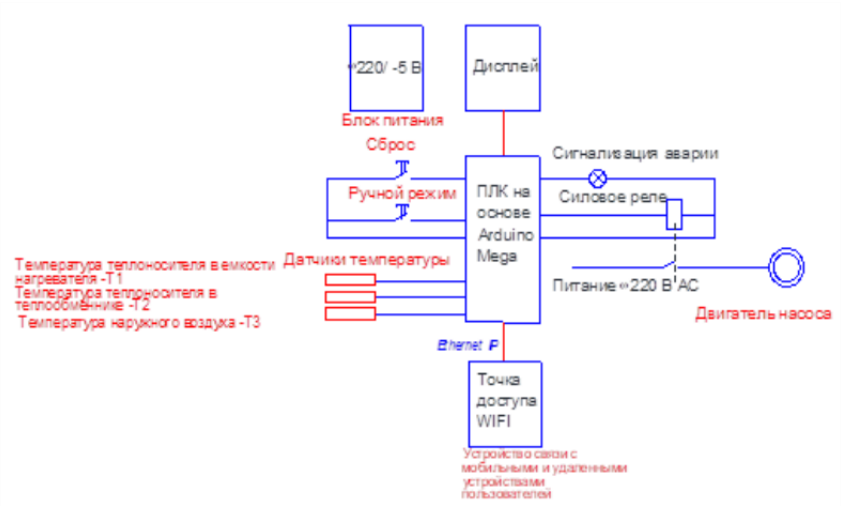

Рисунок 1.16 - Схема внешних соединений контролера.

На рисунках рис. 1.18–1.20 отображен общий вид биогазовой установки с подключенными контроллерами, также общий вид контролера и в рабочем режиме. Как можно заметить, с помощью контроллера можно выполнять мониторинг работы отдельного реактора на мониторах, также данные поступают на сервер.

Часы реального времени позволяют прикреплять временные метки к значениям и событиям. Это важно, т.к. позволяет применять эти значения в большинстве современных SCADA систем и им подобных. Есть возможность строить графики и тренды. В настоящее время идет накопление реальных данных. Из практики наиболее целесообразно использовать данные, поступающие каждые 5-10 мин. Кроме сбора данных контроллер имеет дополнительные функции управления и защиты. Следующие функции: управление циркуляционным насосами, защита от замерзания или периодический перегрев и повышения давления в реакторах.

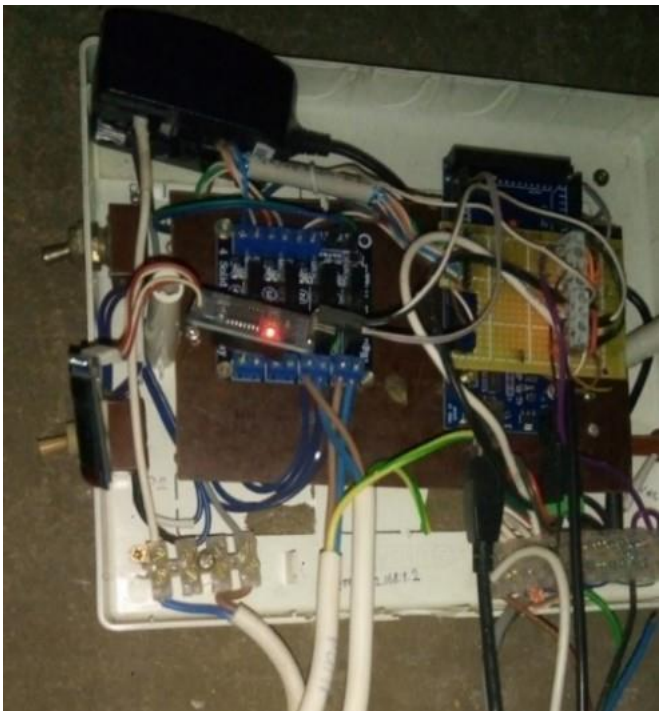

Рисунок 1.17 - Схема внешних соединений контролера.

Рисунок 1.18 - Общий вид биогазовой установки.

1.6 Выводы по разделу, постановка задачи

В первом разделе были проведены следующие исследования, направленные на проблему сбора и моделирования данных из системы геоколлетора, которая является частью системы биогазовой установки.

Проведен научный обзор систем биогазовых устройств и научный обзор в проблемах возникновения и предсказания аварий. Также выполнен анализ существующих решений по методам предсказания аварийности технических

устройств. На основе анализа литературных источников, проведённого в данной главе, были поставлены следующие задачи:

1. Разработать комплексную платформу по повышению эффективности работы биогазовой установки на основе нейронных дифференциальных уравнений для повышения отказоустойчивости оборудования.
2. Создать автоматизированный комплекс по сбору и обработке данных с биогазовой установки.
3. Определить критерии «оптимального» и предаварийного режима работы и обслуживания объектов, минимизации роли «человеческого фактора».
4. Разработать систему мониторинга и «быстрого» оповещения о внештатных (аварийных) ситуациях в работе объекта.

Рисунок 1.19 - Общий вид контролера.

Рисунок 1.20 - Работа контролера.

Для апробации разрабатываемой модели, в период создания объекта исследования, были выполнены работы по моделированию работы одного из узлов биогазовой установки - плоского геоколлектора. Для симуляции данных использовался MatLab, это позволило проводить эксперименты с данными без ожидания получения реальных данных.

Выполнены работы по оцифровке биогазовой установки с помощью контролера Arduino Mega. Данный вид контролера показал свои преимущества для сбора, хранения и передачи оперативных данных о работе установки. На основе данного контролера можно произвести симуляции поведения установки в различных критических условиях и выполнять управление установкой для недопущения аварийных ситуаций.

2 МОДЕЛИРОВАНИЕ ДЛЯ ПРОГНОЗИРОВАНИЯ ВРЕМЕННЫХ РЯДОВ

Временной ряд определяется как набор наблюдений $x_1,x_2,x_3,...,x_N$, расположенных в порядке их появления и соответственно записи в базу данных. Временной ряд называется одномерным, если значения $\{x_i\}_{i=1}^{N}$ являются скалярами или многомерными, если $\{x_i\}_{i=1}^{N}$ являются векторами. Наблюдения $\{x_i\}_i$ могут быть вещественными числами или категориальными значениями. Наблюдения записываются в моменты времени $t_1, t_2,...,t_N$. Моменты времени t_i не обязательно регулярные данные (регулярные интервалы данных), другими словами, промежутки $\{t_i - t_{i+1}\}$ не обязательно должны быть равными для разных i. Будет использовано обозначение множества наблюдений и их соответствующие моменты времени как $\{x_i, t_i\}_{i=1}^{N}$, где N - количество записанных наблюдений.

Цель моделирования временных рядов состоит в том, чтобы предсказать порождающий процесс и описать внутреннюю структуру временного ряда и построить совместное распределение данных $p(x_1,...,x_N)$. В большинстве приложений временные ряды состоят из измерений некоторой непрерывной функции, т.е. реальный мир (процесс). Например, если мы измеряем температуру воздуха, она имеет конкретное значение в любой точке на реальной временной шкале. Другими словами, время является непрерывной скалярной переменной. Этот тип данных называется «непрерывный» временной ряд. Однако из-за ограниченных возможностей наших измерительных инструментов или ограничения хранения данных мы можем быть вынуждены проводить измерения с определенной степенью детализации, например, каждую секунду, минуту и т. д. Полученный набор измерений является аппроксимацией, лежащей в основе непрерывной функции, но он измеряется только в дискретный набор моментов времени. Такие временные ряды называются дискретными, так как переменная времени теперь является дискретной. Если временной ряд измеряется через равные промежутки времени, мы будем называть его регулярный. Если временные промежутки между точками различаются, временной ряд является нерегулярным.

Не все временные ряды могут иметь в основе непрерывную функцию, или в некоторых случаях непонятно, что это за функция. Такие временные ряды обычно описывают серию событий: прибытие клиентов в магазин, историю онлайн-покупок, историю землетрясений и называются событиями. Этот тип временных рядов является дискретным по своей природе. В этой диссертации мы будем моделировать предсказания аварийности технического устройства. Таким образом, в этой работе будут рассмотрены методы для двух основных типов временных рядов: дискретного времени и непрерывного времени. Данные дискретного времени представляют собой конечный набор переменных, измеренных в различные моменты времени, в то время как данные непрерывного времени имеют потенциально бесконечный набор точек измерения на непрерывной временной шкале. Эти два типа временных рядов могут быть

преобразованы друг в друга, например, путем дискретизации непрерывных значений, путем размещения дискретных точек на непрерывной временной шкале. В данной диссертации будет изучено данные в виде непрерывного времени для нерегулярных данных.

Целью большинства подходов анализа временных рядов является моделирование совместного распределения $p(x_1,...x_N)$, где $x_1,...x_N$ случайные величины, описывающие наблюдения во временном ряду. Используя это распределение, можно сформулировать три основные проблемы в анализе временных рядов:

1. Фильтрация: $p(x_i|x_1...x_i)$ – вывести апостериорное распределение по текущему наблюдению x_i с учетом текущих и прошлых данных.

2. Сглаживание: $p(x_i|x_1...x_N)$ – использовать наблюдения как из прошлого, так и из будущего, чтобы сделать вывод. Этот подход можно использовать только в автономном режиме, когда мы знаем всю последовательность $x_1...x_N$. Сюда можно отнести обычную задачу заполнения пропущенных значений.

3. Прогнозирование: $p(x_{i+\sigma}|x_1...x_i)$, где $\sigma > 0$ – предсказывать будущее, используя историю прошлого.

4. Обнаружение аномалий: $p(x_i|x_1...x_N)$ – вывести апостериорное распределение по аномальному наблюдению x_i, которое имеет значительное отклонение.

Естественно, есть и другие задачи, не укладывающиеся в эту постановку, например, классификация временных рядов, причинно-следственный вывод.

Наша задача состоит в том, чтобы предсказать дальнейшие события в измеренных данных временного ряда. Временной ряд представляет собой список векторов по измерениям d и наблюдениям T , $X =< x_1,...,x_d >$, где $x_k = (x_{1,k},x_{2,k},...,x_{T,k})$. Данные каждого временного ряда за период T могут быть представлены в виде парного сегмента $(X_{1:T},Y_{1:T}) = (X_1,Y_1),(X_2,Y_2),...,(X_T,Y_T)$, где X_t - измеренное значение, а Y_t - наблюдаемое событие (неисправность) [46]. $Y_t \in 0,1$ из-за событий прогнозирования: True или False. В данном исследовании мы прогнозируем будущее событие Y_{T+1} на основе данных временного ряда за $(X_{1:T},Y_{1:T})$ и оцениваем их отношения.

2.1 Прогнозирование временных рядов с помощью ARIMA

Одной из самых популярных моделей временных рядов является авторегрессионное интегрированное скользящее среднее (ARIMA) [47]. ARIMA относится к классу линейных моделей, где значение является линейной функцией прошлых наблюдений [48]. Линейные модели привлекательны своей простотой в интерпретации и реализации, но по своей конструкции не могут обрабатывать нелинейные зависимости данных.

Модель ARIMA представляет собой комбинацию двух других широко используемых подходов: Авторегрессионный (AR) и скользящее среднее (MA). Авторегрессивная часть означает, что будущие значения представляют собой комбинацию p прошлых наблюдений, в данном случае линейная функция со случайной ошибкой:

$$\hat{x}_i = c + \sum_{j=1}^{p} \phi_j x_{i-j} + \epsilon_i \tag{2.1}$$

где c есть константа и $\varphi_j, j = 1...p$ есть параметры модели. Целое число p есть порядок модели. Часть скользящего среднего определяется с помощью ошибок прошлых наблюдений и вычисляется с помощью:

$$\hat{x}_i = \mu + \sum_{j=1}^{q} \theta_k \epsilon_{i-k} + \epsilon_i \tag{2.2}$$

где μ – среднее значение временного ряда, $\theta_k, k = 1...q$ – параметры модели. Случайные ошибки $_i$ предполагаются независимыми и одинаково распределенными и $\epsilon_i \sim \mathcal{N}(0,\sigma^2)$ для некоторой постоянной дисперсии σ. Комбинируя AR и MA подходы, получаем модель ARMA:

$$\hat{x}_i = c + \sum_{j=1}^{p} \phi_j x_{i-j} + \sum_{j=1}^{q} \theta_k \epsilon_{i-k} + \epsilon_i \tag{2.3}$$

Однако эту модель можно применять только к стационарным временным рядам, где совместное распределение наблюдения остается неизменным на протяжении всей временной шкалы. Модель ARIMA обобщает эту модель на нестационарный случай.

На практике эта модель оказывается достаточно гибкой для представления различных временных рядов. Однако по-прежнему ограничена линейной формой, которая может быть неадекватной для некоторых приложений. ARIMA предназначена для моделирования одномерных временных рядов с непрерывными значениями и основана на том факте, что данные не имеют пропущенных значений.

2.2 Модели, основанные на пространственном состоянии

Модели в пространстве состояний - это вероятностные модели, которые фиксируют скрытую структуру данных, вводя скрытую переменную для каждого наблюдения, и моделируют динамику между скрытыми переменными [49]. Это мощный и широко используемый класс моделей, позволяющий выполнять интерполяцию и экстраполяцию (прогнозирование), учитывающий тенденции, сезонные эффекты и заполняющий недостающие значения [50]. Начнем с факторизации совместного распределения следующим образом:

$$p(x_1,...,x_N) = p(x_1)p(x_2|x_1)p(x_3|x_2,x_1)...p(x_N|x_1,x_2,...,x_{N-1}) \tag{2.4}$$

Здесь каждое наблюдение зависит от всех предыдущих наблюдений. Асимптотической сложностью вычисления является $O(N^2)$, это связано с тем, что каждый член зависит не более чем от N предыдущих наблюдений. Чтобы уменьшить вычислительные затраты, можно ввести марковское предположение, утверждающее, что текущая точка данных x_i зависит только от предыдущей

точки. Другими словами, при текущем наблюдении x_i будущи наблюдения $x_{i+1}...x_N$ не зависят от прошлых наблюдений $x_1...{}_{i-1}$. Получаем марковскую модель первого порядка модель [51]:

$$p(x_1,...,x_N) = p(x_1)p(x_2|x_1)p(x_3|x_2)...p(x_i|x_{i-1}) \quad (2.5)$$

Однако марковское свойство является сильным предположением, которое на практике не выполняется [52]. Обозначим скрытую переменную как z_i, которая является внутренним или «скрытым» состоянием динамической системы. Скрытое состояние z_i обобщает историю прошлых наблюдений и используется для предсказаний в будущее. Каждому наблюдению x_i соответствует z_i. Чтобы дать представление о том, что скрытая переменная может захватить, рассмотреть серию изображений. Скрытой переменной может быть форма объекта, его поза и освещение, а наблюдение – это изображение, показывающее этот объект. Обратите внимание, что наблюдения больше не будут следовать марковскому свойству, длинно масштабная зависимость моделируется через скрытую переменную, наблюдения становятся условно независимыми друг от друга с учетом скрытой переменной. Кроме наблюдений смоделируем еще и распределение по скрытым состояниям:

$$p(z_{1:N}, x_{1:N}) = p(z_1)p(z_1|x_1) \prod_{i=2}^{N} p(z_i|z_{i-1})p(x_i|z_i) \quad (2.6)$$

Где $p(z_i|z_{i-1})$ - оператор перехода, моделирующий эволюцию системы, $p(x_i|z_i)$ - это распределение выбросов, представляющее, как генерировать наблюдение с учетом скрытого состояния. В данном типе моделей мы не наблюдаем z_i напрямую. Вместо этого мы наблюдаем x_i, который является функцией скрытого состояния z_i. Стоит отметить, что наблюдение x_i не зависит от других скрытых состояний, заданных z_i. Скрытое состояние z_i зависит только от предыдущего состояния z_{i-1} и не зависит ни от одного из предыдущих состояний z_{i-2},z_{i-3}... Таким образом, сложность модели остается линейной по количеству наблюдений, но по-прежнему поддерживают долгосрочные зависимости во временном ряду. Модели, в которых скрытое состояние является функцией предыдущего состояния, называются авторегрессивными.

В моделях в пространстве состояний нам нужно накладывать меньше предположений на данные, поскольку динамика моделируется в скрытом пространстве. Скрытое состояние может объяснить неизвестные основные причины данных, которые больше не нужно моделировать явно. Для сравнения, в ARIMA необходимо явно указывать, что следующее наблюдение должно быть линейной функцией предыдущих наблюдений и их ошибок.

Следующим шагом, можно получить маргинальное распределение по наблюдениям путем интегрирования по скрытым состояниям:

$$p(x_{1:N}) = \int p(z_{1:N}, x_{1:N}) dz_{1:N} \quad (2.7)$$

Используя правило Байеса, вычисляем апостериорное значение по скрытым состояниям:

$$p(z_{1:N}|x_{1:N}) = \frac{p(x_{1:N}|z_{1:N})p(z_{1:N})}{p(x_{1:N})} \tag{2.8}$$

В следующем разделе будет представлено две модели: линейные гауссовские SSM и скрытые марковские модели.

2.2.1 Линейно-гауссовские модели пространства состояний

Мы начнем с модели, которая делает сильное предположение о генеративном процессе - Линейно-гауссовские или фильтра Калмана имеют сильное предположение о генеративном процессе, которое мы рассмотрим [53]. В этой модели распределения принимают форму гауссова, отношения между состояниями и наблюдением являются линейным преобразованием с добавлением шума [54]. Благодаря ограниченной форме этой модели апостериорный вывод для скрытой переменной z поддается обработке. Эту модель можно рассматривать как обобщение модели ARIMA, в которой динамика моделируется в скрытом пространстве, а не в пространстве данных. Связь между латентными состояниями и наблюдениями описывается следующим образом:

$$\begin{gathered}\text{Переходное состояние: } z_i = A_i z_{i-1} + w_i \\ \text{Выбросы (выходы): } x_i = C_i z_i + v_i\end{gathered} \tag{2.9}$$

$A_i \in \mathscr{R}^{p\times p}$ - матрица перехода, где p - размерность скрытого пространства; C_i - матрица излучения $q \times p$, где q - размерность пространства данных. Таким образом, переходы являются линейной проекцией предыдущего состояния с добавлением шума $w_i \sim \mathcal{N}(0,Q_i)$. Наблюдения также являются линейными и также имеют гауссовский шум $v_i \sim \mathcal{N}(0,R_i)$. Переменные, содержашие шум $w_i \sim \mathcal{N}(0,Q_i)$ и $v_i \sim \mathcal{N}(0,R_i)$ предполагаются не коррелированными по временным шагам.

Матрицы, определяющие модель A_i,C_i,R_i,Q_i , могут зависеть или не зависеть от времени и прошлого наблюдения. Если эти матрицы фиксированы, система становится нестационарной или стационарной. На практике параметры A_i,B_i,C_i,D_i необходимо оценивать по наблюдениям. Начальное скрытое состояние z_1 моделируется как распределение Гаусса: $z_1 \sim \mathcal{N}(\mu,\Sigma)$. Следовательно, все остальные состояния z_i также подчиняются распределению Гаусса:

$$\begin{gathered}p_{\theta_i}(z_i|z_{i-1}) = \mathcal{N}(z_i; A_i, z_{i-1}, Q_i) \\ p_{\theta_i}(z_i|z_{i-1}) = \mathcal{N}(z_i; C_i, z_i, R_i)\end{gathered} \tag{2.10}$$

где $\theta_i = A_i,C_i,R_i,Q_i$.

Совместное распределение можно рассчитать следующим образом:

$$p(z_{1:N}, x_{1:N}) = p(z_1)p(z_1|x_1) \prod_{i=2}^{T} p(z_i|z_{i-1})p(x_i|z_i) \tag{2.11}$$

Все компоненты этого распределения принимают гауссову форму, и результаты в виде распределения также является гауссовыми. Фактически все маргинальные, условные и совместные распределения имеют вид гауссова. Благодаря марковским свойствам и линейным гауссовским переходам эта модель имеет большое преимущество, заключающееся в наличии управляемой апостериорной вероятности. Апостериорное значение можно получить с помощью алгоритмов фильтрации Калмана и сглаживания Калмана [54]. Одним из недостатков фильтрации Калмана является плохая масштабируемость. Такая вычислительная сложность делает фильтр Калмана неприменимым для многомерных задач.

2.3 Recurrent Neural Networks

Описанные выше модели обеспечивают удобную структуру с представлениями скрытых состояний и могут быть готовы к использованию во многих приложениях. Однако для того, чтобы модель была управляемой, они делают серьезные предположения, такие как линейность переходов или дискретные скрытые состояния, которые могут быть слишком ограничивающими для многих приложений и недостаточно гибкими для моделирования сложной временной структуры. Ниже будут представлены модели, использующие нелинейные функции перехода. Эти модели более выразительны и накладывают меньше ограничений на динамику, но приносят в жертву легкость логического вывода.

Рекуррентная нейронная сеть (RNN) представляет собой модели со скрытыми переменными, в которых функции перехода и излучения нелинейны и параметризуются с помощью нейронных сетей. RNN разработаны и широко используются для языкового моделирования и машинного перевода [55]. RNN определяют скрытое состояние как нелинейную функцию предыдущего состояния и текущего наблюдения: z_i = *RNNUnit*(z_{i-1},x_i), где *RNNUnit* дифференцируемая нелинейная функция [56]. Начальное латентное состояние z_0 обычно устанавливается равным нулевому вектору. Вероятность наблюдения, обусловленного текущим состоянием, равна распределению Гаусса (для непрерывных данных) и имеет следующий вид: $p(x_i|z_i)$ = *mathcalN*(x_i;$\mu(z_i)$,σ^2), где обычно устанавливается равным постоянное значение, а (z_i) является нелинейной функцией латентного состояния, обычно нейронной связью с прямой связью. Если данные дискретны, мы можем использовать полиномиальное распределение: $p(x_i|z_i)$ = ($\pi(z_i)$).

$$p(x_1, \dots, x_N) = p(z_1)p(z_1|x_1) \tag{2.12}$$

Таким образом, возможности RNN полностью зависят от нелинейной функции перехода и выбор этой функции важен. Здесь мы опишем рекуррентную нейронную сеть с Gated Recurent Units [57]. Блок GRU состоит из двух входов: вход обновления u_t, определяющих количество информации, сохраненной из предыдущего латентного состояния h_t и нового наблюдения u_t, и вход сброса r_t, определение того, какую информацию следует забыть из предыдущего

состояния, на рисунке 2.1 показан пример одной ячейки GRU [58]. Одна ячейка GRU описывается следующим образом:

$$\begin{aligned} u_t &= \sigma(W_{xx}x_t + W_{hu}h_{t-1} + b_u) \\ r_t &= \sigma(W_{xr}x_t + W_{hr}h_{t-1} + b_r) \\ h'_t &= tanh(W_{xh'}x_t + W_{hh'}(r_t \odot h_{t-1}) \\ h_t &= (1 - u_t) \odot h'_t + u_t \odot h_{t-1} \end{aligned} \quad (2.13)$$

Матрицы *W* являются здесь матрицами весов. Ячейка памяти и выход блока h_t никак не разделяются, и следующий выход h_t получается как комбинация (задаваемая выходом u_t) предыдущего выхода h_{t-1} и текущего кандидата в выход h'_t, который зависит от h_{t-1}, но через выход перезагрузки r_t.

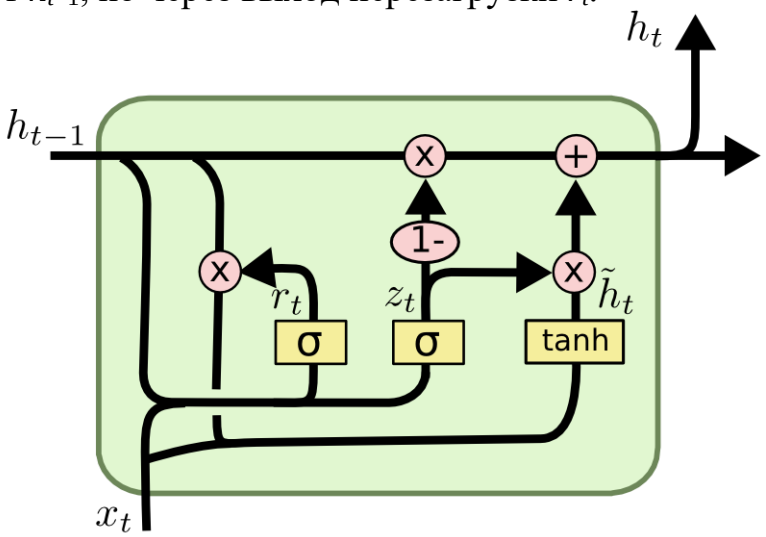

Рисунок 2.1 - GRU ячейка [59]

Ячейки GRU обеспечивают механизм обновления, чтобы сбалансировать сохранение информации из предыдущего состояния и обновление состояния с использованием нового наблюдения, что позволяет моделировать долгосрочные зависимости в данных. Можно обобщить RNN, наложив несколько слоев друг на друга, создав более сложную нелинейную структуру. Чтобы обучить модель, оптимизируются параметры модели, чтобы максимизировать логарифмическую вероятность наблюдений с использованием стохастического градиентного спуска.

Вместо нейронов в сетях LSTM есть блоки памяти, которые связаны через слои. Блок имеет компоненты, которые делают его умнее классического нейрона, и память для последних последовательностей. Блок содержит вентили, которые управляют состоянием блока и выходом. Блок работает с входной последовательностью, и каждый вентиль в блоке использует сигмовидные блоки активации для управления тем, срабатывают они или нет, делая изменения состояния и добавления информации, проходящей через блок, условными. LSTM прогнозирует несколько значений в будущем, изменяются только входные значения для периода в будущем, рисунок 2.2.

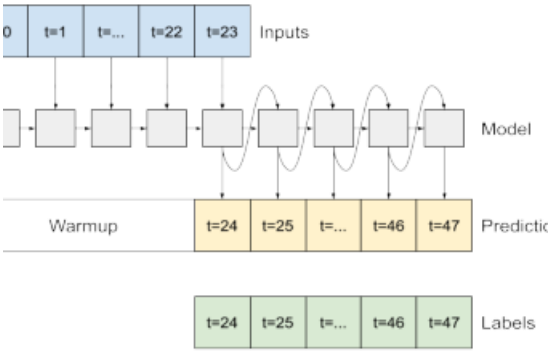

Рисунок 2.2 - Пример использования LSTM для прогнозирования значений во временном ряду.

Как и модели в пространстве состояний, RNN предназначены для предсказаний на один шаг вперед: следующее значение x_i с учетом истории наблюдений $x_1...i-1$. Можно использовать RNN для прогнозов, когда необходимо предсказать нескольких значений в будущем $x_i...x_{i+k}$. Чтобы сгенерировать значения после x_i, используются предсказанные значения в качестве входных данных для следующего шага RNN, чтобы вычислить следующее скрытое состояние. Другим полезным свойством RNN является скрытое состояние z_i, которое должно обобщать историю наблюдений до наблюдения x_i, и мы можем использовать эту сводку для последующих задач, таких как классификация временных рядов.

RNN представляют данные через последовательность дискретных состояний и хорошо подходят для задач, где данные являются последовательными, такими как генерация текста, музыки и видео. В последние годы RNN также стали популярными в области временных рядов благодаря своей способности обрабатывать многомерные наборы данных. Однако RNN не были предназначены для работы с time-series данными, они обрабатывают данные, как серию токенов и не зависят от времени наблюдений. Рекуррентные сети имеют свои недостатки:

– Во-первых, градиенты, в данной архитектуре не затухают, но могут взорваться (exploding gradients). Если матрица весов заметно увеличивает норму вектора градиента при проходе через один «виртуальный слой» обратного распространения, получится, что при проходе через T слоев эта норма возрастет экспоненциально от T.

– Во-вторых, влияние текущего входа или текущего состояния сети, обычно не может распространяться слишком далеко. Влияние текущего входа затухает экспоненциально по мере удаления. Это серьезная проблема, которая не позволяет простым рекуррентным сетям обучаться распознавать далекие зависимости в данных.

2.4 Другие подходы по экстраполяции временного ряда

Для предсказания временного ряда обычно используют ARIMA, RNN, LSTM, и также используются экстремальные градиентные деревья, на пример LightGBM [60-62].

Проблема ARIMA состоит в невозможности уловить сложные паттерны в дате. Обучение RNN долго из-за затухающих и взрывающихся градиентов. LSMT решат эту проблему, но у этой архитектуры малый effective content size. Согласно работе [63], LSTM смотрит только на последние 200 токенов.

Для решения этой проблемы можно использовать трансформер. Трансформер смотрит на всю историю временного ряда, вне зависимости от того, как давно это было, тем самым улавливая паттерны, удаленные во времени. Однако у трансформера есть две проблемы: dot-product self attention нечувствителен к локальному контексту; и ограничения по памяти: затраты пропорциональны квадрату длины последовательности.

Prophet или «Facebook Prophet» - это библиотека с открытым исходным кодом для одномерного (с одной переменной) прогнозирования временных рядов, разработанная Facebook [64]. Эта библиотека пытается решить следующие проблемы, общие для многих бизнес-временных рядов:

–Сезонные эффекты, вызванные поведением человека: недельные, месячные и годовые циклы, впадины и пики в праздничные дни.

–Изменение тренда в связи с появлением новых продуктов и рыночных событий.

–Выбросы.

Модель LightGBM - это платформа с открытым исходным кодом, предоставленная авторами [65]. Алгоритм основан на дереве решений, улучшенном варианте алгоритма дерева решений с градиентным усилением (GBDT). Другой часто используемой моделью на основе GBDT являются модели XGBoost [66] и CatBoost [67]. По сравнению с другими моделями на основе GBDT, такими как модели XGBoost и CatBoost, модель LightGBM имеет 3 подхода для ускорения обучения: на основе гистограммы, односторонняя выборка на основе градиента (GOSS) и объединение эксклюзивных функций (EFB). Модель LightGBM имеет преимущество в более высокой скорости обучения и подходит для крупномасштабных наборов данных.

Целью алгоритма LightGBM является получение оценки $\widehat{F(X)}$ функции $F(X)$, отображающей X в Y, с минимизацией функции потерь $L(Y,F())$.

При повышении градиента каждый новый алгоритм (дерево) b_i добавляется к уже построенной композиции:

$$a_i(x) = a_{i-1}(x) + b_i(x) \tag{2.14}$$

Такой алгоритм корректирует ответы алгоритма $a_i(x)$ на правильные ответы на обучающей выборке. Если рассматривать несколько алгоритмов, то алгоритм такой:

$$L(Y,F(x)) = log(1 + exp(-Y\,F(x))) \quad (2.15)$$

Для задачи классификации функция потерь имеет несколько вариантов, один из вариантов:

$$L(Y,F(x)) = log(1 + exp(-Y\,F(x))) \quad (2.16)$$

$$\sum log(1 + exp(-y_i * (a_n(x_i) + s_i))) \rightarrow \min \quad (2.17)$$

2.5 Выводы по разделу, моделирования для прогнозирования временных рядов

Второй раздел содержит описание нескольких классических подходов по работе с временными рядами, в частности экстраполяции временного ряда. Была предоставлена математическая формулировка задачи регрессии для временного ряда на основе теории вероятности. Выполнена формулировка для экстраполяции данных для следующего классического подхода ARIMA. Описана модель, основанная на пространственном состоянии, которая является одной из проблем в теории оптимизации и открывает возможность интерпретации существования подхода на основе нейросетевых ОДУ. Также описаны подходы, близкие по идеи решения, такие как линейно-гауссовские модели или фильтр Калмана, которые также довольно часто используется в задачах оптимизации в динамических системах. Также уделено внимание RNN (LSTM) алгоритму и его подходу для решения задачи проблем в последовательных данных.

3 НЕЙРОСЕТЕВЫЕ ОБЫКНОВЕННЫЕ ДИФФЕРЕНЦИАЛЬНЫЕ УРАВНЕНИЯ

Введение нейронных ОДУ [68] и других неявных нейро сетевых архитектур [69] позволило открыть новую парадигму машинного обучения, тесно связанную с методами моделирования динамических систем [70]. Например, нейронные ОДУ можно рассматривать как динамические системы с непрерывным временем и как бесконечно мало связанные ResNet, в то время как последние можно рассматривать, как эйлеровую дискретизацию нейронных ОДУ. Таким образом, они используют богатую доступную теорию с обеих сторон, предлагая эффективность памяти, в то время как их рекуррентные аналоги обладают способностью обрабатывать нерегулярные данные и подходят для решения генеративных задач и временных рядов (особенно в физике), приобретая актуальность как для современного машинного обучения, так и для традиционного математического моделирования.

3.1 Обыкновенные дифференциальные уравнения и нейросетевые архитектуры

Рассмотрим обыкновенное дифференциальное уравнение первого порядка с заданным начальным условием [71]:

$$\begin{cases} \frac{\partial z(t)}{\partial t} = f(z(t),t), t_0 \leq t \leq t_1 \\ z(t_0) = z_0 \end{cases} \tag{3.1}$$

Первый вопрос, которым обычно задаются, это существует и единственное ли решение уравнения (3.1). Обычно это проверяется, при условии, что $f(z(t),t)$ является липшицевым, что обычно верно для нейронной сети, которая обычно представляет собой композицию липшицевых функций. В этом случае применима теорема Пикара о существовании решения.

Теорема 1 (Теорема Пикара). *Пусть* $f : [0,T] \times \mathrm{R}^d \to \mathrm{R}^d$ *непрерывна на множестве* R *по t и выполнено условие Липшица по y. Т.е. существует число* $C > 0$*, такое что для всех* t,y_1,y_2 *выполняется условие* $\|f_\theta(t,y_1) - f_\theta(t,y_2)\| \leq C\|y_1 - y_2\|$. *Тогда решение задачи (3.1) и (3.2) существует, единственно и дифференцируемо по крайне мере на промежутке* $f : [0,T] \times \mathrm{R}d$.

Доказательство. Доказательство представлено в книге [72]. □

По сравнению с моделями машинного обучения, не являющимися дифференциальными уравнениями, существуют две дополнительные проблемы, о которых следует всегда помнить:

1. Необходимо находить численное решение дифференциального уравнения. (Аналитическое решение в основном тяжело найти).

2. Необходимо иметь возможность обратного распространения через дифференциальное уравнение, чтобы получить градиенты для параметров нейронной сети θ.

В главе 3.2 будут более подробно рассмотрены оценки и метод обратного распространения.

Физический смысл решения ОДУ: по заданному уравнению скорости $\frac{\partial z(t)}{\partial t}$ найти уравнение траектории $z(t)$. Дифференциальное уравнение (3.1) эквивалентно интегральному уравнению

$$z(t) = z_0 + \int_{t_0}^{t_1} f(z(\theta),\theta)\, d\theta \tag{3.2}$$

Как правило, на практике искать аналитическое решение $z(t)$ затруднительно, поэтому (3.1) или (3.2) решают численно. Одним из наиболее простых методов численного решения ОДУ является метод Эйлера. От непрерывной на отрезке $[t_0,t_1]$ задачи переходят к дискретной задаче, определенной на равномерной сетке с фиксированным шагом δ. Производную аппроксимируют разностной схемой вида

$$\frac{\partial z(t)}{\partial t} \approx \frac{z(t+\delta) - z(t)}{\delta}$$

и переходят к решению задачи

$$\frac{z(t+\delta) - z(t)}{\delta} = f(z(t),t)$$

Тогда $z(t)$ в узлах сетки находятся в явном виде $z(t + \delta) = z(t) + \delta * f(z(t),t)$.

Для того чтобы показать взаимосвязь метода Эйлера с нейросетевыми архитектурами, рассмотрим $z(t)$ - это t-й слой нейросети, а $f(z(t),t)$ - некоторое нелинейное преобразование t-го слоя. Во многих нейросетевых архитектурах (residual networks или recurrent neural network) следующий t+1-й слой получают как сумму предыдущего слоя и нелинейного преобразования:

$$z(t + 1) = z(t) + f(z(t),t) \tag{3.3}$$

Данная схема в точности совпадает со схемой решения ОДУ методом Эйлера при $\delta = 1$. Таким образом, перечисленные выше архитектуры, фактически, реализуют дискретный аналог некоторой непрерывной «траектории» $z(t)$. Нейросетевые ОДУ предоставляют способ работы с непрерывными траекториями $z(t)$ без перехода к их дискретным аналогам.

Нейронное дифференциальное уравнение — это дифференциальное уравнение, использующее нейронную сеть для параметризации векторного поля. Канонический пример - нейронное обыкновенное дифференциальное уравнение.

3.2 Нейросетевые ОДУ

Пусть задано параметризованное семейство функций $f(z(t),t,\theta)$, и пусть при каждом фиксированном значении параметров Θ данные функции задают ОДУ вида

$$\frac{\partial z(t)}{\partial t} = f(z(t),t), t_0 \le t \le t_1 \tag{3.4}$$

На практике $f(z(t),t,\theta)$ - это нейросеть с параметрами θ. Пусть задана функция потерь $L(z(t_1))$, которую мы хотим минимизировать. Искать $z(t_1)$ можно численно как $z(t_1) = ODESolve(z(t_0),t_0,t_1,f,\theta)$, где $ODESolve()$ - некоторый метод численного решения ОДУ. Тогда $L(z(t_1)) = L(ODESolve(z(t_0),t_0,t_1,f,\theta)) = L(z(t_0),t_0,t_1,\theta)$. Таким образом, свободными параметрами, по которым мы можем вести оптимизацию, являются: начальное условие $z(t_0)$, начальный момент времени t_0, конечный момент времени t_1, параметры нейросети θ.

Для того чтобы иметь возможность искать минимум функции потерь градиентными методами, необходимо научиться считать градиенты по свободным параметрам. В первую очередь, рассмотрим схему вычисления $\frac{dL}{d\theta}$.

Можно показать, что

$$\frac{dL}{d\theta} = -\int_{t_0}^{t_1} a(t)^T \frac{\partial f(z(t),t,\theta)}{\partial \theta}\, dt \tag{3.5}$$

где $a(t) = \frac{\partial L}{\partial z(t)}$ - так называемое сопряженное состояние. Производная $\frac{\partial f(z(t),t,\theta)}{\partial \theta}$ считается аналитически. Интеграл (3.5) можно вычислить численно при условии, что мы знаем $a(t) = \frac{\partial L}{\partial z(t)}$.

В общем случае посчитать $a(t) = \frac{\partial L}{\partial z(t)}$ аналитически невозможно. Однако известно, что справедливо равенство

$$\frac{da(t)}{dt} = -a(t)^T \frac{\partial f(z(t),t,\theta)}{\partial \theta} \tag{3.6}$$

Производную $\frac{\partial f(z(t),t,\theta)}{\partial \theta}$ можно вычислить аналитически. Производная $\frac{\partial f(z(t),t,\theta)}{\partial \theta}$ является функцией от $z(t),t,\theta$. Таким образом, для численного решения ОДУ (3.7) достаточно знать значения $z(t)$ в узлах сетки и конечное состояние $a(t_1) = \frac{\partial L}{\partial z(t_1)}$. Значения $z(t)$ в узлах сетки мы можем вычислить, решая ОДУ (3.4). А конечное состояние $a(t_1) = \frac{\partial L}{\partial z(t_1)}$ считается аналитически (известно, как функция потерь зависит от ответа модели). Таким образом, можно численно вычислить значение $\frac{dL}{d\theta}$.

Для вычисления остальных градиентов рассматривают аугментированное ОДУ, в котором

$$z_{aug} = [z,\theta,t], f_{aug} = [f,0,1]$$

К аугментированному ОДУ применяют метод, описанный выше при нахождении $\frac{dL}{d\theta}$, и численно получают все необходимые градиенты, схема вычисления которых представлена ниже.

Algorithm 1: Псевдокод вычисления градиентов в ОДУ

Input: параметеры θ, начальное время t_0, конечное время t_1, конечное состояние $z(t_1)$, градиент $\frac{\partial L}{\partial z(t_1)}$

$s_0 = [z(t_1), \frac{\partial L}{\partial z(t_1)}]$ ▷ Определение начального состояния

Function AugDynamics($[z(t_1), \frac{\partial L}{\partial z(t_1)}], t, \theta$):

  return $[f(z(t), t, \theta), -a(t)\frac{\partial L}{\partial z}, -a(t)\frac{\partial L}{\partial \theta}]$ ▷ Вычисление Якобиана

$[(z(t_0), \frac{\partial L}{\partial z(t_0)}, \frac{\partial L}{\partial \theta}] = ODESolve(s_0, AugDynamics, t_1, t_0, \theta)$

return $\frac{\partial L}{\partial z(t_0)}, \frac{\partial L}{\partial \theta}]$ ▷ Решение обратного во времени ОДУ

Очевидно, что чем точнее мы хотим искать минимум функции потерь, тем точнее нам надо искать производные. Таким образом, возникает некоторый компромисс между точностью решения *ODESolve* и между вычислительной эффективностью.

Во многих приложениях начальное условие $z(t_0)$ может быть задано. Например, пусть дана обучающая выборка (x,y), где x - начальные данные, а y - некоторые целевые значения. Тогда в качестве начальных условий ОДУ задаются входные данные $z(t_0) = x$. В этом случае оптимизация по $z(t_0)$, очевидно, не ведется.

Обучение нейронного дифференциального уравнения обычно означает обратное распространение через решение дифференциального уравнения. Имеется несколько способов выполнить данную операцию, три основных способа дифференцирования через ОДУ:

–Дискретизация, затем оптимизация – неэффективна в использовании памяти (увеличивает объем использованной памяти), но точна и быстра;

–Оптимизация, затем дискретизация - эффективна в использовании памяти, не точна и незначительно медленна в расчетах;

–Обратное решение ОДУ – эффективна в использовании памяти и точна, но незначительно медленна в расчетах.

Предпочтительным подходом является дискретизация, а затем оптимизация. Если это невозможно, как правило, это возникает из-за нехватки памяти, то следующим лучшим вариантом является подход обратного решения ОДУ. Наконец, если это неприемлемо, то можно использовать подход оптимизации, затем дискретизации, но он, как правило, является наименее предпочтительным подходом.

3.2.1 Дискретизация - оптимизация

Данный подход заключается в простом обратном распространении через внутренние операции решения дифференциального уравнения. Решение дифференциального уравнения содержит обычные арифметические операции сложения, умножения и т. д., каждая из которых дифференцируема. Учитывая,

что операция решения представляет собой композицию дифференцируемых операций, она также дифференцируема. Производные вычисляются относительно дискретизированной версии дифференциального уравнения, вычисленного решения, и не по отношению к идеализированному уравнению с непрерывным временем.

Преимущества данного подхода:

Точность градиентов. Вычисленные градиенты будут точными для дискретной модели, по сравнению с численными методами, которые имеют приблизительные вычисления.

Скорость. Часто это самый быстрый способ обратного распространения. Одна из причин этого заключается в том, что полный граф вычислений известен до выполнения обратного распространения, поэтому базовая библиотека автодифференцирования может лучше использовать параллелизм.

Простота реализации. Реализация подхода дискретизируй, затем оптимизируй, как правило, проста: при условии, что решение дифференциальных уравнений написано в автодифференцируемой среде (такой как PyTorch или TensorFlow), градиенты могут автоматически рассчитываться обычным для этих фреймворков способом.

Недостатки данного подхода:

Неэффективное использование памяти. Этот подход неэффективен с точки зрения использования памяти, так как каждая внутренняя операция решения должна быть сохранена. Если затраты памяти на запись операций одного шага дифференциального уравнения составляют N, а T - временной горизонт, тогда этот подход потребляет $O(NT)$ памяти.

Сложность реализации. Если решение дифференциальных уравнений описано без написания в автодифференцируемой среде, то этот подход практически невозможно реализовать.

Для того, чтобы решить проблему неэффективного использования памяти возможно использовать контрольные точки, которые будут содержать значения решения после прямого прохода и использоваться для восстановления значений во время обратного прохода. Этот подход используется в алгоритмах глубокого обучения и в работах [73; 74] рассматривается данный подход для нейронных ОДУ.

3.2.2 Оптимизация - дискретизация

Основная идея подхода оптимизация, затем дискретизация заключается в дифференциации модели для непрерывного времени. Возможно найти численное решение, выполняя обратный обход по параметрам нейронной сети. В работах [68, 75] описан данный подход и его применения для ОДУ.

Рассмотрим теорему, которая доказывает возможности дифференцирования в обратном порядке для уравнения 3.1 с помощью метода сопряженных уравнений, предложенного Понтрягиным Л.С. [76].

Теорема 2. *Пусть выполняется* $y_0 \in \mathrm{R}^d$ *и* $\theta \in \mathrm{R}^m$. *Также* $f : [0,T] \times \mathrm{R}^d \to \mathrm{R}^d$ *непрерывна на множестве* R *по* t *и выполнено условие Липшица по* y. *Тогда существует единственное решение уравнения по крайне мере на промежутке* $f : [0,T] \times \mathrm{R}^d$:

$$\begin{cases} \frac{\partial y(t)}{\partial t} = f_\theta(y(t),t) \\ y(0) = y_0 \end{cases} \tag{3.7}$$

Пусть $L = L(y(T))$ *(скалярная функция) будет являться функцией конечного значения для* $y(T)$. *Тогда* $a_y(t) = \frac{\partial L}{\partial y(t)}$ *и* $a_\theta(0) = \frac{\partial L}{\partial \theta}$, *где* $a_y : [0,T] \to \mathrm{R}^d$ *и* $a_\theta : [0,T] \to \mathrm{R}^d$ *являются решением системы уравнений:*

$$\begin{aligned} a_y(T) &= \tfrac{dL}{dy(T)}, & \tfrac{da_y}{dt}(t) &= -a_y(t)^T \tfrac{\partial f_\theta}{\partial y}(t, y(t)) \\ a_\theta(T) &= 0, & \tfrac{da_\theta}{dt}(t) &= -a_y(t)^T \tfrac{\partial f_\theta}{\partial \theta}(t, y(t)) \end{aligned} \tag{3.8}$$

Доказательство. Докажем уравнение для a_y, а для уравнения с θ можно получить заменив y на $[y,\theta]$ и f_θ на $[f,0]$ (то есть θ рассматривается как дополнительное состояние).

Зная, что y непрерывна и f_θ непрерывна и дифференцируема по y, тогда $t \mapsto \frac{\partial f_\theta}{\partial y}(t,y(t))$ является непрерывной функцией и отрезке $[0,T]$ и ограниченно числом $C > 0$. Соответственно для $a \in \mathrm{R}^d$, тогда $(t, a) \mapsto -a^T \frac{\partial f_\theta}{\partial y}(t, y(t))$

является липшицево для a с липшицевым числом C (зависимая от t). Поэтому основываясь на теореме 1, решение существует и единственно для уравнения 3.8.

Требуется доказать, что $a(t)^T = \frac{dL}{dy(t)}$.

Рассмотрим две точки $s,t \in [0,T]$ с $s < t$, тогда:

$$y(t) = y(s) + \int_s^t f_\theta(u,y(u))du,$$

используя формулу Лейбница для дифференцирования, получим уравнение:

$$\frac{dy(t)}{dy(s)} = I_{d\times d} + \int_s^t \frac{\partial f_\theta}{\partial y}(u,y(u))\frac{\partial y(u)}{\partial y(s)}du, \tag{3.9}$$

Данное уравнение является уравнением чувствительности прямого прохода из теоремы 1 и существует решение на основе теоремы 1.

Для $s,t \in [0,T]$ с $s < t$, возьмем уравнение и продифференцируем $a(t)^T \frac{dL}{dy(t)}$:

$$\frac{d}{dt}\left(a_y(t)^T\frac{dy(t)}{dy(s)}\right) =$$
$$\frac{da_y}{dt}t^T\frac{dy(t)}{dy(s)} + a_y(t)^T\frac{d}{dt}\left(\frac{dy(t)}{dy(s)}\right) =$$
$$\frac{da_y}{dt}t^T\frac{dy(t)}{dy(s)} + a_y(t)^T\frac{\partial f_\theta}{\partial y}(t,y(t))\frac{dy(t)}{dy(s)} =$$
$$\left(\frac{da_y}{dt}t^T\frac{dy(t)}{dy(s)} + a_y(t)^T\frac{\partial f_\theta}{\partial y}(t,y(t))\right)\frac{dy(t)}{dy(s)} = 0,$$

третью строку данного уравнения можно получить, дифференцируя уравнение 3.9, поэтому получаем следующие уравнение:

$$a_y(t) = a_y(t)^T\frac{dy(t)}{dy(t)} = a_y(T)^T\frac{dy(T)}{dy(t)} = \frac{dL}{dy(t)},$$

что является $\frac{dL}{dy_0} = \frac{dL}{dy(0)} = a_y(0)$.

□

Уравнение 3.8 является сопряженным уравнением, это позволяет использовать метод обратного распространения ошибки для обучения нейросетевых дифференциальных уравнений.

Уравнение 3.8 также требует знания решения *y* в качестве входных данных. Обычный подход состоит в том, чтобы найти решения *y*, дополнив уравнения 3.8 исходным нейро ОДУ, решенным в обратном направлении, начиная с численного приближения к *y*(*T*), вычисленного на прямом проходе. В интегральной записи это выглядит следующим образом:

$$y(t) = y(T) + \int_T^t f_\theta(s,y(s))ds,$$

Данный метод называется сопряженный или оптимизируй, затем дискретизируй. Производные вычисляются относительно идеализированной модели с непрерывным временем, а затем сами сопряженные уравнения 3.8 должны быть дискретизированы.

В уравнении 3.8 вектор матрица представляет собой матрицу Якоби. Для реализации в программном коде необходимо выполнять автоматическое дифференцирование для вычисления градиента по графу вычисления.

Имеются уже реализованные алгоритмы автоматического дифференцирования, одним из первых это было реализовано в библиотеке Theano, разработанной в университете Монреаля, в группе глубокого обучения Йошуа Бенджи [77], также реализация имеется в библиотеках ThensorFlow [78] и PyTorch [79]. С деталями реализации автоматического дифференцирования можно ознакомиться в приложении А.

3.3 Генеративные модели временных рядов

Пусть имеется временной ряд $x_{t1},...,x_{tN},...,x_{tM}$. Причем значения $x_{t1},...,x_{tN}$ мы наблюдаем - это обучающая выборка, а значения $x_{tN+1},...,x_{tM}$ предсказываем - тестовая выборка. Рассматривается следующая модель. Обучающая выборка подается на вход рекуррентной нейросети (кодировщик). Кодировщик выдает параметры, как правило, нормального распределения: матожидание μ и стандартное отклонение σ. Из полученного распределения сэмплируется случайная величина $z_{t0} \sim q(z_{t0}|\mu,\sigma)$. Значение z_{t0} подается на вход другой рекуррентной нейросети (декодировщик), которая обучается восстанавливать значения $x_{t1},...,x_{tN}$, поданные на вход кодировщику. Затем генерируются оставшиеся "тестовые" значения. Рекуррентная сеть использует дискретные преобразования вида (3.3), которые авторы статьи заменили на непрерывный аналог ОДУ-сетью. Схема работы ОДУ-сети изображена на рисунке 3.1.

Применение нейронных сетей к данным с нерегулярной выборкой, таким как промышленные данные, временной ряд в наблюдениях технического устройства или объекта является нетривиальной задачей. Как правило, наблюдения помещаются в ячейки (gate) фиксированной размерности, а скрытая динамика дискретизируется для последующего расчета. Это приводит к трудностям с отсутствующими данными и плохо определенными скрытыми переменными. Отсутствующие данные можно устранить с помощью генеративных моделей временных рядов [80, 81] или данных вставок (data imputation) [82].

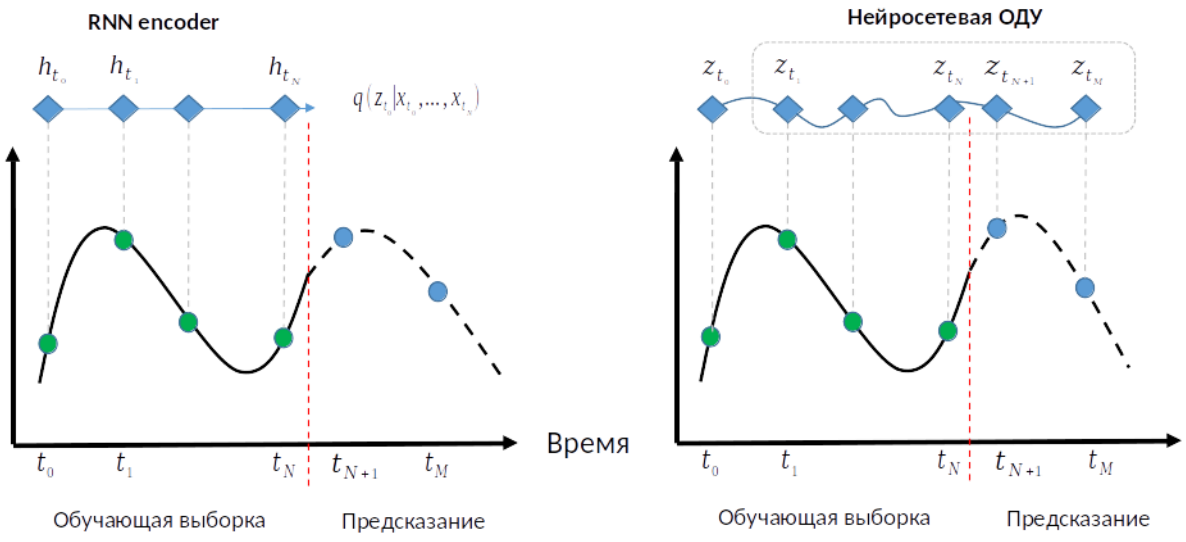

Рисунок 3.1 - Сравнение концепта RNN и нейросетевых ОДУ для временного ряда.

3.3.1 Обучение нейросетевых ОДУ

Другой подход объединяет информацию о метках времени с входными данными RNN [83]. Данный подход не позволяет делать предсказания достаточно точно [84]. В последнее время появился интерес к моделированию временных рядов с использованием ОДУ, которые естественным образом могут

обрабатывать непрерывное время. Раисси и др. [85] использовали нейронную сеть для изучения функциональной формы ОДУ, для каждой последовательности. Лонг и др. [86] изучили форму дифференциальных уравнений в частных производных с помощью свертки ядра.

Используя платформу, неройросетевые ОДУ представляют генеративный подход к моделированию временных рядов с непрерывным временем. Наша модель представляет каждый временной ряд скрытой траекторией.

Каждая траектория определяется из локального начального состояния, z_{t0}, и глобального набора скрытой динамики, общей для всех временных рядов. Мы стремимся изучить основную динамику временного ряда вместо последовательной обработки временного ряда. В модели траектория каждого временного ряда определяется скрытым вектором z_{t0}. Вектор z_{t0} предоставляется в качестве начального условия для решателя ОДУ. При заданных временах наблюдения $t_0,t_1,...,t_N$ и начальное состояние z_{t0}, решатель ОДУ выдает $z_{t1},...,z_{tN}$, которые описывают латентное состояние при каждом наблюдении в моменты времени $t_i;i = 1...N$. Мы формально определяем эту генеративную модель с помощью процедуры выборки:

$$zt0 \sim p(zt0) \quad z_{t1},...,z_{tN} = ODESolve(z_{t0},f,\theta_f,t_0,t_1,...,t_N) \quad (3.10)$$ для каждого$x_{ti} \sim p(x|z_{ti},\theta_x)$

Последовательность $z_{ti};i = 1...N$ может быть представлена, как траектория в скрытом пространстве. Функция f является не зависящей от времени функцией, которая принимает значение z на текущем временном шаге и выводит градиент: $\frac{\partial z(t)}{\partial t} = f(z(t),\theta_f)$. Данная функция параметризуется с помощью нейронной сети. Необходимо отметить, что:

–не указывается конкретная форму ОДУ;

–используется линейные ОДУ первого порядка;

–ОДУ не зависит от времени (градиент зависит только от значения z, а не от времени).

Поскольку f не зависит от времени, для любого скрытого состояния $z(t)$ вся скрытая траектория определяется однозначно. Экстраполяция этой скрытой траектории позволяет нам делать прогнозы сколь угодно далеко вперед или назад во времени. Параметры нейронной сети, определяющей ОДУ, являются общими для всех примеров в обучающем наборе, в то время как латентные состояния z_{ti} зависят от последовательности.

3.3.2 Устойчивость нейросетевых ОДУ

Нейросетевые ОДУ более надежны по сравнению с моделями CNN. В данном разделе предоставлено интуитивное представление о надежности нейронных ОДУ. В нейросетевых ОДУ, обученная модель при наличии данных извлекает информативную карту признаков из обучающей выборки. Нейросетевые ОДУ, выступающие в роли отображения нелинейного

представления, принимают в качестве входных данных карту объектов и выполняют нелинейное отображение. На практике используется методика снижения веса во время обучения, которая упорядочивает норму весов в части извлечения признаков, чтобы можно было контролировать изменение карты признаков в условиях небольшого возмущения на входе. Нейросетевые ОДУ имеют небольшое изменение на карте признаков, которое не приведет к большому отклонению от исходного вывода, связанного с картой объектов.

Надежность нейросетевых ОДУ достигается за счет контроля разницы между соседними интегральными кривыми. По неравенству Гроунолла (см. теорему 3) известно, что разность между двумя конечными состояниями ограничена разностью между начальными состояниями, умноженной на экспоненту динамической константы Липшица. Однако напрямую ограничить постоянную Липшица динамики очень трудно. В качестве альтернативы мы предлагаем достичь цели управления выходным отклонением, выполнив два шага: устранение зависимости динамики от времени и наложение определенного стационарного ограничения.

Теорема 3. *Пусть* $U \subset \mathrm{R}^d$ *задано множество, функция* $f : U \times [0,T] \to \mathrm{R}^d$ *является непрерывной и* $z_1,z_2 : [0,T] \to U$ *удовлетворяют начальным условиям:*

$$\frac{dz_1(t)}{dt} = f(z_1(t),t), z_1(t) = x_1$$

$$\frac{dz_2(t)}{dt} = f(z_2(t),t), z_2(t) = x_2$$

Предположим, существует константа $C > 0$, *такая, что для всех* $t \in [0,T]$ *выполняется условие:*

$$\|f(z_2(t),t) - f(z_1(t),t)\| \leq C\|z_2(t) - z_1(t)\|$$

Тогда для всех $t \in [0,T]$ *выполняется следующие условие:*

$$\|z_1(t) - z_2(t)\| \leq \|x_2 - x_1\|e^{Ct}$$

Доказательство. Нейросетевые ОДУ характеризуются уравнением (1) с динамическим состоянием $f_\theta(z(t),t)$, которое зависит как от состояния $z(t)$ в момент t, так и от времени t. В сравнении, если нейросетевое ОДУ изменено, чтобы быть инвариантным во времени, тогда зависимость от времени в динамике будет исключена. Следовательно, динамика зависит только от состояния z, и можно переписать функцию, как $f_\theta(z)$, а нейросетевое ОДУ может быть представлено в виде:

$$\begin{cases} \frac{dz(t)}{zt} = f_\theta(z(t)) \\ z(0) = z_0 \\ z_{end} = z(T) \end{cases} \tag{3.11}$$

Предположим, что $z_1(t)$ является решением уравнения 3.11 и является положительным числом. Мы определяем, что $M_1 = \{(z_1(t),t)|t \in [0,T], ||z_1(t) - z_1(0)|| \leq \epsilon\}$.Это множество содержит все точки на кривой $z_1(t)$ в течение $[0,T]$, которые также находятся внутри окрестности $z_1(0)$. Для некоторого элемента $(z_1(T'),T') \in M_1$, пусть $\tilde{z}_1(t)$ будет решением 3.11, которое имеет начальные условия $\tilde{z}_1(0) = \tilde{z}_1(T')$. Тогда мы получим для всех $t \in [0,\infty)$:

$$\tilde{z}_1(t) = z_1(t + {}^{T'}) \tag{3.12}$$

Свойство, показанное в уравнении 3.12, известно как стационарное свойство. Это указывает на то, что интегральная кривая $\tilde{z}_1(t)$ представляет собой сдвиг $z_1(t)$ по $-T'$. Мы можем рассматривать $\tilde{z}_1(0)$, как слегка искаженную версию $z_1(0)$, и нас интересует, насколько велика разница между $\tilde{z}_1(T)$ и $z_1(T)$. В надежной модели разница должна быть небольшой. Из уравнения 3.11 получаем $\|\tilde{z}_1(T) - z_1(T)\| = \|z_1(T + T') - z_1(T)\|$. Поскольку $T' \in [0,T]$, разность между $z_1(T)$ и $\tilde{z}_1(T)$ может быть ограничена следующим образом:

$$||\tilde{z_1}(T) - z_1(T)|| = \| \int_T^{T+T'} f_\theta(z_1(t))\, \mathrm{d}t \| \leq \| \int_T^{2T} |f_\theta(z_1(t))|\, \mathrm{d}t \| \tag{3.13}$$

где все нормы являются l_2 нормами и $|f_\theta|$ обозначают поэлементную абсолютную операцию вектор-функции f_θ. То есть разница между $\tilde{z}_1(T)$ и $z_1(T)$ может быть ограничена только с использованием информации кривой $z_1(t)$. Для любого $t' \in [0,T]$ и элемента $(z_1(t'),t') \in M_1$ рассмотрим интегральную кривую, начинающуюся с $z_1(t')$. Отличие между выходным состоянием этой кривой и $z_1(T)$ удовлетворяет неравенству 3.14.

Поэтому мы предлагаем добавить в функцию потерь дополнительный член L_{ss} при обучении стационарного нейросетевого ОДУ:

$$L_{ss} = \sum_{i=1}^{N} \| \int_T^{2T} |f_\theta(z_i(t))|\, \mathrm{d}t \| \tag{3.14}$$

где N - количество выборок в обучающей выборке, а $z_i(t)$ - решение, начальное состояние которого равно признаку i-й выборки. Выражением L_{ss} называется функция стационарных потерь. В устойчивой динамической системе состояния стабилизируются вокруг фиксированной точки, известной как стационарное состояние по мере того, как время стремится к бесконечности. Если мы сможем гарантировать, что L_{ss} мала, для каждой выборки выходы всех точек в M_i стабилизируются вокруг $z_i(T)$. Следовательно, модель надежна. Эта

модификация нейронного ОДУ получила название стационарного нейронного ОДУ, инвариантного во времени.

3.4 Асимптотическая сложность нейросетевых ОДУ

Нейросетевые ОДУ на каждой итерации проводят оценку функции f, для упрощения мы предположим, что f это нейронная сеть прямого порядка. Пусть K будет количеством слоев и H - максимальное число между размерностью скрытых слоев и размерностью данных. На основе данных предположений, память, необходимая для сохранения параметров нейронной сети, равна $\mathcal{O}(KH^2)$.

Асимптотическая сложность нейросетевых ОДУ линейно зависит от сложности f, которая определяется ODESolve. Сложность f, как прямого и обратного обхода нейронной сети, зависит от жесткости решения ОДУ и выбора решателя. Пусть сложность f прямого обхода будет представлена как L. Для того чтобы выполнить прямую оценку сложности вычисления нейросетевого ОДУ, нам необходимо оценить функцию L раз, что приводит к сложности $\mathcal{O}(LKH^2)$. Следующим шагом будет сохранение результата последней оценки нейронной сети, которая имеет стоимость памяти $\mathcal{O}(H)$.

Для случая с обратным распространением ошибки необходимо вычислить градиенты для каждого параметра нейронной сети f во всех L итерациях ODESolve. Асимптотическая сложность для данного случая равна $\mathcal{O}(LKH^2)$. Для вычисления памяти в нейросетевых ОДУ нужно рассчитать в два этапа: первое – хранение градиентов за L итераций, что потребовало $\mathcal{O}(LKH)$, и второе – память $\mathcal{O}(KH^2)$ для текущего вычисления градиентов. Итого, общая стоимость памяти для обратного распространения ошибки равна $\mathcal{O}(LKH + KH^2)$.

Если мы используем adjoint метод вместо обратного распространения, требуется решить ОДУ в обратном направлении во времени, т.е. больше не нужно хранить активацию L итераций ODESolve, так как результаты можно восстановить, решив исходную ОДУ в обратном направлении во времени. Поэтому необходимо сохранить только $\mathcal{O}(KH)$ активаций один раз. Переменные adjoint и градиенты имеют объем памяти $\mathcal{O}(KH^2)$, в итоге с использованием adjoint метода требуется $\mathcal{O}(KH + KH^2)$ памяти.

В сопряженном методе мы решаем другое ОДУ на обратном проходе, которое содержит следующие операции:

−повторное решение исходного ОДУ; – решение для сопряженной переменной;

−решение ОДУ для вычисления градиентов.

Как итог, количество вычислений функции будет отличаться от L, которая была обозначена ранее, поэтому количество вычислений на обратном проходе будем обозначать, как L. Решение обратного прохода имеет временную сложность $\mathcal{O}(LKH^2)$.

В таблице 3 представлена оценка асимптотической сложности и памяти нейронных ОДУ. Стоит отметить, что время обучения модели кодер-декодер, такое же как у модели RNN(LSTM), время выполнения является линейной

функцией и завесит от количества точек данных во временном ряду (обучающей выборке).

Таблица 3 - Асимптотическая сложность Нейросетевых ОДУ для обучающей выборки. K - количество слоев, H - размер скрытых слоев, L - количество итераций, T - время.

Модель	Память	Время
Нейросетевые ОДУ: Подсчет градиента с использованием adjoin method	$\mathcal{O}(KH + KH^2)$	$\mathcal{O}(LKH^2)$
RNN (LSTM)	$\mathcal{O}(8T^2 + 4T)$	$\mathcal{O}(T^3)$

3.5 Выводы по разделу Нейросетевых ОДУ

В настоящем разделе рассмотрены основные теоретические результаты нейросетевых ОДУ. Поставлена задача решения ОДУ, описан метод Эйлера, и показана его связь с нейросетевыми архитектурами. Описан метод обучения ОДУ-сетей. В разделе большое внимание уделено возможным приложениям теоретических результатов. Рассмотрены такие задачи как обучение с учителем, предсказание временных рядов. Показано, в чем ОДУ-сети «выигрывают» у стандартных архитектур.

Нейросетевые ОДУ могут быть применены для решения следующих задач:

–В задачах с временными рядами. Во временных рядах повсеместно встречаются беспорядочные или нерегулярные данные. Различные каналы могут наблюдаться на разных частотах, данные могут отсутствовать, временные ряды могут иметь переменную длину и т. д. Обработка дискретных данных в режиме непрерывного времени предлагает способ обработки нерегулярных данных на том же основании, что и «обычные» данные. Здесь также могут быть сделаны связи с такими темами, как идентификация системы и обучение с подкреплением, хотя они не будут широко освещаться в настоящей работе.

–Описание физических событий (биологическое, потока жидкости и т. д.). Модели дифференциальных уравнений, основанные на механистической теории, уже повсеместно используются в классическом математическом моделировании. Однако такие модели, основанные на теории, в какой-то момент перестают отражать детали реальности. Объединив существующие модели с глубоким обучением (с его высокопроизводительными аппроксиматорами функций), мы можем сократить разрыв между теорией и наблюдениями.

–Решение классических задач с традиционными подходами, основанными на «дискретизации» процесса обучения. Мы уже видели параллели между дифференциальными уравнениями и глубоким обучением: очень успешная стратегия разработки моделей глубокого обучения состоит в том, чтобы просто взять соответствующее дифференциальное уравнение, а затем дискретизировать его.

Таким образом, нейросетевые ОДУ имеют ряд преимуществ:

–Структура, подобная нейронной сети, обеспечивает аппроксимацию функций с высокой пропускной способностью и простоту обучения.

–Структура, подобная дифференциальному уравнению, предлагает убедительные априорные данные о пространстве модели, эффективности памяти и теоретическом понимании с помощью хорошо изученной и проверенной литературы.

В данной главе показана оценка асимптотической сложности и памяти нейронных ОДУ в сравнении с RNN(LSTM) и ее эффективность благодаря adjoin методу. По сравнению с классической литературой по дифференциальным уравнениям нейронные дифференциальные уравнения обладают практически беспрецедентными возможностями моделирования. По сравнению с современной литературой по глубокому обучению нейронные дифференциальные уравнения предлагают последовательную теорию того, «что делает хорошую модель».

4 ЭКСПЕРИМЕНТАЛЬНЫЕ ИССЛЕДОВАНИЯ ЭФФЕКТИВНОСТИ СИСТЕМЫ

Чтобы оценить точность предсказания нейросетевых ОДУ, использовались следующие метрики: RSME, MAE, MAPE.

RSME (root squatted error, среднеквадратичное отклонение) [87] - стандартное отклонение отклонений (ошибок предсказания). Остатки - это мера того, насколько далеко от линии регрессии находятся точки данных; RMSE – это мера того, насколько разбросаны эти остатки. Другими словами, эта мера подсказывает, насколько сконцентрированы данные вокруг линии наилучшего соответствия. Среднеквадратичная ошибка обычно используется в прогнозировании и регрессионном анализе для проверки экспериментальных результатов.

$$RSME = \sqrt{\frac{1}{n}(y - \hat{y})^2} \tag{4.1}$$

,где y – известные значения, $y\hat{}$ – предсказанное значение.

MAE (Mean absolute error, средняя абсолютная ошибка) измеряется как среднее значение абсолютной ошибки и является положительным значением [88]. В отличие от RMSE, изменения в MAE линейны и поэтому интуитивно понятны. В MAE разные ошибки не взвешиваются больше или меньше, но значения расчетов увеличиваются линейно с увеличением ошибок.

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y - \hat{y}|^2 \tag{4.2}$$

,где y – известные значения, $y\hat{}$ – предсказанное значение, n - количество элементов.

MAPE (Mean absolute percentage error), также известная как среднее абсолютное процентное отклонение, является метрикой оценки для проблем регрессии. Идея этой метрики – быть чувствительной к относительным ошибкам. Например, она не изменяется глобальным масштабированием целевой переменной.

$$MAPE = \sqrt{\frac{100\%}{n}\sum|\frac{y - \hat{y}}{y}|} \tag{4.3}$$

,где y – известные значения, $y\hat{}$ – предсказанное значение, n - количество элементов.

4.1 Применение методов data imputation

Методы imputation - это методы, при которых недостающие данные заполняются для создания полной матрицы данных, которая может быть проанализирована стандартными методами.

Проблема пропущенных данных возникает по разным причинам, таким как проблема в оборудовании, потери данных, аномальные значения, которые удаляются, как ошибочные и др.

Часто используется метод подстановки среднего, при котором отсутствующие значения заменяются переменным средним баллом. Ниже показаны несколько примеров методов imputation [82]:

–Linear interpolation;

–Polynomial interpolation;

–Kalman Smoothing;

–Moving Average;

–K-Nearest Neighbors;

–Random Forest;

–Multiple Singular Spectral Analysis;

–Expectation Maximisation;

Был выполнен анализ использования методов imputation для одного временного ряда по показанию pH в агрессивной среде, у данных имеются пропущенные данные, как показано на рисунке 4.1. Интервал времени в 18 месяцев (Июнь 2007 по Декабрь 2008) был скрыт для проведения эксперимента.

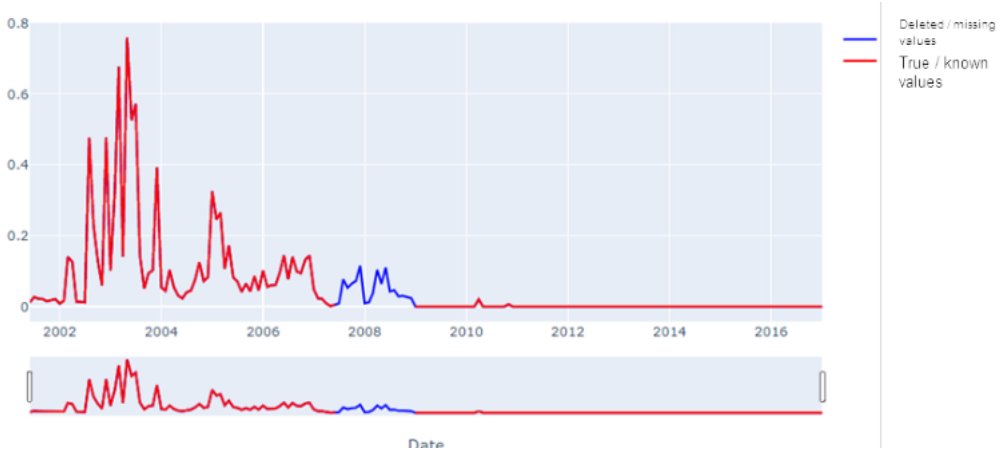

Рисунок 4.1 - Пример временного ряда с пропущенными данными.

Ниже предоставлены примеры реализации нескольких методов imputation для данного примера на рис. 4.2–4.7.

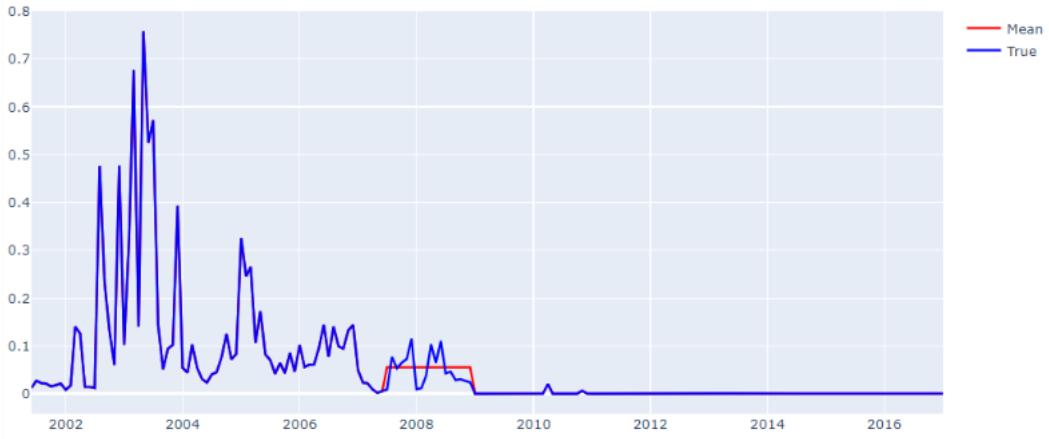

Рисунок 4.2 - Подстановка среднего значения для пропущенного интервала времени.

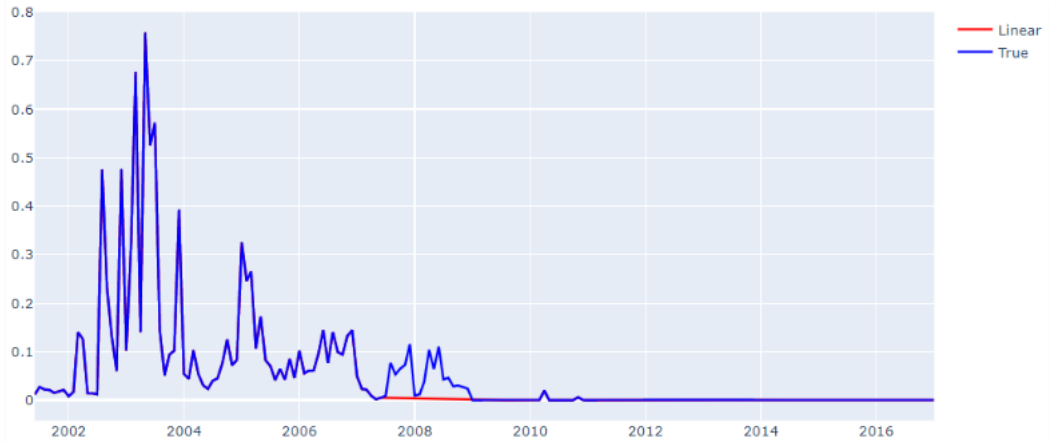

Рисунок 4.3 - Подстановка значений линейной интерполяции для пропущенного интервала времени.

Были рассмотренны методы imputation (подстановка среднего значения, значений линейной интерполяции, значений полиномиальной интерполяции, значений сплайн интерполяции и значений скользящим средним окном) для выбранной скважины и со скрытым интервалом значений, не показали хороших результатов. Пример подстановки значений скользящим средним окном из 18 шагов для пропущенного интервала времени представлен на рисунке 4.6. Без использования метрик визуально можно увидеть, что результаты не достаточные для того, чтобы решить данную проблему.

В связи с этим мы рассмотрели возможность использования алгоритмов LSTM и Prophet для решения задачи пропущенных значений, но с отличием в постановке задачи. Модели, построенные на LSTM и Prophet используются для данной задачи, как регрессионная постановка задачи. Пропущенный интервал (скрытый интервал) является выборкой, на которой будет проверяться точность модели. Интервал от начала до скрытого интервала является выборкой данных для обучения модели. Для оценки точности модели используется RMSE.

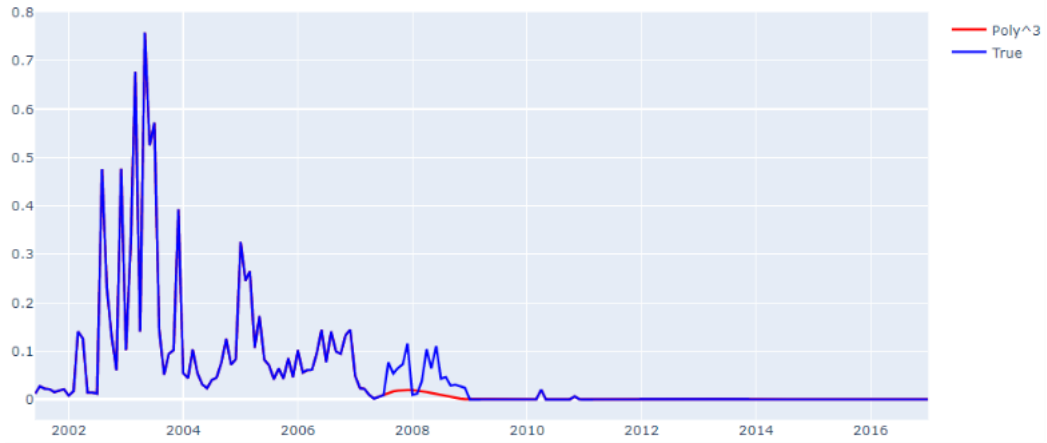

Рисунок 4.4 - Подстановка значений интерполяции 3го порядка для пропущенного интервала времени.

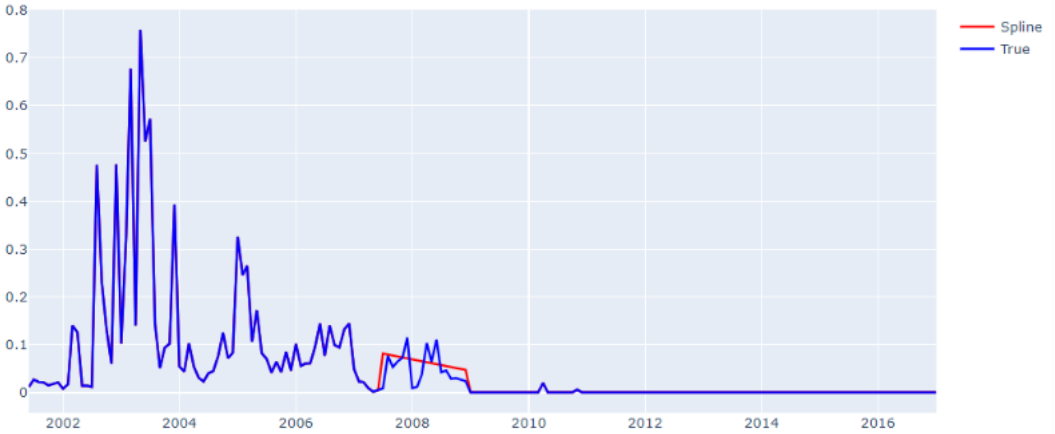

Рисунок 4.5 - Подстановка значений сплайн интерполяции для пропущенного интервала времени.

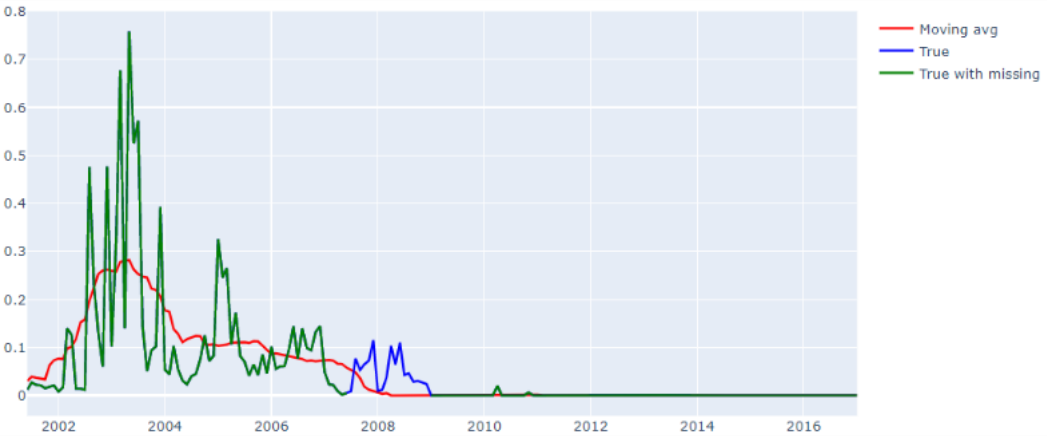

Рисунок 4.6 - Подстановка значений скользящим средним окном из 18 шагов для пропущенного интервала времени.

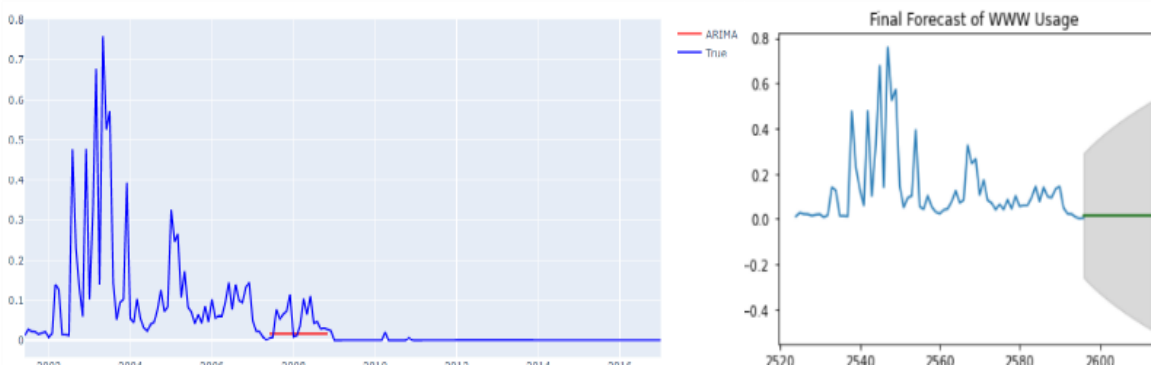

Рисунок 4.7 - Подстановка значений, предсказанных с помощью ARIMA для пропущенного интервала времени.

Prophet или «Facebook Prophet» - это библиотека с открытым исходным кодом для одномерного (с одной переменной) прогнозирования временных рядов, разработанная Facebook [64] стр.40.

Данная библиотека пытается решить следующие проблемы, общие для многих бизнес-временных рядов:

– Сезонные эффекты, вызванные поведением человека: недельные, месячные и годовые циклы, спады и пики в праздничные дни.

– Смена тренда в связи с новыми продуктами и рыночными событиями.

– Выбросы.

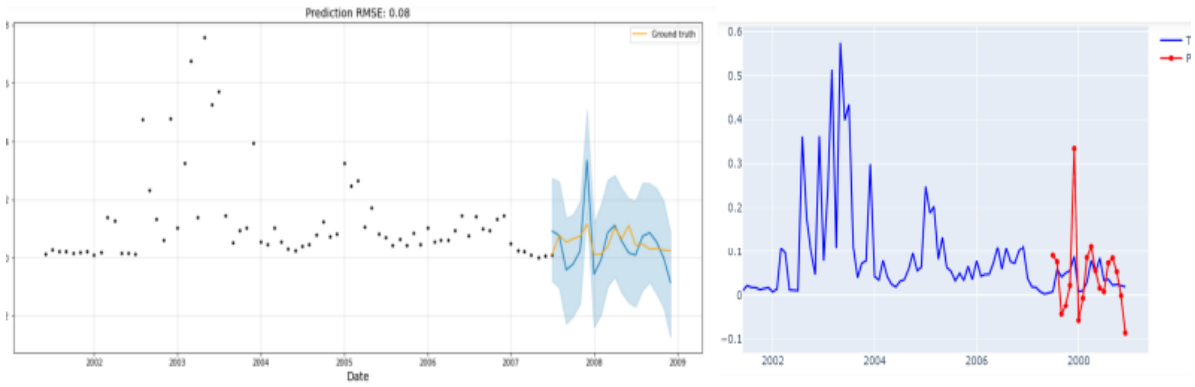

Рисунок 4.8 - Результаты предсказания с доверительным интервалом и результаты предсказания в другом представлении.

Точность модели Prophet составила RMSE: 0,07. Сеть с долговременной краткосрочной памятью или сеть LSTM - это рекуррентная нейронная сеть, которая обучается с использованием обратного распространения во времени и преодолевает проблему исчезающего градиента [89].

Вместо нейронов в сетях LSTM есть блоки памяти, которые связаны через слои. Блок имеет компоненты, которые делают его умнее классического нейрона, и память для последних последовательностей. Блок содержит вентили, которые управляют состоянием блока и выходом. Блок работает с входной последовательностью, и каждый вентиль в блоке использует блоки активации

сигмовидной формы, чтобы контролировать, срабатывают они или нет, делая изменение состояния и добавление информации, проходящей через блок, условным.

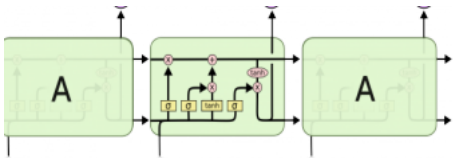

Рисунок 4.9 - Пример архитектуры LSTM.

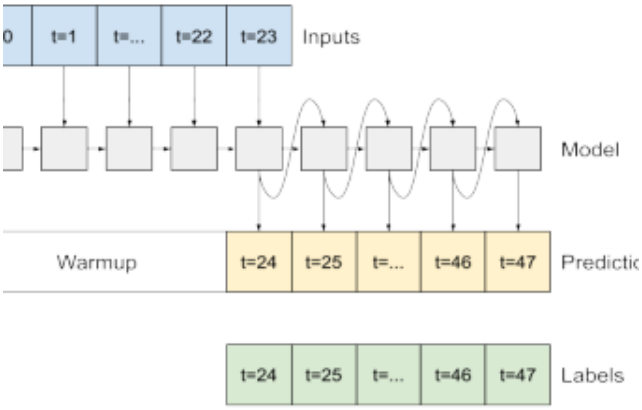

Рисунок 4.10 - Пример использования LSTM для предсказания значений во временном ряде.

Для проведения эксперимента была выбрана простая архитектура LSTM. Точность модели LSTM составила RMSE: 0,06.

Модели LSTM и Prophet показывают обнадеживающие результаты для задачи пропущенных данных в значениях временного ряда. Но данный подход имеет возможность использования, когда имеется большое количество исторических данных и незначительные интервалы пропущенных данных. В этой связи подход нейросетевых ОДУ имеет больше перспективы применения и эффективности по сравнению data imputation.

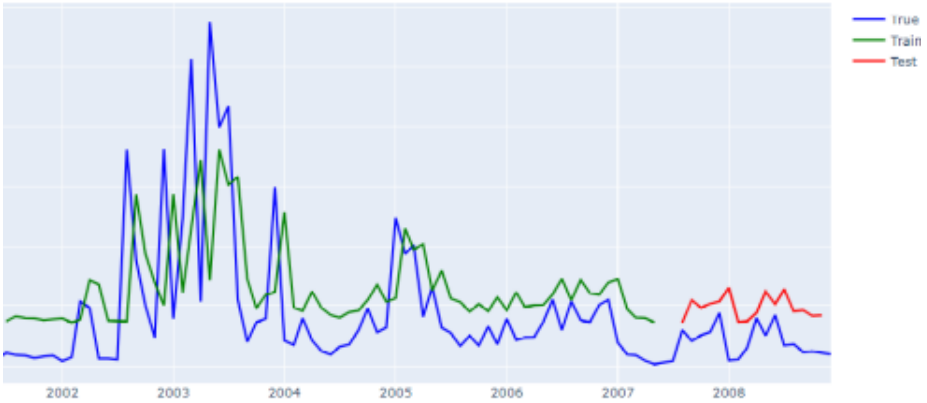

Рисунок 4.11 - Результаты предсказания LSTM для пропущенного интервала.

4.2 Эксперимент с данными от биогазовой установки

В этой главе предоставлена способность модели нейросетевых ОДУ экстраполировать временные ряды на реальных примерах, данных, собранных с биогазовой установки. Временные ряды имеют, как регулярные, так и нерегулярные временными точки. Временной ряд представляет собой собранные данные за один год по следующим показателям: температура, pH и давление из различных узлов биогазовой установки. Шаг сбора данных равен 10 мин, в среднем один временной ряд содержит 52 тысячи точек. Обучение проводилось на временном ряде за период 11 месяцев, последний месяц использовался для валидации модели [90].

Реализация выполнена с помощью библиотеки глубокого обучения TensorFlow [78] стр.4 и оптимизация нейронной сети выполнялась с помощью оптимизатора Adam [91]. Расчеты выполняются на настольном компьютере: процессор Intel Core i7 8700 3,2 ГГц, NVIDIA RTX 2080 8Gb и 32 ГБ оперативной памяти.

На рис. 4.12–4.14 показаны примеры экстраполяции нейросетевых ОДУ и их сравнение с оригинальными значениями. Реконструкции из скрытого ОДУ были получены путем выборки из апостериорных по скрытым траекториям и декодирования его в пространство данных.

На рисунке 4.12 показаны результаты экстраполяции временного ряда на один месяц с помощью нейросетевых ОДУ, обучение было выполнено за 9 месяцев. Результат модели на обучающей выборки по метрикам получился следующим *RSME* = 13.23, *MAE* = 9.68, *MAPE* = 12.6. За один месяц разница между предсказанным и оригинальными значениями в значениях температуры составляет 35.9, что является небольшим значением. Также на рисунке линии трендов предсказанных и оригинальных значений очень близки.

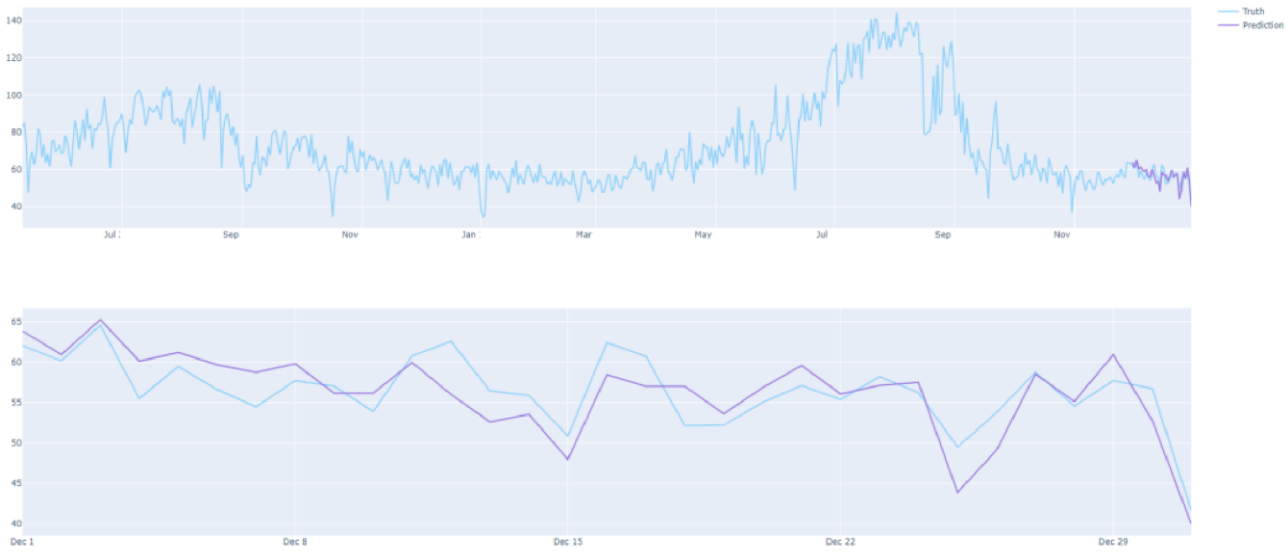

Рисунок 4.12 - Результаты предсказания LSTM для пропущенного интервала.

На рисунке 4.13 показаны результаты экстраполяции временного ряда на один месяц с помощью нейросетевых ОДУ, обучение было выполнено за 7 месяцев. Результат модели на обучающей выборке по метрикам получился следующим *RSME* = 6.68,*MAE* = 5.06,*MAPE* = 5.51. За один месяц разница между предсказанным и оригинальными значениями в значениях давления составляет 377.59, что является большим значением. Это подсказывает об отклонении работы установки, в данном случае причиной отклонения является поломка датчика сбора показаний давления в одном из резервуаров. Также на рисунке линии трендов предсказанных и оригинальных значений сильно отклоняются.

На рисунке 4.14, как и на предыдущем рисунке, показаны результаты экстраполяции временного ряда на один месяц с помощью нейросетевых ОДУ, обучение было выполнено за 8 месяцев, и обучающая выборка имеет цикличные показания. Результат модели на обучающей выборке по метрикам получился следующим *RSME* = 89.05, *MAE* = 71.03, *MAPE* = 34.36. Как и на предыдущем рисунке, за один месяц разница между предсказанными и оригинальными значениями в значениях температуры составляет 7507.93, что является большим значением. Это подсказывает об отклонении работы установки, в данном случае причиной отклонения является не соблюдение технического регламента, что привело к снижению температуры в резервуаре и приводит к снижению эффективности установки в целом. Также на рисунке линии трендов предсказанных и оригинальных значений сильно отклоняются.

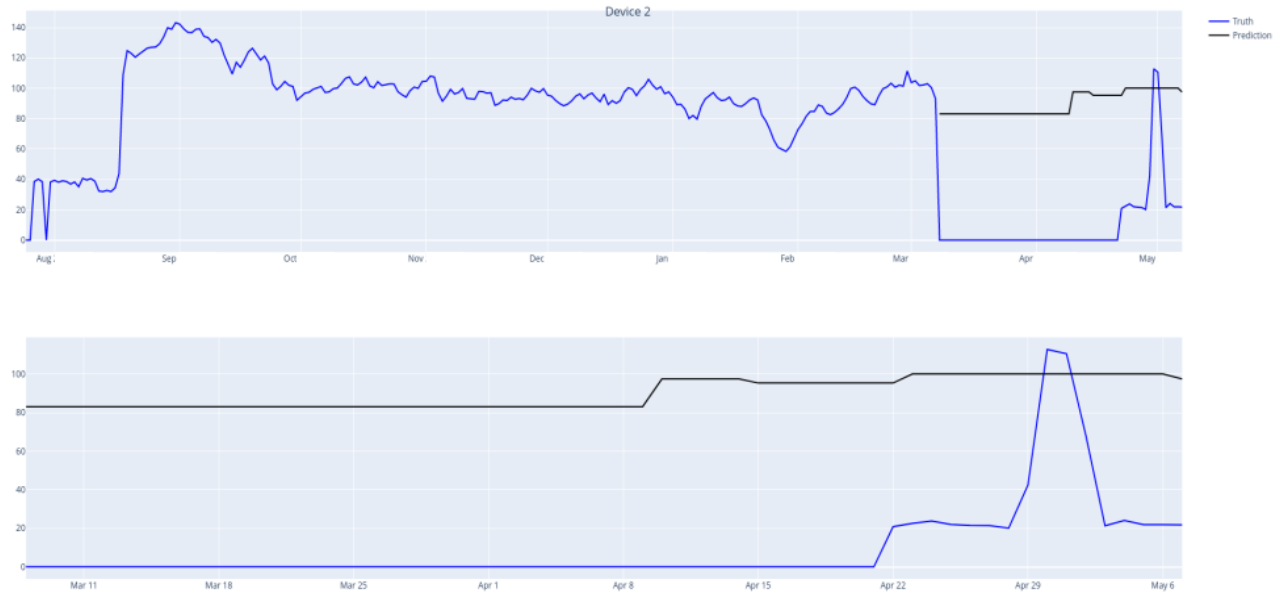

Рисунок 4.13 - Результаты предсказания LSTM для пропущенного интервала.

В таблице 4 представлено сравнение 4 алгоритмов предсказания за один месяц по тренду температурных показаний, обучающая выборка была за 11 месяцев собранных с биогазовой установки данных. Как можно заметить по трем

метрикам, нейросетевые ОДУ показали лучшие результаты, т.е. меньшие ошибки по метрикам RSME, MAE и MAPE.

Таблица 4 - Сравнение моделей по 3 метрикам для данных по температуре, собранных за 1 месяц работы установки.

Модель	RSME	MAE	MAPE
Prophet FB	8.5	7.81	7.8
LightGBM	8.1	7.5	7.3
LSTM	7.63	6.8	6.4
NeuralODE	6.68	5.06	5.51

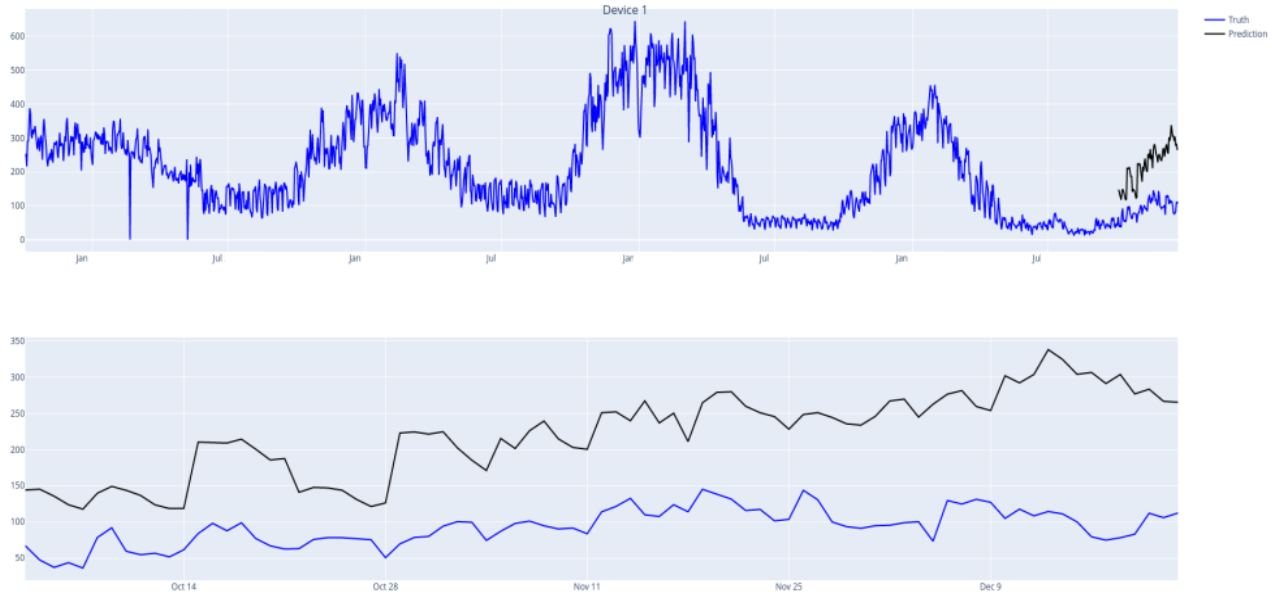

Рисунок 4.14 - Результаты предсказания LSTM для пропущенного интервала.

4.3 Описание системы мониторинга

Система мониторинга технических систем - это технология управления системой, которая сочетает в себе датчики в реальном времени, обработку данных для конкретных задач, например: выявление аномалий, прогнозирование и интерпретация данных для принятия решений. Хотя концепция интеллектуального мониторинга существует с момента внедрения ИТ в производство, начиная с 1990-х годов [92-94], в последние годы наблюдается всплеск интереса к данному направлению, в основном это связанно с быстрым развитием Интернета вещей (IoT), кибербезопасности и достижениями в области искусственного интеллекта. В недавнем отчете прогнозируется, что к 2025 году будет от 25 до 50 миллиардов подключенных устройств [95]. Каждое интеллектуальное устройство, способное воспринимать окружающую среду и обмениваться информацией по сети, становится потенциальным генератором

данных. Вместе эти устройства будут генерировать большие данные, которые можно представить, как 4V (объем - volume, разнообразие - variety, скорость - viscosity, достоверность - veracity). Такие объемы данных требуют своевременной обработки для улучшения понимания и принятия решений [96]. Возможности собирать, обрабатывать и анализировать большие данные в режиме реального времени во многих областях все еще не хватает [97].

Ключевыми требованиями к интеллектуальной системе мониторинга технического устройства являются обработка данных, такие как: извлечение, преобразование и загрузка данных, обнаружение событий и визуализация. Однако многие устаревшие платформы мониторинга построены на реляционных базах данных, требующих, чтобы данные были сначала сохранены и проиндексированы, прежде чем их можно будет обрабатывать, что создает задержку в обработки данных. Повсеместное присутствие интеллектуальных устройств и сенсорных сетей требует фундаментального сдвига в парадигме проектирования: от дизайна системы мониторинга на основе клиент-сервер к облачному или даже периферийному дизайну, где основная часть вычислений выполняется на периферии, например, интеллектуальные устройства мониторинга [98-100]. Независимо от платформы, фундаментальная задача связана с непрерывной обработкой потока информации по мере ее поступления, с сохранением данных.

На рисунке 4.15 показан процесс генерирования, обработки сданных последующей визуализацией и предсказанием аварийности на техническом устройстве. Одним из устройств для применения полученных результатов может быть биогазовая установка. Зная, насколько популярен данный вид генерирования энергии, а также зная насколько часто там проходят аварии, данное исследование было бы интересно данным компаниям.

На рисунке 4.15 показан общий дизайн системы для мониторинга технического устройства, который состоит из пяти уровней, а именно уровня сбора данных, уровня хранения и обработки данных, уровня интеллектуального анализа, уровня мониторинга и визуализации и уровня определения знаний по аварийности системы. На уровне сбора данных типы входных данных могут включать измерения временных рядов точечных значений и векторов (многомерные данные) и выходные данные модели (многомерные данные).

На уровне хранения используется база данных NoSQL (не реляционная) InfluxDB (https://github.com/influxdata/influxdb) для хранения данных с технических систем. В отличие от традиционных баз данных SQL, использующих предопределенную схему базы данных, базы данных NoSQL используют динамическую схему и лучше всего подходят для приложений, требующих высокой производительности, гибкости и масштабируемости. Для сбора и передачи данных для хранения используется экосистем Kafka, распространяемая Confluent. Уровень интеллектуального анализа - это разработанный алгоритм на основе нейросетевых ОДУ. Уровень мониторинга и визуализации – это дашборд, поддерживающий совместную визуальную аналитику данных.

Рисунок 4.15 - Схема потока данных и задачи, решаемые в проекте.

Kafka разработан вокруг четырех ключевых концепций: брокер, топик, производитель и потребитель. Kafka предоставляет способ организации сообщений, которые, в свою очередь, служат промежуточными контейнерами данных для записей, которые должны передаваться между приложениями/системами. Схема тематических данных определяется пользователем для различных типов датчиков. Внутри каждая тема организована в несколько разделов для более быстрого поиска информации и избыточности данных. Производитель Kafka пишет в тему, а потребитель Kafka читает из раздела. Брокер Kafka - это аппаратный узел в распределенной системе, который выполняет фактическое чтение и запись, а также балансировку нагрузки. Пользователь несет ответственность за определение производителя и потребителя. Kafka темы (topic) предоставляет разработчикам интерфейсы прикладного программирования (API) для создания настраиваемых производителей и потребителей. Кроме того, коннекторы Kafka позволяют настраивать источники/приемники, которые соединяют темы Kafka с известными приложениями или системами данных через стандартные интерфейсы, такие как JDBC (реляционные базы данных SQL), распределенную файловую систему Hadoop (HDFS) и Amazon S3. Пользовательские коннекторы используются для связи протоколы связи с уровнем сбора данных и уровнем обнаружения знаний, автоматизируя обмен информацией между уровнями.

Визуализация играет важную роль в получении знаний о техническом устройстве и поддержке принятия решений, особенно в науках об окружающей

среде [101]. В последние годы в бизнес-аналитике появилось несколько дашбордов с открытым исходным кодом. В работе было рассмотрено решение Apache Superset (https://superset.incubator.apache.org), данный продукт имеет богатый набор инструментов визуализации данных (включая карты), простой в использовании интерфейс для загрузки и преобразования данных и бесшовную интеграцию с часто используемыми хранилищами данных.

Сервис-ориентированная архитектура, представленная на рисунке 4.16, является общей. Для демонстрации мы развертываем Kafka Confluent версия 7.1.1 и Superset версия 1.4.2 в одной и той же системе Ubuntu Linux версия 20.04, работающей на сервере со следующими характеристиками: видеокарта GeForce RTX2080 SUPER 8GB, процессор Intel Xeon Bronze 3204 6 ядер и 64 ГБ оперативной памяти. Superset обслуживается с помощью веб-сервера nginx (https://www.nginx.com). Скрипты и код реализации алгоритма выполнены на Python.

Superset (https://superset.incubator.apache.org/) - это продукт с открытым исходным кодом, размещенный на платформе Apache Foundation Github, и разработан с использованием Flask (https://flask.palletsprojects.com/) очень компактного фреймворка Python (https://www.python.org/) для вебразработки. Часть, которая генерирует интерактивные графики, вместо этого использует NVD3 (http://nvd3.org/) - библиотеку javascript, построенную на D3.js [102]. Любая информационная панель, созданная с помощью Superset, состоит из серии графиков. Каждый из них можно изменить в размере, переместить относительно других или отобразить на весь экран. Кроме того, каждый набор данных, представленный на графике, также можно экспортировать в формате CSV или JSON или с помощью запросов SQL. Графики создаются на основе собранных данных, которые извлекаются из подключенных источников данных и которыми можно управлять в Apache Superset. Superset предоставляет два основных интерфейса: первый - это Rich SQL IDE (интерактивная среда разработки) под названием SQL Lab, с помощью которой пользователь может иметь немедленный и гибкий доступ к данным или писать определенные SQL запросы, второй - это анализ данных. Интерфейс позволяет преобразовывать таблицы данных в интерактивные визуализации. С деталями развертывания веб приложения можно ознакомиться в приложении Г.

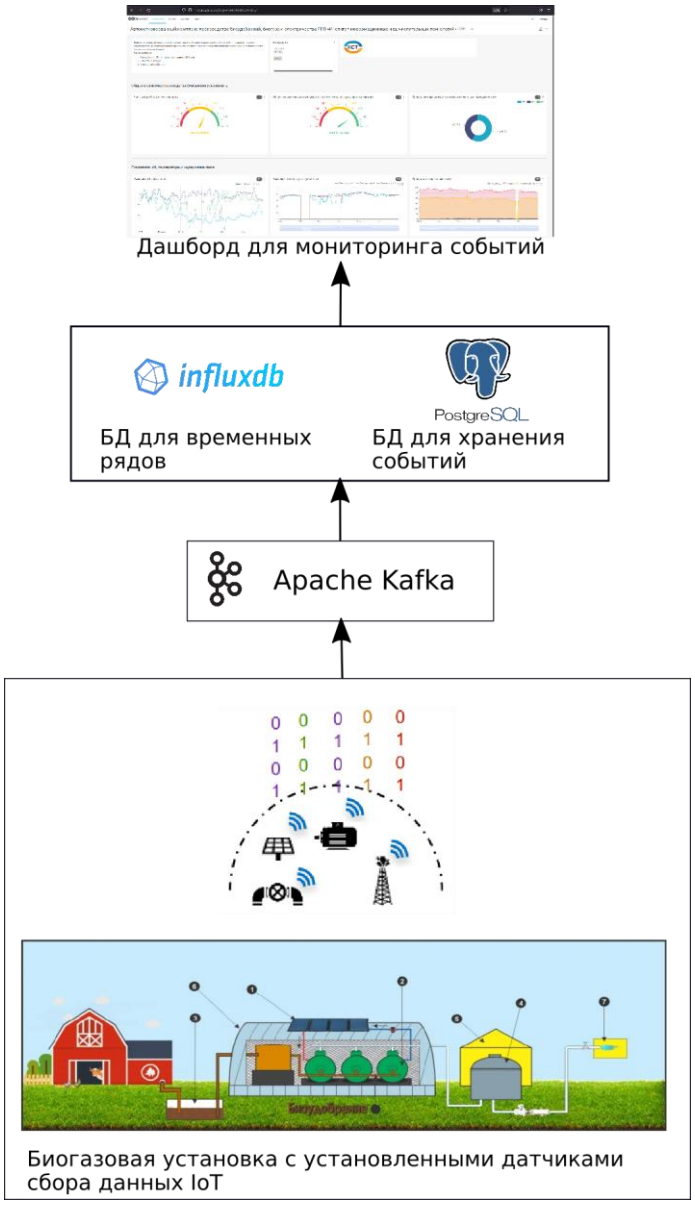

Рисунок 4.16 - Сервисно - ориентированная архитектура решения для мониторинга биогазовой установки.

4.4 Интерфейс дашборда

Приложение написано на Vue.js версии 2.6.12 с использованием Light Blue Vue admin dashboard. Приложение стандартно запускается на порту:3000, но при конфликте с запущенным PostgREST запускается на порту:3001. http://localhost:3001/#/app/dashboard - главная страница. Страница состоит из нескольких частей:

Рисунок 4.17 - Основная страница Dashboard.

Боковая панель, с возможностью перехода между страницами и центральная часть, где имеются следующие объекты:

–фильтрация по времени;

–последние показатели: pH, температура, давление, CH_4,CO_2,O_2,H_2S;

–схема реакторов;

–показатели: чистое производство энергии, компост, газы (CH_4,CO_2,O_2,H_2S).

Код основной страницы дашборда находится в файле *root//src//pages//Visits//*. Код состоит из двух основных частей:

Шаблон <template>...</template>. В рисунке 4.18 описывается разметка страницы с использованием HTML тегов.

JavaScript скрипт <script>...</script>. Здесь описывается логика страницы, импортированные компоненты и библиотеки, функции и переменные.

Используется синтаксис JavaScript.

```
1   /* eslint-disable no-console */
2   <template>
3     <div class="visits-page">
4       <Widget
5         title="<h5>Фильтры</h5>"
6         customHeader
7         collapse
8         class="w-100 datepicker-widget"
9         style="display: flex"
10      >…
72      </Widget>
73      <b-row v-if="data.loading">…
104     </b-row>
105     <b-row v-if="!data.loading">…
266     </b-row>
267   </div>
268 </template>
```

Рисунок 4.18 - Код Visits.vue <template>...</template>.

```
270  <script>
271  /* eslint-disable no-console */
272  import Vue from "vue";
273  import Widget from "@/components/Widget/Widget";
274  import Map from "./components/Map/Map";
275  import Calendar from "./components/Calendar/Calendar";
276  import AreaChart from "./components/AreaChart/AreaChart";
277  import AnimatedNumber from "animated-number-vue";
278  import Datepicker from "vuejs-datepicker";
279  import { en, ru } from "vuejs-datepicker/dist/locale";
280  import { GChart } from "vue-google-charts";
281  import Loader from "@/components/Loader/Loader";
282
283  export default {…
569  };
570  </script>
571
572  <style src="./Visits.scss" lang="scss" />
573
```

Рисунок 4.19 - Код Visits.vue <script>...</script> и <style . . ./>.

Страница Dashboard: http://localhost:3001/#/app/realtime - страница с графиками в реальном времени. Страница состоит из нескольких частей и представлена на рисунке 4.20.

Рисунок 4.20 - Страница realtime data

Дашбор состоит из боковой панели, с возможностью перехода между страницами и центральная часть, где находятся:

–фильтрация по времени;

–pH;

–температура в Цельсиях;

–газы (CH_4,CO_2,O_2,H_2S);

–чистое производство энергии.

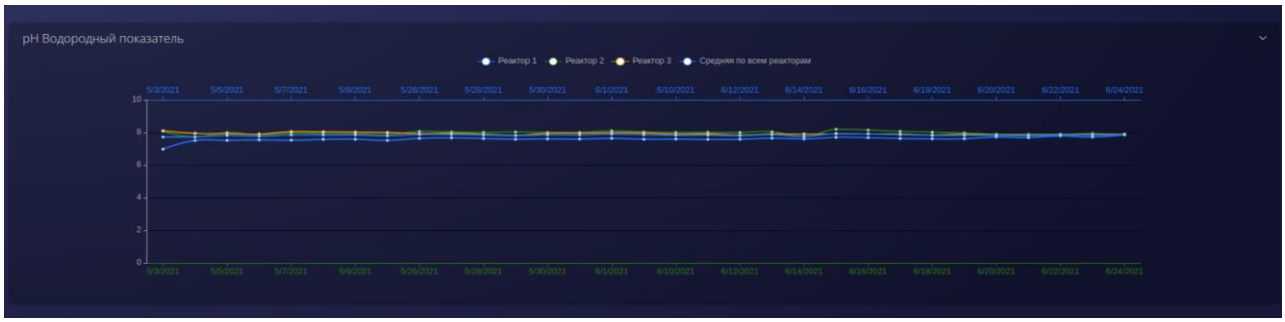

Рисунок 4.21 - График pH показателя.

Данный график показывает показатель pH за определенный промежуток времени: Ось Y - количественный показатель и ось X - временной показатель.

Рисунок 4.22 - График газов CH_4,CO_2,O_2,H_2S.

Код графиков в реальном времени находится в файле *root//src//pages//RealTime//*. Код состоит из двух основных частей: Шаблон <template>...</template>. Здесь описывается разметка страницы с использованием HTML тегов.

JavaScript скрипт <script>...</script>. Здесь описывается логика страницы, импортированные компоненты и библиотеки, функции и переменные.

Используется синтаксис JavaScript.

Рисунок 4.23 - График чистого производства энергии.

Рисунок 4.24 - Код Visits.vue <template>...</template>.

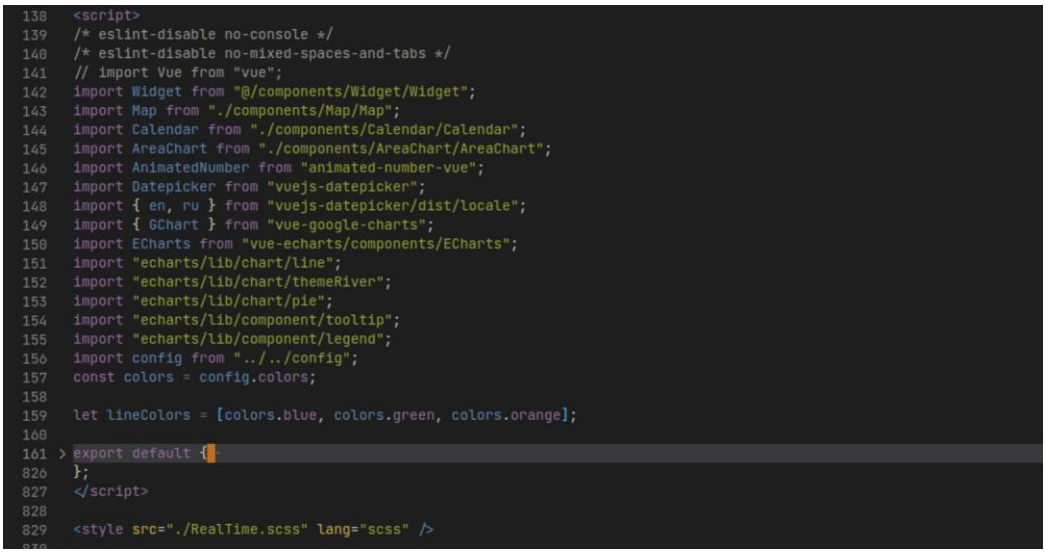

Рисунок 4.25 - Код Visits.vue <script>...</script> и <style . . . />.

4.5 Выводы по разделу, по эксперименту с данными и дашборд

В данной главе показан эксперимент, доказывающий эффективность предложенного подхода по сравнению с 3 алгоритмами (Prophet, LightGBM и LSTM). Нейросетевые ОДУ показали лучшие результаты, т.е. меньшие ошибки по метрикам RSME, MAE и MAPE. Кроме того, показана неэффективность методов вставки (methods imputation) для нестационарных и нецикличных данных. Данные методы зачастую используются для решения проблем нерегулярности временного ряда и приводят к смещенности предсказания обученных классических моделей машинного обучения. Кроме того, в данном разделе описана архитектура предлагаемого решения, который основан на open-source решениях. В деталях представлена архитектура разработанного дашборда.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1 Moreno V. C., Papasidero S., Scarponi G. E., Guglielmi D., Cozzani V. Analysis of accidents in biogas production and upgrading // Renewable Energy. - 2016. - T. 96. - C. 1127-1134.

2 GeneralElectric. How big data and the industrial internet can help southwest save $100 million on fuel. - URL: http://www.gereports.com/big - data - industrial - internet - can - help - southwest - save - 100 - million - fuel.

01.06.2021.

3 Qantas tech deal with GE tipped to save millions in fuel bills with smart app. - URL: http ://www . afr . com / technology / apps / business /qantas-tech-dealwith-ge-tipped-to-save-millions-in-fuel-bills-with-smart-app-20161006grw5kv. 01.06.2021.

4 MIT. GE’s big bet on data and analytics. - URL:

https://www.sloanreview.mit.edu/case-study/ge-big-bet-on-data-and-analytics.

01.06.2021.

5 Balevic D., Hartman S., Youmans R. Heavy-duty gas turbine operating and maintenance considerations // GE Energy, Atlanta, GA. - 2010.

6 Lee J., Scott L. Zero-breakdown machines and systems: productivity needs for next-generation maintenance. - 2006.

7 Jardine A. K., Lin D., Banjevic D. A review on machinery diagnostics and prognostics implementing condition-based maintenance // Mechanical systems and signal processing. - 2006. - T. 20, № 7. - C. 1483-1510.

8 Tahan M., Muhammad M., Karim Z. A. A framework for intelligent conditionbased maintenance of rotating equipment using mechanical condition monitoring // MATEC web of conferences. T. 13. - EDP Sciences. 2014. - C. 05011.

9 Scarponi G. E., Guglielmi D., Moreno V. C., Cozzani V. Risk assessment of a biogas production and upgrading plant // Chemical engineering. - 2015. - T. 43.

10 Santoli L. de, Paiolo R., Basso G. L. An overview on safety issues related to hydrogen and methane blend applications in domestic and industrial use // Energy Procedia. - 2017. - T. 126. - C. 297-304.

11 Montoya F. G., Gómez J., Cama A., Zapata-Sierra A., Martínez F., De La Cruz

J. L., Manzano-Agugliaro, F. A monitoring system for intensive agriculture

based on mesh networks and the android system // Computers and Electronics in Agriculture. - 2013. - T. 99. - C. 14-20.

12 Deniz N. N., Chelotti J. O., Galli J. R., Planisich A. M., Larripa M. J., Rufiner H. L., Giovanini L. L. Embedded system for real-time monitoring of foraging behavior of grazingmcattle using acoustic signals // Computers and Electronics in Agriculture. - 2017. - T. 138. - C. 167-174.

13 Schröder V., Schalau B., Molnarne M. Explosion protection in biogas and hybrid power plants // Procedia engineering. - 2014. - T. 84. - C. 259-272.

14 Cerrada M., Sánchez R. V., Li C., Pacheco F., Cabrera D., de Oliveira J. V., Vásquez R. E. A review on data-driven fault severity assessment in rolling bearings // Mechanical Systems and Signal Processing. - 2018. - T. 99. - C. 169-196.

15 Wall J. D., Harwood C. S., Demain A. Bioenergy. - ASM Press. 2008.
16 Silveira S. Bioenergy-realizing the potential. - Elsevier, 2005.
17 Shaukat S. Progress in biomass and bioenergy production. - BoD–Books on Demand, 2011.
18 Dahiya A. Bioenergy: Biomass to biofuels. - Academic Press, 2014.
19 Hakeem K. R., Jawaid M., Rashid U. Biomass and bioenergy. - Springer, 2016.
20 Holm-Nielsen J. B., Ehimen E. A. Biomass supply chains for bioenergy and biorefining. - Woodhead Publishing, 2016.
21 Chandola V., Banerjee A., Kumar V. Anomaly detection: A survey // ACM computing surveys (CSUR). - 2009. - T. 41, № 3. - C. 1-58.
22 Aggarwal C. C. An introduction to outlier analysis // Outlier analysis. - Springer, 2017. - C. 1-34.
23 Zohrevand Z., Glässer U., Tayebi M. A., Shahir H. Y., Shirmaleki M., Shahir
A. Y. Deep learning based forecasting of critical infrastructure data // Proceedings of the 2017 ACM on Conference on Information and Knowledge Management. - 2017. - C. 1129-1138.
24 Suter B. W. The multilayer perceptron as an approximation to a Bayes optimal discriminant function // IEEE transactions on neural networks. - 1990. - T. 1, № 4. - C. 291.
25 Deng L., Seltzer M. L., Yu D., Acero A., Mohamed A. R., Hinton G. Binary coding of speech spectrograms using a deep auto-encoder // Eleventh Annual Conference of the International Speech Communication Association. - Citeseer. 2010.
26 Zeng N., Wang Z., Zhang H., Liu W., Alsaadi F. E. Deep belief networks for quantitative analysis of a gold immunochromatographic strip // Cognitive Computation. - 2016. - T. 8, № 4. - C. 684-692.
27 Krizhevsky A., Sutskever I., Hinton G. E. Imagenet classification with deep convolutional neural networks // Advances in neural information processing systems. - 2012. - T. 25.
28 Mikolov T., Karafiát M., Burget L., Cernocký J., Khudanpur S. Recurrent neural network based language model. // Interspeech. T. 2. - Makuhari. 2010. - C. 1045-1048.
29 Zeng N., Zhang H., Song B., Liu W., Li Y., Dobaie A. M. Facial expression recognition via learning deep sparse autoencoders // Neurocomputing. - 2018. T. 273. - C. 643-649.
30 Li S., Jin X., Xuan Y., Zhou X., Chen W., Wang Y. X., Yan X. Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting // Advances in Neural Information Processing Systems. - 2019. - T. 32.
31 Wu N., Green B., Be X., O'Banion S. Deep transformer models for time series forecasting: The influenza prevalence case // arXiv preprint arXiv:2001.08317. - 2020.

32 Merembayev T., Amirgaliyev Y., Kunelbayev M., Yedilkhan D. Thermal Loss Analysis of a Flat Plate Solar Collector Using Numerical Simulation // CMCcomputers materials & continua. - 2022. - Т. 73, № 3. - С. 4627-4640.

33 Amirgaliyev Y., Wójcik W., Kunelbayev M., Merembayev, T. Theoretical prerequisites of electric water heating // NEWS of National Academy of Sciences of the Republic of Kazakhstan. - 2019. - Vol. 6. - С. 54–63.

34 Hao W., Lu Y., Lai Y., Yu H., Lyu M. Research on operation strategy and performance prediction of flat plate solar collector with dual-function for drying agricultural products // Renewable Energy. - 2018. - Т. 127. - С. 685696.

35 Jafarkazemi F., Ahmadifard E. Energetic and exergetic evaluation of flat plate solar collectors // Renewable energy. - 2013. - Т. 56. - С. 55-63.

36 Hollands K. G. T., Unny T. E., Raithby G. D., Konicek, L. Free convective heat transfer across inclined air layers. - 1976.- С. 189–193.

37 Tabor H. Radiation, convection and conduction coefficients in solar collectors // Bull. Res. Counc. Isr., Sect. C;(Israel). - 1958. - Т. 6.

38 Amirgaliyev Y. Solar-Driven Resources of the Republic of Kazakhstan, News of the National Academy of Sciences of the Republic of Kazakhstan // Series of Geology and Technical Sciences. - 2018. - Т. 3. - С. 430.

39 Yedilkhan A., Merembayev T., Kunelbayev M. Dynamic simulation of a solar hot water heating system for Kazakhstan climate conditions // 2018 14th International Conference on Electronics Computer and Computation (ICECCO). - IEEE. 2018. - С. 206-212.

40 Faizan V., Galib M., Khan A., Afzal S. Arduino web server for efficiently control and monitoring of solar power system // Int. J. Comput. Eng. Appl. - 2014. - Т. 8, № 1. - С. 148-157.

41 Salamone F., Belussi L., Danza L., Ghellere M., Meroni, I. An open source low-cost wireless control system for a forced circulation solar plant // Sensors. - 2015. - Т. 15, № 11. - С. 27990-28004.

42 Yu Q. S., Duan M. Y., Zhang T. S., Wu H. G., Lu S. K. An wireless collection and monitoring system design based on Arduino // Advanced Materials Research. Т. 971. - Trans Tech Publ. 2014. - С. 1076-1080.

43 Gad H., Gad H. E. Development of a new temperature data acquisition system for solar energy applications // Renewable energy. - 2015. - Т. 74 - С. 337-343.

44 Luijten H. How to measure temperature with your Arduino and a DS18B20. - 2015. - URL: https://www.tweaking4all.com/hardware/arduino/arduinods18b20-temperature-sensor/. 20.05.2022.

45 Vargas-Salgado C., Aguila-Leon J., Chiñas-Palacios C., Hurtado-Perez E. Lowcost web-based Supervisory Control and Data Acquisition system for a microgrid testbed: A case study in design and implementation for academic and research applications // Heliyon. - 2019. - Т. 5, № 9.

46 Амиргалиев Е. Н., Кунелбаев М., Мерембаев Т., Козбакова А. Х., Сундетов Т. Р., Иржанова, А. А. Система управления контроллерами плоского солнечного коллектора с термосифонной циркуляцией // Вестник Казахстанско-Британского технического университета. – 2021.- Т. 16, № 1, С.55-60.

47 Ruiz A. P., Flynn M., Large J., Middlehurst M., Bagnall A. The great multivariate time series classification bake off: a review andexperimental evaluation of recent algorithmic advances // Data Mining and Knowledge Discovery. - 2021. - T. 35, № 2. - C. 401-449.

48 Piccolo D. A distance measure for classifying ARIMA models // Journal of time series analysis. - 1990. - T. 11, № 2. - C. 153-164.

49 Gilbert K. An ARIMA supply chain model // Management Science. - 2005. - T. 51, № 2. - C. 305-310.

50 Nise N. S. Control systems engineering. - John Wiley & Sons, 2020.

51 Aliev F., Larin V. B. Optimization of linear control systems: analytical methods and computational algorithms. - CRC Press, 1998.

52 Howard R. A. Dynamic programming and markov processes. - 1960.

53 Kijima M. Markov processes for stochastic modeling. - Springer, 2013.

54 Kalyan A., Mohta A., Polozov O., Batra D., Jain P., Gulwani S. Neural-guided deductive search for real-time program synthesis from examples // arXiv preprint arXiv:1804.01186. - 2018.

55 Harvey A. C. Forecasting, structural time series models and the Kalman filter. 1990.

56 Graves A. Generating sequences with recurrent neural networks // arXiv preprint arXiv:1308.0850. - 2013.

57 Irsoy O., Cardie C. Opinion mining with deep recurrent neural networks // Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP). - 2014. - C. 720-728.

58 Chung J., Gulcehre C., Cho K., Bengio Y. Empirical evaluation of gated recurrent neural networks on sequence modeling // arXiv preprint arXiv:1412.3555. - 2014.

59 Cho K., Van Merriënboer B., Bahdanau D., Bengio Y. On the properties of neural machine translation: Encoder-decoder approaches // arXiv preprint arXiv:1409.1259. - 2014.

60 Understanding LSTM Networks. - 2015. - URL: http://colah.github.io/posts/2015-08-Understanding-LSTMs. 27.08.2022.

61 Merembayev T., Yunussov R., Yedilkhan A. Machine learning algorithms for classification geology data from well logging // 2018 14th International Conference on Electronics Computer and Computation (ICECCO). - IEEE. 2018. - C. 206-212.

62 Merembayev T., Yunussov R., Yedilkhan A. Machine learning algorithms for stratigraphy classification on uranium deposits // Procedia Computer Science. 2019. - T. 150. - C. 46-52.

63 Merembayev T., Kurmangaliyev D., Bekbauov B., Amanbek Y. A Comparison of machine learning algorithms in predicting lithofacies: Case studies from Norway and Kazakhstan // Energies. - 2021. - T. 14, № 7. - C. 1896.

64 Khandelwal U., He H., Qi P., Jurafsky D. Sharp nearby, fuzzy far away: How neural language models use context // arXiv preprint arXiv:1805.04623. - 2018.

65 Taylor S. J., Letham B. Forecasting at scale // The American Statistician. - 2018. - T. 72, № 1. - C. 37-45.

66 Ke G., Meng Q., Finley T., Wang T., Chen W., Ma W., Liu T. Y. Lightgbm: A highly efficient gradient boosting decision tree // Advances in neural information processing systems. - 2017. - Т. 30.

67 Chen T., He T., Benesty M., Khotilovich V., Tang Y., Cho H., Chen K. Xgboost: extreme gradient boosting // R package version 0.4-2. - 2015. - Т. 1, № 4. - С. 1-4.

68 Dorogush A. V., Ershov V., Gulin A. CatBoost: gradient boosting with categorical features support // arXiv preprint arXiv:1810.11363. - 2018.

69 Chen R. T., Rubanova Y., Bettencourt J., Duvenaud, D. Neural ordinary differential equations / R. T. Chen [и др.] // Advances in neural information processing systems. - 2018. - Т. 31.

70 Tzen B., Raginsky M. Neural stochastic differential equations: Deep latent gaussian models in the diffusion limit // arXiv preprint arXiv:1905.09883. - 2019.

71 Kidger P., Morrill J., Foster J., Lyons, T. Neural controlled differential equations for irregular time series // Advances in Neural Information Processing Systems. - 2020. - Т. 33. - С. 6696-6707.

72 Yedilkhan A., Murat K., Beibut A., Aliya K., Ainur K., Timur M., Azhibek D. Mathematical justification of thermosyphon effect main parameters for solar heating system // Cogent Engineering. - 2020. - Т. 7, № 1. - С. 1851629.

73 Арнольд В. Обыкновенные дифференциальные уравнения. - Litres, 2017.

74 Griewank A. Achieving logarithmic growth of temporal and spatial complexity in reverse automatic differentiation // Optimization Methods and software. - 1992. - Т. 1, № 1. - С. 35-54.

75 Gholami A., Keutzer K., Biros G. Anode: Unconditionally accurate memoryefficient gradients for neural odes // arXiv preprint arXiv:1902.10298. - 2019.

76 Hager W. W. Runge-Kutta methods in optimal control and the transformed adjoint system // Numerische Mathematik. - 2000. - Т. 87, № 2. - С. 247-282.

77 Понтрягин Л. С., Болтянский В. Г., Гамкрелидзе Р. В., Мищенко Е. Ф. Математическая теория оптимальных процессов - М.: Наука, 1969. - 408с.

78 Bastien F., Lamblin P., Pascanu R., Bergstra J., Goodfellow I., Bergeron A., Bengio Y. Theano: new features and speed improvements // arXiv preprint arXiv:1211.5590. - 2012.

79 Abadi M., Agarwal A., Barham P., Brevdo E., Chen Z., Citro C., Zheng X. Tensorflow: Large-scale machine learning on heterogeneous distributed systems // arXiv preprint arXiv:1603.04467. - 2016.

80 Paszke A., Gross S., Chintala S., Chanan G., Yang E., DeVito Z., Lerer A. Automatic differentiation in pytorch. - 2017.

81 Alvarez M. A., Lawrence N. D. Computationally efficient convolved multiple output Gaussian processes // The Journal of Machine Learning Research. - 2011. - Т. 12. - С. 1459-1500.

82 Futoma J., Hariharan S., Heller K. Learning to detect sepsis with a multitask Gaussian process RNN classifier // International conference on machine learning. - PMLR. 2017. - С. 1174-1182.

83 Gelman A., Hill J. Data analysis using regression multilevel/hierarchical models. - Cambridge university press, 2006.

84 Li Y., Du N., Bengio S. Time-dependent representation for neural event sequence prediction // arXiv preprint arXiv:1708.00065. - 2017.

85 Lipton Z. C., Kale D., Wetzel R. Directly modeling missing data in sequences with rnns: Improved classification of clinical time series // Machine learning for healthcare conference. - PMLR. 2016. - С. 253-270.

86 Raissi M., Perdikaris P., Karniadakis G. E. Multistep neural networks for datadriven discovery of nonlinear dynamical systems // arXiv preprint arXiv:1801.01236. - 2018.

87 Long Z., Lu Y., Ma X., Dong B. Pde-net: Learning pdes from data // International Conference on Machine Learning. - PMLR. 2018. - С. 32083216.

88 Ait-Amir B., Pougnet P., El Hami A. Meta-model development // Embedded mechatronic systems 2. - Elsevier, 2020. - С. 157-187.

89 Schneider P., Xhafa F. Anomaly Detection and Complex Event Processing Over IoT Data Streams: With Application to EHealth and Patient Data Monitoring. - Academic Press, 2022.

90 Hochreiter S., Schmidhuber J. LSTM can solve hard long time lag problems // Advances in neural information processing systems. - 1996. - Т. 9.

91 Merembayev T., Amirgaliyev Y. Anomaly detection in solar hot water system using machine learning // Bulletin of the National Engineering Academy of the Republic of Kazakhstan. - 2021. - Vol. 3. - P. 34–44.

92 Kingma D. P., Ba J. Adam: A method for stochastic optimization // arXiv preprint arXiv:1412.6980. - 2014.

93 Bache T. C., Bratt S. R., Wang J., Fung R. M., Kobryn C., Given J. W. The intelligent monitoring system / T. C. Bache [и др.] // Bulletin of the Seismological Society of America. - 1990. - Т. 80, 6B. - С. 1833-1851.

94 Sixsmith A. An evaluation of an intelligent home monitoring system // Journal of telemedicine and telecare. - 2000. - Т. 6, № 2. - С. 63-72.

95 Athanasiadis I. N., Mitkas P. A. An agent-based intelligent environmental monitoring system // Management of Environmental Quality: An International Journal. - 2004.

96 Chui M., Collins M., Patel M. The Internet of Things: Catching up to an accelerating opportunity, McKinsey Global Institute, November 2021.

97 Gandomi A., Haider M. Beyond the hype: Big data concepts, methods, and analytics // International journal of information management. - 2015. - Т. 35, № 2. - С. 137-144.

98 Амиргалиев Е., Мерембаев Т. Проектирование архитектуры хранения данных в сети гелиоколлекторов // Вестник КазНИТУ, Казахский национальный исследовательский технический университет имени К.И.Сатпаева. - 2020. - Т. 4. - С. 212-216.

99 Shi W., Cao J., Zhang Q., Li Y., Xu, L. Edge computing: Vision and challenges // IEEE internet of things journal. - 2016. - Т. 3, № 5. - С. 637-646.

100 Wong B. P., Kerkez B. Real-time environmental sensor data: An application to water quality using web services // Environmental Modelling & Software. - 2016. - T. 84. - C. 505-517.

101 Granell C., Havlik D., Schade S., Sabeur Z., Delaney C., Pielorz J., Mon J. L. Future Internet technologies for environmental applications // Environmental Modelling & Software. - 2016. - T. 78. - C. 1-15.

102 Laniak G. F., Olchin G., Goodall J., Voinov A., Hill M., Glynn P., Hughes A.Integrated environmental modeling: a vision and roadmap for the future // Environmental Modelling & Software. - 2013. - T. 39. - C. 3-23.

ПРИЛОЖЕНИЕ А

Автоматическое дифференцирования

Пусть $f_1,...,f_n$ - некоторый набор функций, производные которых можно вычислять. Тогда для любой композиции этих функций $x \mapsto\rightarrow f(x) = f_{im}(...(f_{i1}(x))...)$, причем $i_1,...,i_m \in 1,...,n$. Производная вычисляется по цепочке в графе:

$$\frac{df}{dx} = \frac{df_{i_m}}{df_{i_{m-1}}} \cdots \frac{df_{i_2}}{df_{i_1}} \frac{df_{i_1}}{dx} \tag{A.1}$$

В общем виде алгоритм такой: предположим, что нам задан некоторый направленный ациклический граф вычислений $G = (V,E)$, вершинами которого являются функции $g \in V$, причем часть вершин соответствует входным переменным $x_1,...,x_n$ и не имеет входящих ребер, одна вершина не имеет исходящих ребер и соответствует функции f (весь граф вычисляет эту функцию), а ребра показывают зависимости между функциями, стоящими в узлах. Тогда мы уже знаем, как получить функцию f, стоящую в «последней» вершине графа: для этого достаточно двигаться по ребрам и вычислять каждую функцию в топологическом порядке.

Есть два основных подхода для расчета градиентов в графе. Первым является алгоритм прямого распростронения ошибки. Автоматическое дифференцирование в данном случае, также известное как чувствительность пересылки, происходит путем рекурсивного вычисления.

Второй подход это алгоритм обратного распростронения ошибки или обратная чувствительность, потому что частные производные считаются в направлении, обратном ребрам графа вычислений.

Библиотеки PyTorch и TensorFlow позволяют задать граф вычислений эффективно, с распараллеливанием и переносом на видеокарты, вычисляют градиент по этому графу.

ПРИЛОЖЕНИЕ Б

Примеры реализации нейросетевых ОДУ
Листинг Б.1 Python code:

```
import argparse
import os
import matplotlib
import numpy as np
import numpy.random as npr
matplotlib.use('agg')
import matplotlib.pyplot as plt
import tensorflow as tf

tf.enable_eager_execution()
tf.keras.backend.set_floatx('float64')

parser = argparse.ArgumentParser()
parser.add_argument('--adjoint', type=eval, default=False)
parser.add_argument('--visualize', type=eval, default=True)
parser.add_argument('--niters', type=int, default=2000)
parser.add_argument('--lr', type=float, default=0.01)
parser.add_argument('--gpu', type=int, default=0)
parser.add_argument('--train_dir', type=str, default='latent')
args = parser.parse_args()

from tfdiffeq import odeint, move_to_device

class LatentODEfunc(tf.keras.Model):

    def __init__(self, latent_dim=4, nhidden=20):
        super(LatentODEfunc, self).__init__()
        self.fc1 = tf.keras.layers.Dense(nhidden, activation='elu')
        self.fc2 = tf.keras.layers.Dense(nhidden, activation='elu')
        self.fc3 = tf.keras.layers.Dense(latent_dim)
        self.nfe = 0

    def call(self, t, x):
        self.nfe += 1

        out = self.fc1(x)
        out = self.fc2(out)
        out = self.fc3(out)
```

```
        return out


class RecognitionRNN(tf.keras.Model):

    def __init__(self, latent_dim=4, obs_dim=2, nhidden=25, nbatch=1):
        super(RecognitionRNN, self).__init__()
        self.nhidden = nhidden
        self.nbatch = nbatch
        self.i2h = tf.keras.layers.Dense(nhidden, activation='tanh')
        self.h2o = tf.keras.layers.Dense(latent_dim * 2)

    def call(self, x, h):
        combined = tf.concat((x, h), axis=1)
        h = self.i2h(combined)
        out = self.h2o(h)
        return out, h

    def initHidden(self):
        return tf.zeros([self.nbatch, self.nhidden], dtype=tf.float64)


class Decoder(tf.keras.Model):

    def __init__(self, latent_dim=4, obs_dim=2, nhidden=20):
        super(Decoder, self).__init__()
        self.fc1 = tf.keras.layers.Dense(nhidden, activation='relu')
        self.fc2 = tf.keras.layers.Dense(obs_dim)

    def call(self, z):
        out = self.fc1(z)
        out = self.fc2(out)
        return out


class RunningAverageMeter(object):

    def __init__(self, momentum=0.99):
        self.momentum = momentum
        self.reset()

    def reset(self):
        self.val = None
        self.avg = 0
```

```
    def update(self, val):
        if self.val is None:
            self.avg = val
        else:
            self.avg = self.avg * self.momentum + val * (1 - self.momentum)
        self.val = val


def log_normal_pdf(x, mean, logvar):
    const = tf.convert_to_tensor(np.array([2. * np.pi]), dtype=tf.float64)
    const = move_to_device(const, device)
    const = tf.log(const)
    return -.5 * (const + logvar + (x - mean) ** 2. / tf.exp(logvar))


def normal_kl(mu1, lv1, mu2, lv2):
    v1 = tf.exp(lv1)
    v2 = tf.exp(lv2)
    lstd1 = lv1 / 2.
    lstd2 = lv2 / 2.

    kl = lstd2 - lstd1 + ((v1 + (mu1 - mu2) ** 2.) / (2. * v2)) - .5
    return kl


def save_states(orig_ts, orig_trajs, samp_ts, samp_trajs):
    ots = orig_ts.numpy()
    otjs = orig_trajs.numpy()
    sts = samp_ts.numpy()
    stjs = samp_trajs.numpy()

    orig_ts_path = os.path.join(args.train_dir, 'orig_ts')
    orig_trajs_path = os.path.join(args.train_dir, 'orig_trajs')
    samp_ts_path = os.path.join(args.train_dir, 'samp_ts')
    samp_trajs_path = os.path.join(args.train_dir, 'samp_trajs')

    np.save(orig_ts_path, ots)
    np.save(orig_trajs_path, otjs)
    np.save(samp_ts_path, sts)
    np.save(samp_trajs_path, stjs)


def restore_states():
```

```
    orig_ts_path = os.path.join(args.train_dir, 'orig_ts.npy')
    orig_trajs_path = os.path.join(args.train_dir, 'orig_trajs.npy')
    samp_ts_path = os.path.join(args.train_dir, 'samp_ts.npy')
    samp_trajs_path = os.path.join(args.train_dir, 'samp_trajs.npy')

    ots = tf.convert_to_tensor(np.load(orig_ts_path), dtype=tf.float64)
    otjs = tf.convert_to_tensor(np.load(orig_trajs_path), dtype=tf.float32)
    sts = tf.convert_to_tensor(np.load(samp_ts_path), dtype=tf.float32)
    stjs = tf.convert_to_tensor(np.load(samp_trajs_path), dtype=tf.float32)

    states = dict(orig_ts=ots, orig_trajs=otjs,
                  samp_ts=sts, samp_trajs=stjs)

    return states


if __name__ == '__main__':
    latent_dim = 4
    nhidden = 20
    rnn_nhidden = 25
    obs_dim = 2
    nspiral = 1000
    start = 0.
    stop = 6 * np.pi
    noise_std = .3
    a = 0.
    b = .3
    ntotal = 1000
    nsample = 100
    device = 'gpu:' + str(args.gpu) if tf.test.is_gpu_available() else 'cpu'

    with tf.device(device):
        orig_trajs, samp_trajs, orig_ts, samp_ts = generate_spiral2d(
            nspiral=nspiral,
            start=start,
            stop=stop,
            noise_std=noise_std,
            a=a, b=b
        )

        orig_ts = tf.convert_to_tensor(orig_ts, dtype=tf.float64)
        orig_trajs = tf.convert_to_tensor(orig_trajs, dtype=tf.float64)
        samp_trajs = tf.convert_to_tensor(samp_trajs, dtype=tf.float64)
        samp_ts = tf.convert_to_tensor(samp_ts, dtype=tf.float64)
```

```
    func = LatentODEfunc(latent_dim, nhidden)
    rec = RecognitionRNN(latent_dim, obs_dim, rnn_nhidden, nspiral)
    dec = Decoder(latent_dim, obs_dim, nhidden)
    optimizer = tf.train.AdamOptimizer(args.lr)
    loss_meter = RunningAverageMeter()

    saver = tf.train.Checkpoint(func=func, rec=rec, dec=dec, optimizer=optimizer)

    if args.train_dir is not None:
        if not os.path.exists(args.train_dir):
            os.makedirs(args.train_dir)
        else:
            if tf.train.checkpoint_exists(args.train_dir):
                path = tf.train.latest_checkpoint(args.train_dir)

                if path is not None:
                    saver.restore(path)

                    states = restore_states()
                    orig_trajs = states['orig_trajs']
                    samp_trajs = states['samp_trajs']
                    orig_ts = states['orig_ts']
                    samp_ts = states['samp_ts']
                    print('Loaded ckpt from {}'.format(path))

    for itr in range(1, args.niters + 1):
        with tf.GradientTape() as tape:
            h = rec.initHidden()
            for t in reversed(range(samp_trajs.shape[1])):
                obs = samp_trajs[:, t, :]
                out, h = rec(obs, h)
            qz0_mean, qz0_logvar = out[:, :latent_dim], out[:, latent_dim:]
            epsilon                                                                    =
tf.convert_to_tensor(np.random.randn(*qz0_mean.shape.as_list()),
dtype=qz0_mean.dtype)
            z0 = epsilon * tf.exp(.5 * qz0_logvar) + qz0_mean

            pred_z = tf.transpose(odeint(func, z0, samp_ts), [1, 0, 2])
            pred_x = dec(pred_z)

            # compute loss
            noise_std_ = tf.zeros(pred_x.shape, dtype=tf.float64) + noise_std
            noise_logvar = 2. * tf.log(noise_std_)
```

```
            logpx = tf.reduce_sum(log_normal_pdf(
                samp_trajs, pred_x, noise_logvar), axis=-1)
            logpx = tf.reduce_sum(logpx, axis=-1)
            pz0_mean = pz0_logvar = tf.zeros(z0.shape, dtype=tf.float64)
            analytic_kl = tf.reduce_sum(normal_kl(qz0_mean, qz0_logvar,
                                        pz0_mean, pz0_logvar), axis=-1)
            loss = tf.reduce_mean(-logpx + analytic_kl, axis=0)

        params = (list(func.variables) + list(dec.variables) + list(rec.variables))
        grad = tape.gradient(loss, params)
        grad_vars = zip(grad, params)

        optimizer.apply_gradients(grad_vars)
        loss_meter.update(loss.numpy())

        print('Iter: {}, running avg elbo: {:.4f}'.format(itr, -loss_meter.avg))

        if itr != 0 and (itr + 1) % 100 == 0:
            if args.train_dir is not None:
                ckpt_path = os.path.join(args.train_dir, 'ckpt')

                saver.save(ckpt_path)
                save_states(orig_ts, orig_trajs, samp_ts, samp_trajs)
                print('Stored ckpt at {}'.format(ckpt_path))

    print('Training complete after {} iters.'.format(itr))

    if args.visualize:
        h = rec.initHidden()
        for t in reversed(range(samp_trajs.shape[1])):
            obs = samp_trajs[:, t, :]
            out, h = rec(obs, h)
        qz0_mean, qz0_logvar = out[:, :latent_dim], out[:, latent_dim:]
        epsilon = tf.convert_to_tensor(np.random.randn(*qz0_mean.shape.as_list()),
dtype=tf.float64)
        z0 = epsilon * tf.exp(.5 * qz0_logvar) + qz0_mean
        orig_ts = tf.convert_to_tensor(orig_ts, dtype=tf.float32)

        z0 = z0[0:1]

        ts_pos = np.linspace(0., 2. * np.pi, num=2000)
        ts_neg = np.linspace(-np.pi, 0., num=2000)[::-1].copy()
        ts_pos = tf.convert_to_tensor(ts_pos, dtype=tf.float32)
        ts_neg = tf.convert_to_tensor(ts_neg, dtype=tf.float32)
```

```
zs_pos = odeint(func, z0, ts_pos)
zs_neg = odeint(func, z0, ts_neg)

xs_pos = dec(zs_pos)
xs_neg = tf.reverse(dec(zs_neg), axis=[0])

xs_pos = xs_pos.numpy().squeeze(1)
xs_neg = xs_neg.numpy().squeeze(1)
orig_traj = orig_trajs[0].numpy()
samp_traj = samp_trajs[0].numpy()

plt.figure()
plt.plot(orig_traj[:, 0], orig_traj[:, 1],
        'g', label='true trajectory')
plt.plot(xs_pos[:, 0], xs_pos[:, 1], 'r',
        label='learned trajectory (t>0)')
plt.plot(xs_neg[:, 0], xs_neg[:, 1], 'c',
        label='learned trajectory (t<0)')
plt.scatter(samp_traj[:, 0], samp_traj[
          :, 1], label='sampled data', s=3)
plt.legend()
```

ПРИЛОЖЕНИЕ В

Патент на полезную модель №5591

ҚАЗАҚСТАН РЕСПУБЛИКАСЫ РЕСПУБЛИКА КАЗАХСТАН

REPUBLIC OF KAZAKHSTAN

ПАТЕНТ
PATENT

№ 5591

ПАЙДАЛЫ МОДЕЛЬГЕ / НА ПОЛЕЗНУЮ МОДЕЛЬ / FOR UTILITY MODEL

(21) 2020/0650.2

(22) 13.07.2020

(45) 20.11.2020

(54) Күн коллекторларын қашықтан бақылау жүйесі
Система дистанционного мониторинга солнечных коллекторов
Remote monitoring system for solar collectors

(73) Амиргалиев Едилхан Несипханович (KZ)
Amirgaliyev Yedilkhan Nesipkhanovich (KZ)

(72) Кунелбаев Мурат Меркебекович (KZ)
Kunelbayev Murat Merkebekovich (KZ)
Калимолдаев Максат Нурадилович (KZ)
Kalimoldayev Maksat Nuradilovich (KZ)
Сундетов Талғат Рысбекұлы (KZ)
Sundetov Talgat Rysbekuly (KZ)
Даулбаев Салауат Муратович (KZ)
Daulbayev Salauat Muratovich (KZ)
Мерембаев Тимур Жумаканович (KZ)
Merembayev Timur Zhumakanovich (KZ)

ЭЦҚ қол қойылды
Подписано ЭЦП
Signed with EDS

Е. Оспанов
Е. Оспанов
Y. Ospanov

«Ұлттық зияткерлік меншік институты» РМК директоры
Директор РГП «Национальный институт интеллектуальной собственности»
Director of the «National Institute of Intellectual Property» RSE

ПРИЛОЖЕНИЕ Г

Техническое развертывание дашборда

Разработанное приложение Vue.js состоит из 3 основных частей:

1. База данных PostgreSQL. В базе данных хранятся таблицы с необходимыми данными. Все данные представленные на страницах приложения представлены в базе данных.
2. Веб-сервер PostgREST. Позволяет работать с базой данных без разработки отдельного сервера.
3. Веб-приложение. Приложение написано на Vue.js.
   Развертывание системы требует следующих шагов. В случае ручной установки требуется выполнить следующие шаги:
   a) Установить PostgreSQL:
   b) Скачать PostgreSQL https://www.postgresql.org/download/
   c) Подробная документация PostgreSQL https://www.postgresql.org/docs/.
   d) Установить PostgREST:
   e) Скачать и установить PostgREST https://postgrest.org/en/stable/install.html.
   f) Руководство по первоначальной установке https://postgrest.org/en/stable/tutorials/tut0.html.
   g) Для работы PostgREST необходим файл конфигурации в корневой папке. Пример, конфигурации файла postgrest.conf: Стандартное подключение по URI. Документация

Листинг Г.1 bash | version
https://www.postgresql.org/docs/current/ libpq-connect.html
LIBPQ- | CONNSTRING db-uri = "postgres://user:pass@host:5432/dbname"

Название базы данных для REST клиентов db-schemas = "api"Роль в базе данных используемая без аутентификации. Должно отличаться от пользователя в базе данных db-uri db-anon-role = "anon"

(b) Запустить PostgREST (через терминал): ./postgrest postgrest.conf
(c) Запустить PostgREST (через терминал): ./postgrest postgrest.conf
(d) Клонировать репозиторий веб-приложение (через терминал): git clone https://github.com/aidynb/biogas-vue.git
(e) Перейти в директорию (через терминал): cd biogas-vue

(f) Установить зависимости (через терминал): npm i

(g) После успешной установки запустить приложение (через терминал): yarn serve

(h) Перейти на http://localhost:3001/

Установка с помощью Docker. Требуется запустить Postgrest в отдельном Docker контейнере:

Листинг Г.2 bash version sudo docker run --rm --net=host -p 3000:3000

Листинг Г.3 bash version

```
-e PGRST_DB_URI="postgres://authenticator:biogas@localhost
   :5432/postgres"
-e PGRST_DB_SCHEMA="api"
-e PGRST_DB_ANON_ROLE="biogas_read" postgrest/postgrest
```

Данная команда загружает образ postgrest/postgrest и запускает контейнер Docker. Опции:

1. –rm – удаляет контейнер после остановки.
2. –net=host – обращение к localhost внутри приложения будут перенаправляться к localhost хоста.
3. -p 3000:3000 – открывает порт 3000 для приложения на хосте.

Листинг Г.4 bash version

```
4.-e PGRST_DB_URI="postgres://authenticator:
                    biogas@localhost:5432/postgres"
```

– указывает строку подключения к PostgreSQL.

Листинг Г.5 bash version

```
5.  -e PGRST_DB_SCHEMA="api"
```

– указывает схему внутри PostgreSQL.

Листинг Г.6 bash version

```
6.  -e PGRST_DB_ANON_ROLE="biogas_read"
```

– указывает роль PostgREST для считывания данных из PostgreSQL.

Запустить Веб-приложение в отдельном Docker контейнере:

Листинг Г.7 bash version

```
sudo docker build -t biogas-vue.
sudo docker run -it --name biogas-vue -p 8000:8000 biogas-vue -t biogas-vue
5it (--interactive, --tty)
```

–оставляет открытым STDIN, и аллоцирует псевдо-tty –name biogas-vue – устанавливает имя контейнера. -p 8000:8000 – открывает порт 8000 на хосте для приложения.

Dockerfile веб-приложения:

Dockerfile состоит из нескольких слоев, каждый из которых выполняет свою определенную функцию: FROM ubuntu:latest

1. обозначает на чем основан образ.

Листинг Г.8 bash version

```
RUN apt-get update && apt-get upgrade -y && apt-get install -y
  nodejs npm
```

2. Обновляет внутренние пакеты и устанавливает дополнительные для корректной работы приложения.

Листинг Г.9 bash version

```
RUN apt remove cmdtest
RUN npm install -g yarn
```

3. Установка пакетного менеджера yarn. # делаем каталог ’app’ текущим рабочим каталогом WORKDIR /app
4. Устанавливает текущий рабочий каталог # копируем оба ’package.json’ и ’package-lock.json’ (если есть) COPY package*.json ./
5. Копирует файлы package.json и package-lock.json с описанием существующих зависимостей приложения. # устанавливаем зависимости проекта RUN yarn
6. Устанавливает все необхожимые зависимости. # копируем файлы и каталоги проекта в текущий рабочий каталог (т.е. в каталог ’app’) COPY.

Копирует все файлы и директории приложения. EXPOSE 8000 CMD ["yarn "serve"] Открывает порт: 8000 для внешнего мира и указывает команду запуска приложения.