

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Факультет

вычислительной техники

Кафедра

«Вычислительная техника»

Направление подготовки: 09.04.01 «Информатика и вычислительная техника»

Магистерская программа: «Прикладной искусственный интеллект»

программа:

МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ

на тему

РАЗРАБОТКА МЕТОДА И СРЕДСТВ
ПРОГНОЗИРОВАНИЯ РАСХОДА
ЭНЕРГОНОСИТЕЛЯ И ОПЛАТЫ
КОММУНАЛЬНЫХ УСЛУГ

Студент

(подпись, дата)

Попов Антон Фёдорович

(ФИО полностью)

Руководитель

(подпись, дата)

Финогеев А.Г.

(фамилия, инициалы)

Нормоконтролёр

(подпись, дата)

Бычков А.С.

(фамилия, инициалы)

Рецензент

(должность, место работы)

(подпись, дата)

(фамилия, инициалы)

Работа допущена к защите (протокол зас. кафедры от 28.05.25 № 16)

Заведующий кафедрой

(подпись)

М.А. Митрохин

(фамилия, инициалы)

Работа защищена с отметкой отлично (протокол заседания ГЭК от

25 июня 2025 № 16)

Секретарь ГЭК

(подпись)

Е.И. Гурин

(фамилия, инициалы)

Пенза 2025

Федеральное государственное бюджетное образовательное учреждение
высшего образования
«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Факультет

вычислительной техники

Кафедра

«Вычислительная техника»

«Утверждаю»

Заведующий кафедрой

М.А. Митрохин

« 29 » 09 2023 г.

ЗАДАНИЕ НА МАГИСТЕРСКУЮ ДИССЕРТАЦИЮ

1. Студент __Попов Антон Фёдорович гр. 23ВВИМ1 факультета вычислительной техники
2. Тема работы «Разработка метода и средств прогнозирования расхода энергоносителя и оплаты коммунальных услуг»

Тема утверждена приказом ПГУ № 986/о от 29 сентября 2023 г.

3. Руководитель магистерской диссертации д.т.н., профессор кафедры САПР Финогеев А.Г.

4. Задание на работу (назначение разработки, исходные данные и т.п.):

Назначение разработки: Разработка методов и моделей, позволяющих спрогнозировать потребление энергоресурсов исходя из истории показаний собственников.

Входные данные: Перечень лицевого счетов с данными по ежемесячному потреблению коммунальных ресурсов (холодной воды, электроэнергия и т.д.)

Результат: Система прогнозирования расхода коммунальных ресурсов.

5. Перечень подлежащих разработке вопросов:

Исследование существующих методов прогнозирования и архитектур нейронных сетей;

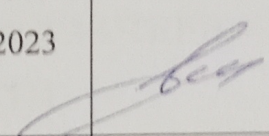
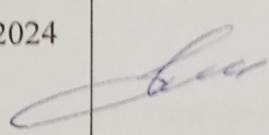
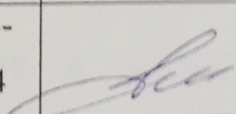
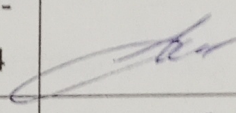
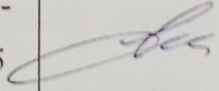
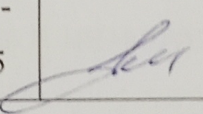
Определение подходящей архитектуры искусственных нейронных сетей для применения в реализации средств прогнозирования;

Разработка средств прогнозирования расхода коммунальных ресурсов;

Разработка системы прогнозирования расхода коммунальных ресурсов;

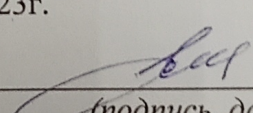
Тестирование прогнозирования расхода коммунальных ресурсов;

6. Календарный график выполнения работы

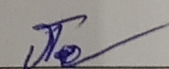
№ п/п	Наименование этапов работы	Объем работы	Срок выполнения	Подпись руководителя
1	Исследование существующих методов прогнозирования и архитектур нейронных сетей	10%	29.09.2023- 01.12.2023	
2	Определение подходящей архитектуры искусственных нейронных сетей для применения в реализации средств прогнозирования	13%	02.12.2023- 05.02.2024	
3	Разработка средств прогнозирования расхода коммунальных ресурсов	30%	06.02.2024- 05.08.2024	
4	Разработка системы прогнозирования расхода коммунальных ресурсов	22%	06.08.2024- 10.11.2024	
5	Тестирование прогнозирования расхода коммунальных ресурсов	10%	11.11.2024- 10.03.2025	
6	Оформление магистерской диссертации	15%	10.03.2025- 25.05.2025	

Дата выдачи задания " 29 " сентября 2023г.

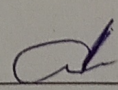
Руководитель магистерской диссертации

 24.09.23
(подпись, дата)

Задание к исполнению принял студент

 19.05.25
(подпись, дата)

Работу к защите допустить
Декан факультета

 25.06.25
(подпись, дата)

РЕФЕРАТ

Магистерская работа содержит 61 страницу, 41 рисунок, 4 таблицы, 12 источников, 3 приложения.

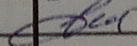
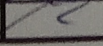
ПРОГНОЗИРОВАНИЕ РАСХОДА КОММУНАЛЬНЫХ РЕСУРСОВ,
МЕТОДЫ И СРЕДСТВА ПРОГНОЗИРОВАНИЯ, ИСКУССТВЕННЫЕ
НЕЙРОННЫЕ СЕТИ, РЕКУРРЕНТНЫЕ НЕЙРОННЫЕ СЕТИ

Объектом исследования является прогнозирования расхода коммунальных ресурсов.

Предметом исследования являются методы и средства прогнозирования расхода коммунальных ресурсов.

Цель данной работы состояла в анализе моделей и методов прогнозирования, а также разработке средств прогнозирования расхода коммунальных ресурсов, в частности – расхода ресурса на отопление и воду.

В результате проделанной работы была разработана система прогнозирования расхода коммунальных ресурсов в соответствии с выбираемым лицевым счётом, типом прогнозируемого ресурса и интервалом прогноза.

					ПГУ1.090401.13.001 ПЗ		
Изм.	Лист	№ докум.	Подпись	Дата			
Разраб.		Попов А. Ф.		19.05.23	Разработка метода и средств прогнозирования расхода энергоносителя и оплаты коммунальных услуг Пояснительная записка		
Провер.		Финогеев А. Г.		19.06.23			
Н. Контр.		Бычков А. С.		20.06.23	ФВТ, каф. ВТ гр.23ВВИм1		
Утв.		Митрохин М. А.		20.06.23			

СОДЕРЖАНИЕ

ПЕРЕЧЕНЬ СОКРАЩЕНИЙ И ОБОЗНАЧЕНИЙ	5
ВВЕДЕНИЕ.....	6
1 Понятия сферы ЖКХ	8
1.1 Жилищно-коммунальное хозяйство	8
1.2 Многоквартирный дом	8
1.3 Индивидуальный прибор учёта	9
1.4 Общедомовой прибор учёта	11
2 Обзор существующих методов прогнозирования	13
2.1 Упрощенные методы прогнозирования	13
2.2 Прогнозирование на основе трендовых моделей	14
2.3 Прогнозирование с использованием сезонной компоненты	15
2.4 Регрессионный анализ.....	16
2.5 Методы экспертных оценок.....	17
2.6 Комплексные методы прогнозирования.....	18
2.7 Вывод	18
3 Анализ архитектур нейронных сетей	19
3.1 Многослойный перцептрон	19
3.2 Сверточные нейронные сети	20
3.3 Рекуррентные нейронные сети	21
3.4 Долгая краткосрочная память	22
3.5 Трансформер	23
4 Назначение разработки.....	26
4.1 Функциональное назначение	26
4.2 Эксплуатационное назначение	26
4.3 Требования к программе	26
4.3.1 Требования к функциональным характеристикам	26
4.3.2 Требования к организации входных данных	27
4.3.3 Требования к организации выходных данных.....	27
4.3.4 Макет интерфейса приложения	27
4.3.5 Основные требования к данным, на основе которых формируется прогноз	28

4.3.6 Основные требования к выходным данным	29
5 Средства разработки	30
5.1 Язык разработки.....	30
5.2 Среда разработки	31
5.2.1 Google Colab	31
5.3 Библиотеки	32
5.3.1 Tensorflow/Keras.....	32
5.3.2 Numpy.....	34
6 Разработка системы	35
6.1 Анализ и обработка входных данных.....	35
6.2 Обучение нейронной сети.....	40
6.3 Проектирования взаимодействия между клиентом и сервером	49
7 Тестирование системы.....	54
ЗАКЛЮЧЕНИЕ	58
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ	59
ПРИЛОЖЕНИЕ А Листинг программы	61
ПРИЛОЖЕНИЕ А.1 Файл «index.html».....	62
ПРИЛОЖЕНИЕ А.2 Файл «NetworkLearning.ipynb»	65
ПРИЛОЖЕНИЕ Б Диаграммы.....	71
ПРИЛОЖЕНИЕ Б.1 Диаграмма последовательности – взаимодействие с системой	72
ПРИЛОЖЕНИЕ Б.2 Диаграмма прецедентов - работа пользователя с клиентом.	73
ПРИЛОЖЕНИЕ В Презентация	74

ПЕРЕЧЕНЬ СОКРАЩЕНИЙ И ОБОЗНАЧЕНИЙ

В настоящей работе применяют следующие сокращения и обозначения

ЖКХ – жилищно-коммунальное хозяйство

ИПУ – индивидуальный прибор учёта

ЛС – лицевой счёт

МКД – многоквартирный дом

ОДПУ – общедомовой прибор учёта

СУБД – система управления базами данных

API (Application Programming Interface) – описание способов взаимодействия одной компьютерной программы с другими

RNN (Recurrent neural network) – рекуррентная нейронная сеть

ВВЕДЕНИЕ

Расчёт коммунальных услуг — это важный процесс, обеспечивающий прозрачность и справедливость в начислении платежей за использование различных ресурсов. В соответствии с Постановлением Правительства РФ от 6 мая 2011 г. N 354 "О предоставлении коммунальных услуг собственникам и пользователям помещений в многоквартирных домах и жилых домов", потребителям могут быть представлены следующие виды услуг:

- холодное водоснабжение,
- горячее водоснабжение,
- водоотведение,
- электроснабжение,
- газоснабжение,
- отопление,
- обращение с твердыми коммунальными отходами.

Размер оплаты за коммунальную услугу определяется на основании показаний индивидуальных приборов учёта (далее - ИПУ), а также тарифов, установленных местными органами власти или управляющими компаниями. Для корректного расчёта необходимо учитывать не только фактическое потребление ресурсов, но и особенности начислений, такие как наличие льгот или перерасчётов за некачественные услуги.

Кроме того, важным аспектом является регулярность и точность передачи показаний счётчиков. Жители обязаны вовремя сообщать данные, чтобы избежать начисления среднемесячных норм и, в дальнейшем при продолжительном отсутствии показаний, нормативного потребления, которые могут привести к переплатам, либо недоплатам - если среднемесячные нормы (либо нормативные нормы) окажутся меньше фактического потребления. В некоторых регионах внедряются современные технологии, такие как дистанционное считывание показаний, что упрощает процесс и минимизирует ошибки. Понимание структуры расчёта

коммунальных услуг помогает гражданам более осознанно подходить к вопросам энергосбережения и рационального использования ресурсов.

Существует проблема, связанная с передачей показаний ИПУ: в соответствии с п.3 гл. III Постановления Правительства РФ от 6 мая 2011 г. N 354, «приходящийся на i -е жилое помещение (квартиру) или нежилое помещение объем (количество) холодной воды, предоставленный за расчетный период на общедомовые нужды в многоквартирном доме, оборудованном коллективным (общедомовым) прибором учета холодной воды, определяется» как разница общедомового потребления (исходя из показаний ОДПУ) и суммарное индивидуальное потребления в МКД – которое включает в себя суммарно фактическое, среднемесячное и нормативные потребления по каждому помещению жилого дома. При такой схеме расчёта может возникнуть проблема – если индивидуальное потребление окажется даже на 15-20% меньше общедомового, то неучтённые объёмы будут распределены между собственниками всех помещений МКД как расходы на общедомовые нужды, а это – дополнительные затраты для конечных получателей услуги.

В таком случае можно попробовать спрогнозировать возможное потребление при идеальных условиях, сравнить его с фактическим, и скорректировать результат таким образом, при котором расходы на общедомовые нужды окажутся минимальны.

1 Понятия сферы ЖКХ

Дадим описание некоторых общих терминов из сферы жилищно-коммунального хозяйства.

1.1 Жилищно-коммунальное хозяйство

Жилищно-коммунальное хозяйство (ЖКХ) — отрасль экономики, обеспечивающая функционирование жилых зданий (домов), сетей снабжения водой, теплом и так далее, и сооружений, создающих безопасное, удобное и комфортабельное проживание и нахождение в них людей [3].

ЖКХ включает в себя также объекты социальной инфраструктуры для обслуживания жителей.

1.2 Многоквартирный дом

Многоквартирный дом — здание, в котором расположены жилые помещения (квартиры), нежилые помещения, места общего пользования (напр., лестничная клетка), общедомовое имущество (рисунок 1).



Рисунок 1 – Многоквартирный жилой дом

С юридической точки зрения многоквартирным домом признается совокупность двух и более квартир, имеющих самостоятельные выходы либо на земельный участок, прилегающий к жилому дому, либо в помещения общего пользования в таком доме. Многоквартирный дом содержит в себе

элементы общего имущества собственников помещений в таком доме в соответствии с жилищным законодательством.

В соответствии с определением ЖК РФ ч.3, ст.16 «Квартирой признается структурно обособленное помещение в многоквартирном доме, обеспечивающее возможность прямого доступа к помещениям общего пользования в таком доме и состоящее из одной или нескольких комнат, а также помещений вспомогательного использования, предназначенных для удовлетворения гражданами бытовых и иных нужд, связанных с их проживанием в таком обособленном помещении.»

Многоквартирный дом состоит из:

- жилых помещений (квартир или комнат, если это общежитие). Помещения могут принадлежать на праве собственности как физическим, так и юридическим лицам — частным или государственным организациям,
- нежилых помещений (как правило, помещения, расположенные на первых этажах зданий, и принадлежащие различным юридическим лицам или гражданам),
- иные помещения, не являющиеся частями квартир и обслуживающие более одного помещения, в том числе лестничные площадки, лифты, чердаки и подвалы, в которых имеется инженерное оборудование, обслуживающее более одного помещения, ограждающие и несущие конструкции, земельный участок, на котором расположен многоквартирный дом и иное общее имущество собственников помещений в многоквартирном доме,
- инженерных систем и коммуникаций.

1.3 Индивидуальный прибор учёта

В соответствии с определением п.2 гл. I Постановления Правительства РФ от 6 мая 2011 г. N 354, "индивидуальный прибор учета" - средство измерения (совокупность средств измерения и дополнительного оборудования), устанавливаемое на одно жилое или нежилое помещение в многоквартирном доме (за исключением жилого помещения в коммунальной квартире), на жилой дом (часть жилого дома) или домовладение при наличии

технической возможности и используемое для определения объемов (количества) потребления коммунального ресурса в каждом из указанных помещений, жилым доме (части жилого дома) или домовладении (рисунок 2).



Рисунок 2 – Индивидуальный прибор учёта расхода горячей/холодной воды "Пulsар" Ду 15

Установка ИПУ приносит множество преимуществ, которые положительно сказываются как на жильцах, так и на общем состоянии дома и окружающей среды. Из данных преимуществ можно выделить следующие:

- ИПУ позволяют собственникам оплачивать только фактическое потребление коммунального ресурса, вместо нормативного или среднемесячного, что зачастую оказывается выгоднее – жители квартиры потребляют гораздо меньше нормативного потребления. Следует упомянуть, что возможна и обратная ситуация – фактическое потребление превышает нормативное (среднемесячное), и за счёт этого собственник экономит средства. Данную проблему частично решает применение повышающего коэффициента 1.5 (который применяется в случае непередачи показания или отсутствия прибора учёта, но только в случае, если существует возможность установки прибора учёта) к нормативному потреблению, что увеличивает расходы,

- ИПУ используется для ежемесячного измерения потребления коммунального ресурса. Это позволяет владельцам помещений с

прибором\приборами учёта оценить, сколько ресурсов они потребляют, и возможно, начать чуть бережнее и экономней относиться к трате ресурсов,

- показания с ИПУ передаются в ручном/автоматическом режиме, что позволяет на ранних этапах выявить возможные неисправности с прибором учёта и быстро провести ремонт либо замену прибора,

- данные с приборов позволяют управляющим компаниям и жильцам получать точную информацию о потреблении, что облегчает планирование и контроль расходов.

Во многих регионах установка таких приборов является обязательной или рекомендуется для повышения энергоэффективности и экологической ответственности.

1.4 Общедомовой прибор учёта

В соответствии с определением п.2 гл. I Постановления Правительства РФ от 6 мая 2011 г. N 354, "коллективный (общедомовой) прибор учета" - средство измерения (совокупность средств измерения и дополнительного оборудования), устанавливаемое в многоквартирном доме при наличии технической возможности и используемое для определения объемов (количества) коммунального ресурса, поданного в многоквартирный дом. Для определения объемов (количества) отведенных из многоквартирного дома сточных вод используются коллективный (общедомовой) прибор учета сточных вод, а при его отсутствии - коллективные (общедомовые) приборы учета холодной воды и горячей воды в порядке, установленном настоящими Правилами" (рисунок 3).



Рисунок 3 – Общедомовой (коллективный) прибор учёта расхода тепловой энергии Пульс СТК-15-И

2 Обзор существующих методов прогнозирования

На сегодняшний день известно более сотни методов прогнозирования социально-экономических процессов, однако наиболее часто применяют на практике 15-20 из них. Выбор метода прогнозирования должен при решении определенной (исследовательской) задачи обеспечить функциональную полноту, достоверность и точность прогноза (то есть удовлетворять критериям истинности и верифицируемости) и уменьшить затраты времени [4]. Методы прогнозирования:

- формализованные, эвристические, фактологические, фактографические – базируются на логическом анализе фактов, статистических данных, оценок с использованием, например, математических моделей,
- интуитивные, экспертные – применяются при исследовании и раскрытии ранее неизвестного, в том числе при отсутствии необходимой статистической информации,
- индивидуальные и коллективные - метод интроспекций, приемы аналогий («наивные» методы), метод Дельфи, «мозгового штурма», морфологический анализ и др.,
- логические и математические (экономико-математические); трендовые, тренд-сезонные, на основе регрессионных моделей; имитационные и др.; и средств на прогнозирование (обеспечивать необходимый уровень эффективности) [2, 4].

2.1 Упрощенные методы прогнозирования

Иногда, исходя из целей прогнозирования, исследователю не имеет смысла строить сложные математические модели - например, для получения простого и быстрого прогноза на несколько наблюдений вперед (экспресс прогнозирование) или при прогнозировании факторов в регрессионном анализе. В этом случае целесообразно обратиться к так называемым «упрощенным» методам. Эти методы просты и требуют минимум априорных предположений о процессах, протекающих в объектах. Кроме того, более

сложные математико-статистические модели не всегда дают более точные прогнозы.

К данной группе методов относят:

- метод прогнозирования на основе среднего абсолютного прироста,
- метод прогнозирования на основе среднего темпа роста,
- метод Дрейфа,

и другие.

Преимущества:

- методы просты и позволяют получить результат за короткий срок,
- методы удобны для прогнозирования процессов с 1 – 2 факторами, влияющими на результаты.

В недостатках - практически невозможно получить достоверные результаты для многофакторного процесса.

2.2 Прогнозирование на основе трендовых моделей

Трендовая модель – это простейшая регрессионная модель, описывающая изменение прогнозируемого или анализируемого показателя в зависимости от времени. Данный метод прогнозирования предполагает, что 1) тенденция развития исследуемого объекта сохраниться в будущем, 2) исследуемое явление развивается по плавной траектории, которую можно выразить математически (рисунок 4).

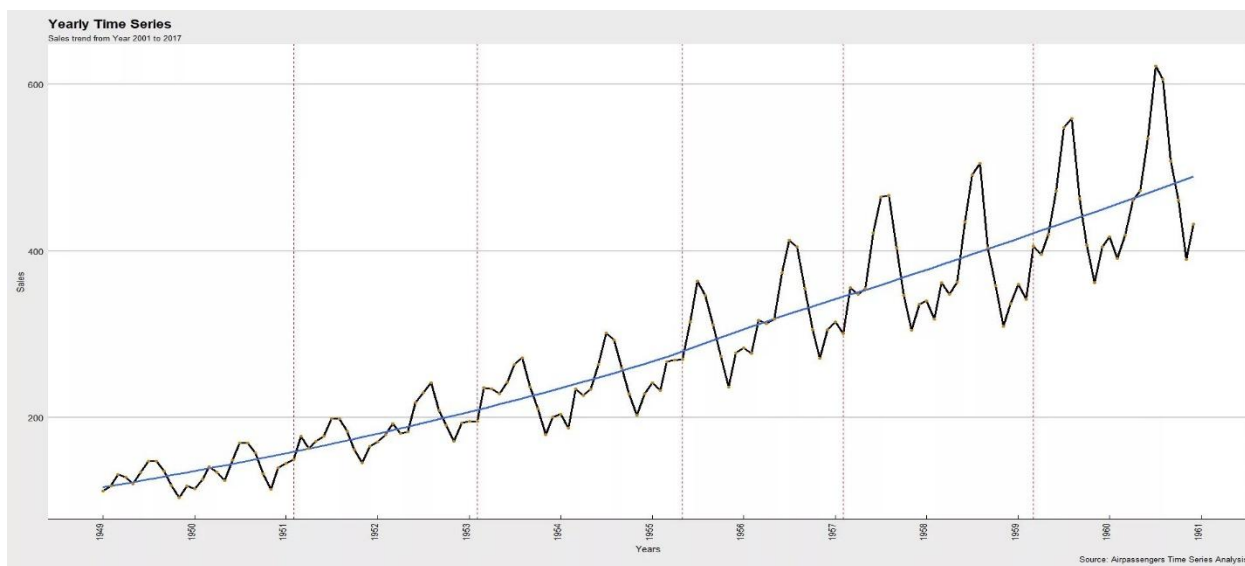


Рисунок 4 – Временной ряд с трендовой и сезонной составляющей

Из преимуществ - удобен для прогноза показателя, функция которого во времени всегда возрастает или убывает. Из недостатков - прогнозируемый показатель должен содержать трендовую составляющую.

2.3 Прогнозирование с использованием сезонной компоненты

Сезонные колебания представляют собой регулярно повторяющиеся подъемы и снижения уровней динамического ряда внутри рассматриваемого периода на протяжении ряда лет (для года – в ежемесячном или поквартальном разрезе). На макроуровне сезонные колебания могут «проявляться» в численности трудовых ресурсов, в динамике доходов, уровнях цен и др. (рисунок 5)

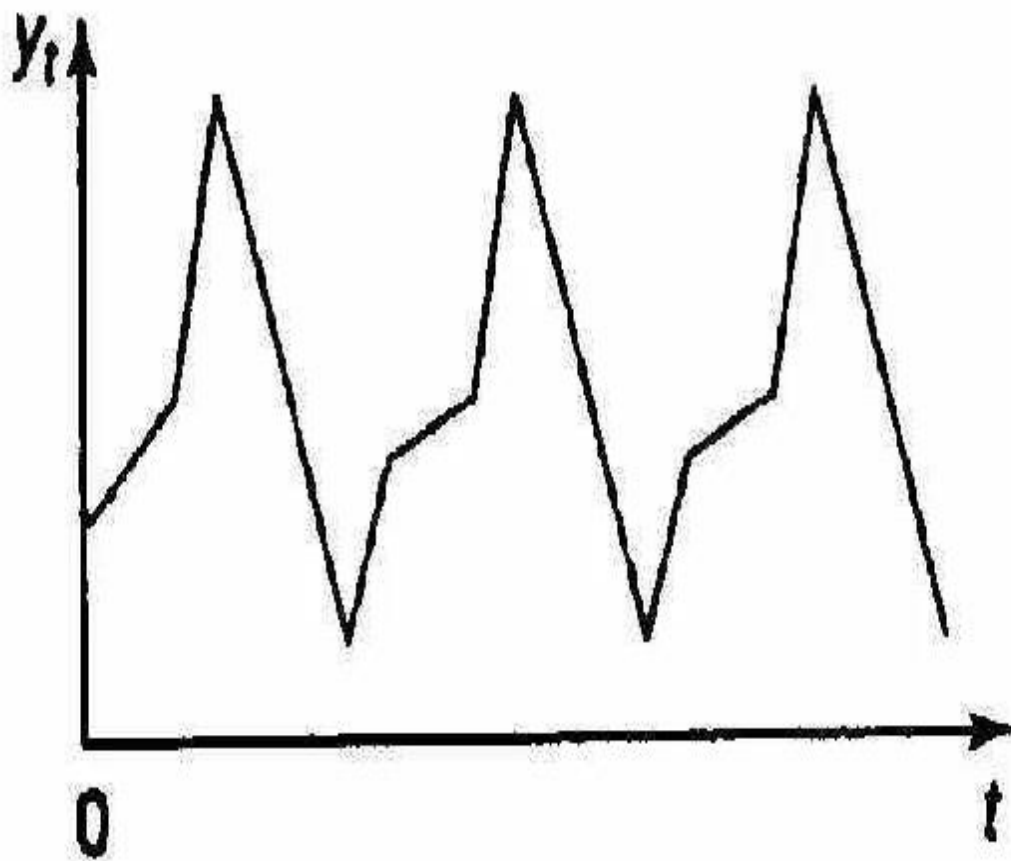


Рисунок 5 – Временной ряд с сезонной составляющей

Существует две модели сезонности:

аддитивная:

$$Y_t = T * S * E;$$

и мультипликативная:

$$Y_t = T + S + E,$$

где T - трендовая компонента;

S - сезонная компонента;

E - случайная компонента.

Поскольку динамический ряд складывается из трех составляющих, влияние каждой из компонент может быть различно, например, чем больше угол наклона линии тренда к среднему значению уровня ряда Y_t , тем больше влияние тренда. Чем больше плавная кривая Y_s отклоняется от линии тренда, тем значительное влияние сезонности. Чем ближе фактические уровни временного ряда Y_t подходят к плавной линии точек, Y_s тем меньше влияние случайной составляющей.

Из преимуществ - удобен для анализа показателей, имеющих периодический характер.

2.4 Регрессионный анализ

Регрессионный анализ подразумевает построение модели зависимости уровня динамики экономического показателя от уровня динамики экономических показателей – аргументов (факторов). Регрессионные модели могут включать различное количество переменных величин. Простейшими моделями являются однофакторные модели, в которых фактором является время (t). Многофакторные модели линейного и нелинейного типа позволяют учитывать влияние нескольких факторов на уровень и динамику прогнозируемого показателя. Такими могут быть модели, описывающие

макроэкономические производственные функции, модели спроса и предложения трудовых ресурсов, модели анализа спроса на отдельные товары и услуги в зависимости от доходов населения, уровня цен, уровня насыщения и др. Переменные регрессионной модели подразделяются на экзогенные (внешние) и эндогенные (внутренние) переменные. Например, в модели производственной функции Кобба-Дугласа, эндогенной переменной является объем производства, а эндогенными переменными являются трудовые ресурсы, капитал.

Из преимуществ - удобен для прогнозирования многофакторных моделей. Из недостатков - требуется корреляция между прогнозируемыми показателями.

2.5 Методы экспертных оценок

Качественные методы преимущественно основываются на экспертных оценках. Суть экспертных методов заключается в том, что прогнозные оценки даются, на основе заключений экспертов. Эксперты подробно аргументируют свою точку зрения о состоянии того или иного объекта или изучаемой проблемы, полагаясь на: интуицию, прежний опыт, творчество и воображение. Экспертные методы прогнозирования используют в случаях:

- когда из-за значительной сложности объекта прогнозирования невозможно учесть влияние многих факторов,
- наличие высокой степени неопределенности информации, имеющейся в прогностической базе или при отсутствии какой-либо информации об объекте прогнозирования. Например, данные методы могут применяться при прогнозировании спроса на новые товары и услуги, эффективности внедрения инноваций в экономику, цены на энергоносители, оценки влияния факторов на исследуемый объект, оценки вероятности исполнения альтернативных прогнозов.

Из преимуществ - простота и применяемость прогнозирования для любых ситуаций, даже в условиях неполноты информации. Из недостатков - субъективизм экспертного мнения.

2.6 Комплексные методы прогнозирования

Опыт применения разнообразных подходов к прогнозированию доказывает, что каждый метод приводит к разным результатам. Итак, как правило, выходит несколько отличных прогнозов одного экономического показателя. Стоит вопрос: преобладает ли какой-либо метод над остальными, и возможно ли каким-то образом скомбинировать прогнозы, полученные разными методами, чтобы построить обобщенный прогноз, который будет более точным, чем индивидуальные?

Объединение можно осуществлять как на основании прогнозов, полученных из разных источников, например, экспертным путем и с помощью фактографических методов, так и с применением прогнозов, построенных с помощью статистических моделей одного класса.

Как уже отмечалось, цель комбинирования прогнозов – повышение качества прогнозирования, единственным (универсальным) критерием которого обычно выступает точность прогнозирования, измеряемая абсолютной либо относительной ошибкой прогнозирования. В соответствии с этой целью комбинирования прогнозов является уменьшение величины ошибки прогнозирования [1].

2.7 Вывод

В результате анализа методов и моделей прогнозирования, принято решение использовать для работы систему, основанную на методе регрессионного анализа.

Такое решение было принято, потому что такой алгоритм наиболее подходит для основного направления разработки программы – расход энергоресурсов и расчет оплаты зависят от множества факторов.

3 Анализ архитектур нейронных сетей

Далее будет рассмотрено несколько популярных архитектур нейронных сетей, а также охарактеризованы их преимущества и недостатки.

3.1 Многослойный перцептрон

Многослойный перцептрон (Multilayer Perceptron, MLP) – базовая нейронная сеть, состоящая из множества слоев нейронов, соединенных между собой. Каждый нейрон в слое получает входные данные от нейронов предыдущего слоя и применяет нелинейную функцию активации для вычисления своего выхода [6]. Обычно упоминают три слоя перцептрона:

- входной слой, на который подают исходные данные,
- выходной слой, куда приходят результаты обработки,
- и скрытые слои, которые выполняют основную функцию.

Схема строения многослойного перцептрона представлена на рисунке 6:

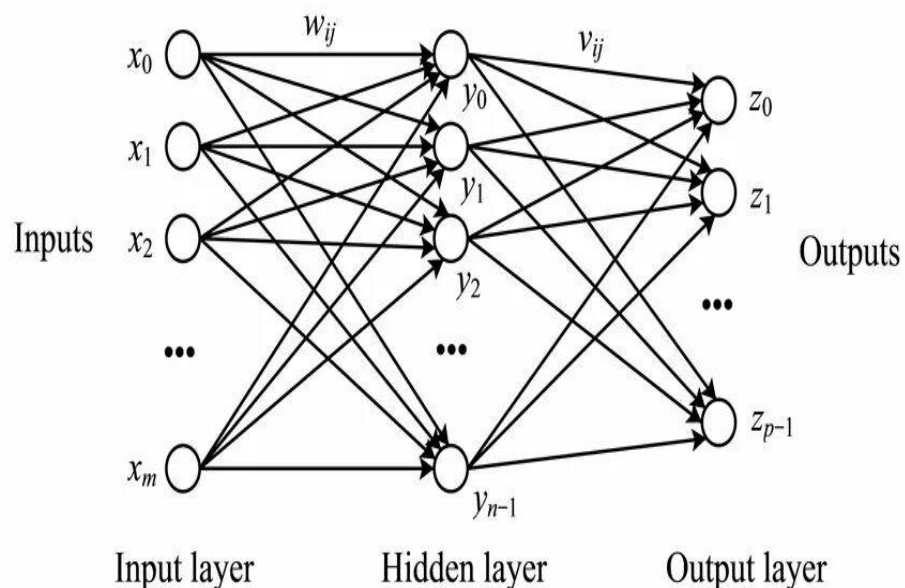


Рисунок 6 – Структура многослойного перцептрона

Следует упомянуть, что скрытые слои могут отличаться функциями активации, но в основном придерживаются одного типа для всех слоев.

Данный тип архитектуры описал Д. Румельхарт в своей монографии «Parallel Distributed Processing: Explorations in the Microstructures of

Cognition», в которой также предложил использовать для его обучения алгоритм обратного распространения ошибки.

MLP обучается с помощью алгоритма обратного распространения ошибки (backpropagation). В процессе обучения веса и смещения нейронов корректируются для минимизации ошибки между предсказанными и фактическими значениями.

3.2 Сверточные нейронные сети

Сверточные нейронные сети (Convolutional Neural Network, CNN) – это тип нейронной сети, специализированной для обработки пространственных данных, таких как изображения. Она использует сверточные фильтры, которые скользят по входным данным и выделяют определенные пространственные паттерны. CNN обучается с использованием алгоритма обратного распространения ошибки, аналогично MLP. Однако в CNN используются дополнительные операции, такие как пулинг (pooling) и нормализация (normalization), которые помогают предотвратить переобучение и извлечь более устойчивые признаки. Была предложена Яном Лекуном в 1988 году в статье «Backpropagation Applied to Handwritten Zip Code Recognition» [8], в которой нейронная сеть с указанной архитектурой, обученная на 7000+ оцифрованных рукописных цифрах, с очень высокой точностью классифицировала тестовые данные (схема работы данной нейронной сети и логика выполнения свертки представлены на рисунке 7).

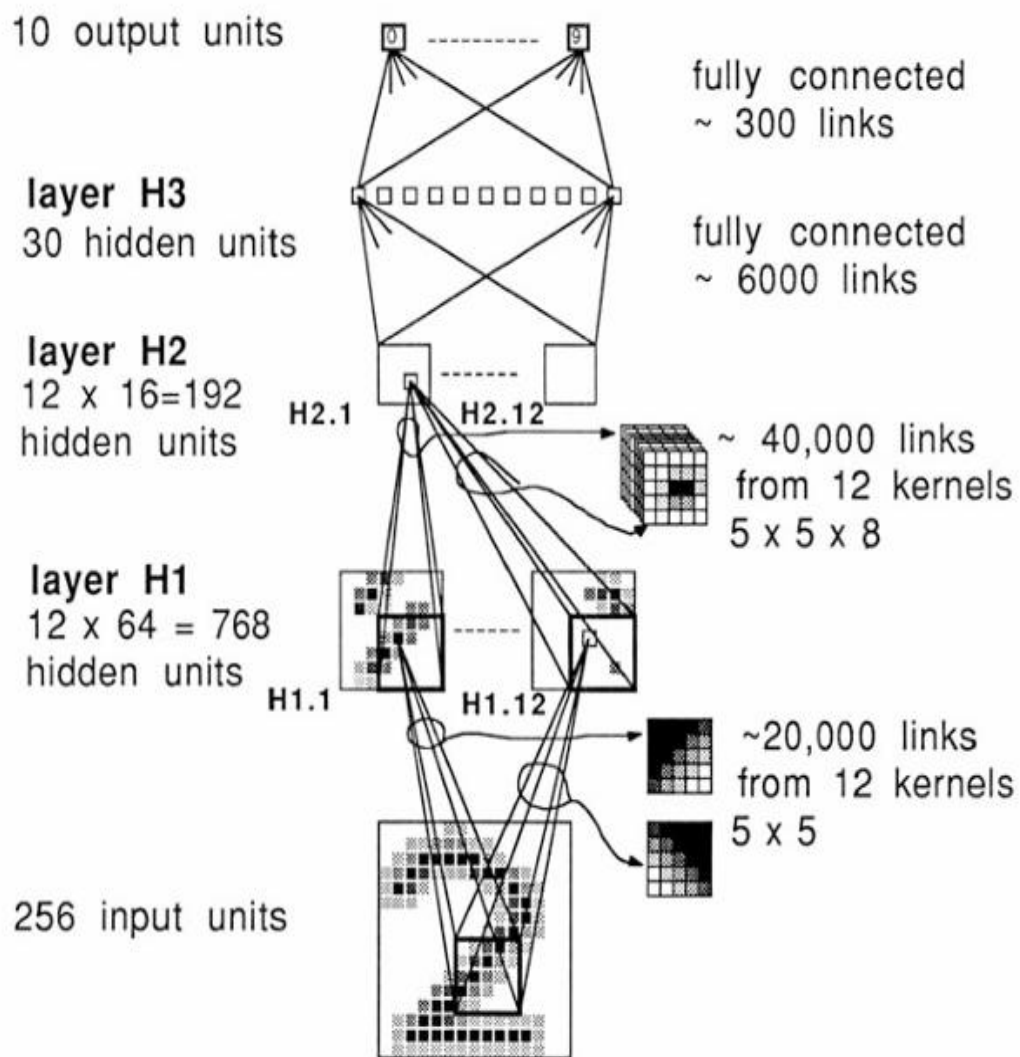


Рисунок 7 – Структура CNN

3.3 Рекуррентные нейронные сети

Рекуррентные нейронные сети (RNN) – это тип нейронной сети, разработанный для обработки последовательных данных, таких как тексты, временные ряды и речь. В отличие от MLP и CNN, RNN обладает памятью, которая позволяет учитывать прошлые состояния для прогнозирования будущих состояний [9] (рисунок 8).

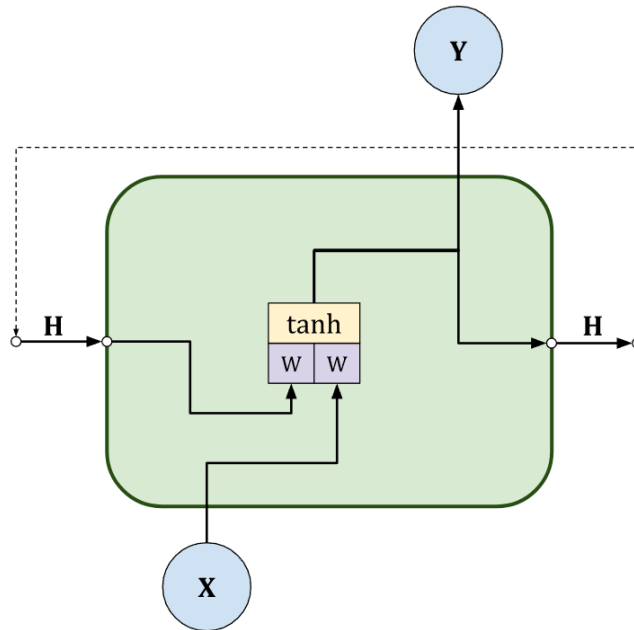


Рисунок 8 - Узел рекуррентной нейронной сети

RNN обучается с использованием алгоритма обратного распространения ошибки, но с модификациями, которые учитывают циклическую природу сети. В RNN используются механизмы, такие как LSTM (Long Short-Term Memory) или GRU (Gated Recurrent Unit), которые помогают сети сохранять информацию о долгосрочных зависимостях в данных.

3.4 Долгая краткосрочная память

Долгая краткосрочная память (LSTM) – это специализированный тип RNN, который решает проблему исчезающего градиента, возникающую при обучении глубоких RNN (рисунок 9).

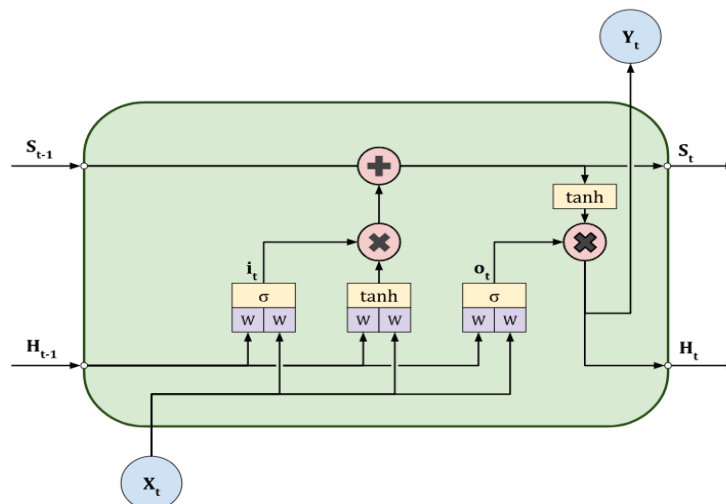


Рисунок 9 - Узел долговременной краткосрочной памяти

Он использует специальные механизмы памяти для запоминания информации на протяжении длительных временных интервалов. В процессе обучения данная архитектура анализирует последовательные данные и запоминает контекстные зависимости, что делает LSTM особенно полезной для задач, связанных с временными рядами или обработкой естественного языка.

Для минимизации общей ошибки LSTM на всём множестве тренировочных последовательностей, итеративный градиентный спуск – например, метод обратного распространения ошибки - развёрнутый во времени может быть использован для изменения каждого из весов пропорционально его производной в зависимости от величины ошибки. Главной проблемой градиентного спуска для стандартных рекуррентных нейронных сетей является то, что градиенты ошибок уменьшаются с экспоненциальной скоростью по мере увеличения временной задержки между важными событиями. Для LSTM-блоков, при распространении величины ошибки в обратном направлении от выходного слоя, ошибка оказывается заперта в памяти блока. Это называют «каруселью ошибок», которая непрерывно «скармливает» ошибку обратно каждому из вентиляей, пока они не будут натренированы отбрасывать значение.

3.5 Трансформер

Трансформер (Transformer) – это архитектура нейронных сетей, которая произвела революцию в обработке естественного языка (NLP) и достигла впечатляющих результатов в различных задачах, таких как машинный перевод, анализ текста, автоматическое составление текстов и других (рисунок 10).

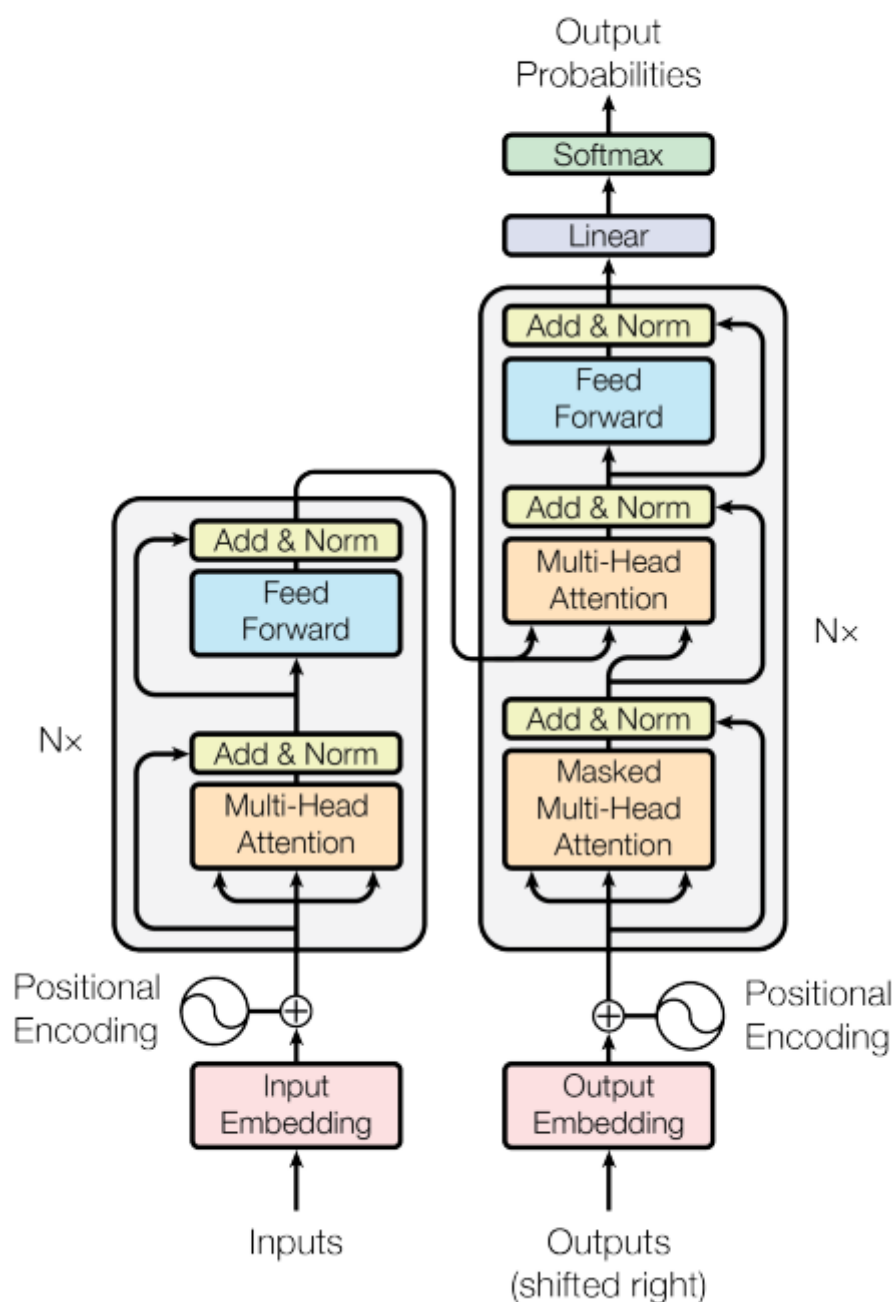


Рисунок 10 – Схема сети-трансформера

Обучение трансформеров – это сложный процесс, который включает в себя несколько этапов: подготовку данных, обучение, оценку и использование. Сначала данные, обычно текст, преобразуются в формат, понятный для нейронной сети, включающий токенизацию, нормализацию и встраивание слов. Затем данные делятся на обучающую, валидационную и тестовую выборки. Обучение происходит с помощью алгоритма обратного

распространения, где модель делает предсказания и корректирует свои веса, чтобы минимизировать ошибку. В процессе обучения применяются различные методы оптимизации и регуляризации для улучшения точности и предотвращения переобучения. После обучения модель оценивается на тестовой выборке, чтобы определить ее эффективность и выбрать наилучшую модель. Ключевыми моментами обучения трансформеров являются использование большого количества данных, вычислительные ресурсы и правильный подбор гиперпараметров и методов регуляризации.

4 Назначение разработки

4.1 Функциональное назначение

Разрабатываемое программное обеспечение должно представлять собой приложение для персонального компьютера, которое получает от пользователя лицевой счёт, а также некоторый набор параметров, в соответствии с которым будет проведен прогноз, формирует прогноз и выводит в качестве результата прогнозируемый расход энергоресурсов в соответствующих единицах измерений либо сумму начислений в рублях. Программа будет представлять собой клиент-серверное приложение, где клиентская часть – небольшой веб-сайт, а серверная часть – ари (рисунок 11).

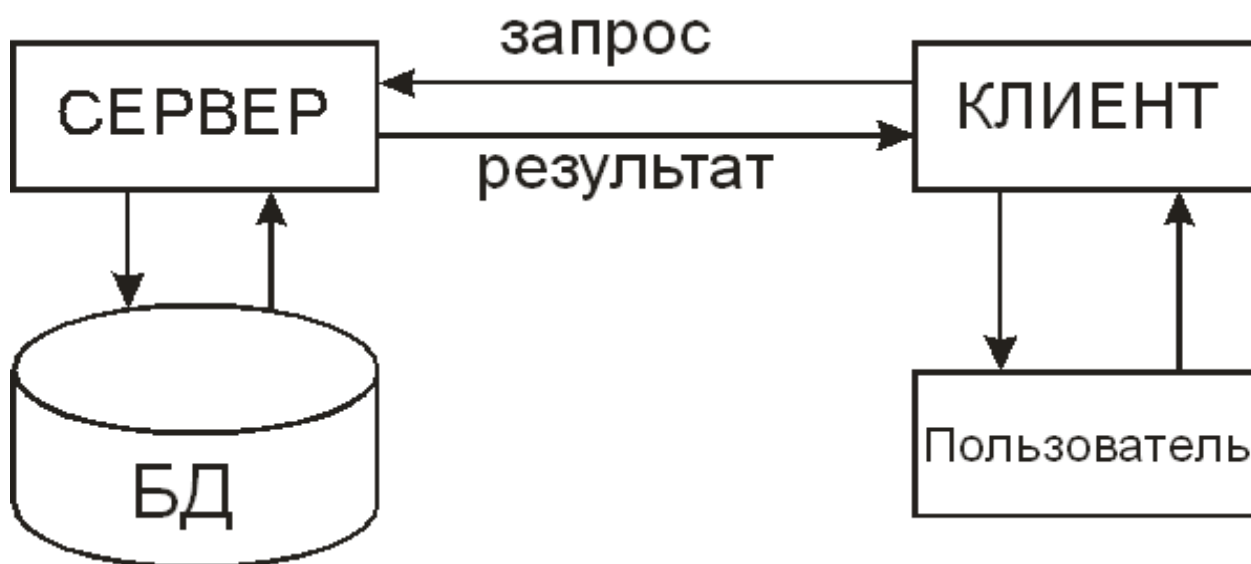


Рисунок 11 – Клиент-серверная архитектура

4.2 Эксплуатационное назначение

Программное обеспечение должно эксплуатироваться на персональном компьютере. Рациональнее всего будет использовать программное обеспечение на сервере системы биллинга, так как там уже будет большая база данных лицевых счетов и данных по ним для приложения.

4.3 Требования к программе

4.3.1 Требования к функциональным характеристикам

Программа должна обеспечивать возможность выполнения перечисленных ниже функций:

- возможность ввода лицевого счета (далее – ЛС), с учётом проверки наличия такового в базе данных,
- выбор типа услуги для получения прогноза из доступных по ЛС,
- выбор прогноза – прогноз по объемам потребления либо прогноз по сумме начислений,
- выбор условий прогноза, доступный для соответствующего типа прогноза,
- вывод прогноза в виде графика «период – объем в у.е. либо сумма в рублях»,
- возможность сохранения полученных результатов в виде изображения, либо в JSON файле определенного содержимого [5].

4.3.2 Требования к организации входных данных

На вход программы должен подаваться ЛС потребителя, тип услуги, для которой производится прогноз, тип прогноза, а также условие прогноза. В зависимости от выбранного условия прогноза могут появиться дополнительные поля для ввода.

4.3.3 Требования к организации выходных данных

Выходные данные приложения будут выведены на графический интерфейс. В окне интерфейса будет выведен график, содержащий результаты прогноза, а также две кнопки – для загрузки результата в качестве изображения и загрузке в виде json файла, содержащего входные данные, а также результаты прогноза.

4.3.4 Макет интерфейса приложения

После открытия приложения нужно ввести ЛС потребителя, выбрать услугу из выпадающего списка, выбрать тип прогноза – объема либо суммы начислений, выбрать условия прогноза и, при необходимости, ввести дополнительные данные, и нажать на кнопку «Вывести прогноз». После расчёта прогноза будет выведен график с результатами, а также кнопки для скачивания результатов. Примерный макет интерфейса представлена на рисунке 12.

Введите лицевой счёт: Выберите услугу:

Прогноз для: ☒ Объем потребления ☐ Сумма начисления

Прогноз:

Результаты прогноза:



Рисунок 12 – Интерфейс приложения

4.3.5 Основные требования к данным, на основе которых формируется прогноз

Для формирования корректного прогноза в базе данных ЛС должна содержаться следующая информация:

- площадь помещения, к которому привязан ЛС,
- поставляется ли услуга на данный ЛС,
- показания счётчиков либо расход ресурса за предыдущие месяцы (в случае наличия индивидуального прибора учёта на ЛС),
- нормативы потребления на услуги,
- тарифы на услуги,
- признаки определения объема потребления услуги за предыдущие месяцы – по ИПУ, по среднему, по нормативу,
- дата окончания поверки ИПУ (при наличии).

4.3.6 Основные требования к выходным данным

В качестве результатов прогнозирования в интерфейсе должен отобразиться график, показывающий потребления за текущий\предыдущий периоды, а так же значения прогноза.

Должен присутствовать механизм сохранения результатов прогноза в двух форматах: в виде JSON файла и в виде PNG изображения графика.

5 Средства разработки

5.1 Язык разработки

В качестве языка разработки был выбран Python (рисунок 13).

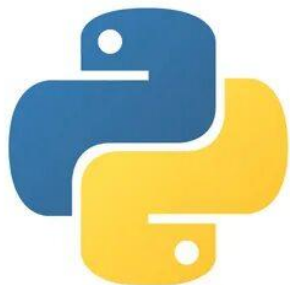


Рисунок 13 – Python (логотип)

Python - высокоуровневый язык программирования общего назначения с динамической строгой типизацией и автоматическим управлением памятью, ориентированный на повышение производительности разработчика, читаемости кода и его качества, а также на обеспечение переносимости написанных на нём программ. Язык является полностью объектно-ориентированным в том плане, что всё является объектами.

Необычной особенностью языка является выделение блоков кода отступами. Python — интерпретируемый язык, использующийся в том числе для написания скриптов. Недостатками языка являются зачастую более низкая скорость работы и более высокое потребление памяти написанными на нём программами по сравнению с аналогичным кодом, написанным на компилируемых языках, таких как С или С++. Python является самым популярным языком разработки в мире, согласно индексу ТЮВЕ, который показывает процент запросов в поисковых системах относительно всех запросов. Python также является основным языком для аналитиков данных и разработчиков нейронных сетей – этому способствует обилие библиотек и удобство работы в целом.

5.2 Среда разработки

5.2.1 Google Colab

Google Colab (Colaboratory) — это облачный сервис от Google, который позволяет писать и выполнять Python-код в браузере без установки дополнительных программ (рисунок 14).



Рисунок 14 – Google Colaboratory (логотип)

Данный сервис обладает несколькими преимуществами:

- возможность использовать GPU/TPU – любая более-менее крупная нейронная сеть уже с трудом обучается на CPU, и в основном крупные сети обучают на GPU, поэтому наличие такой возможности в Google Colab – определенно плюс,
- удобное меню навигации по файлу, которое автоматически подхватывает созданные вами заголовки,
- удобство совместной работы. Легко поделиться проектом по ссылке, как обычным Google-документом, настроить уровень доступа (чтение, комментирование или редактирование), вносить правки так, чтобы они автоматически синхронизировались у всех пользователей. Также при совместной работе у всех участников согласуются наличие и версии библиотек. Если чего-то нет у вас, то нет и у товарищей по проекту. Значит, код `!pip install scikit-learn` нужен либо всем, либо никому. Согласование происходит само собой, тратить время на синхронизацию не придётся.

Google Colab будет использоваться для большей части разработки, но понадобится также локальная IDE для сборки проекта и публикации.

5.2.2 Visual Studio Code

Visual Studio Code (VS Code) — текстовый редактор, разработанный Microsoft для Windows, Linux и macOS (рисунок 15).

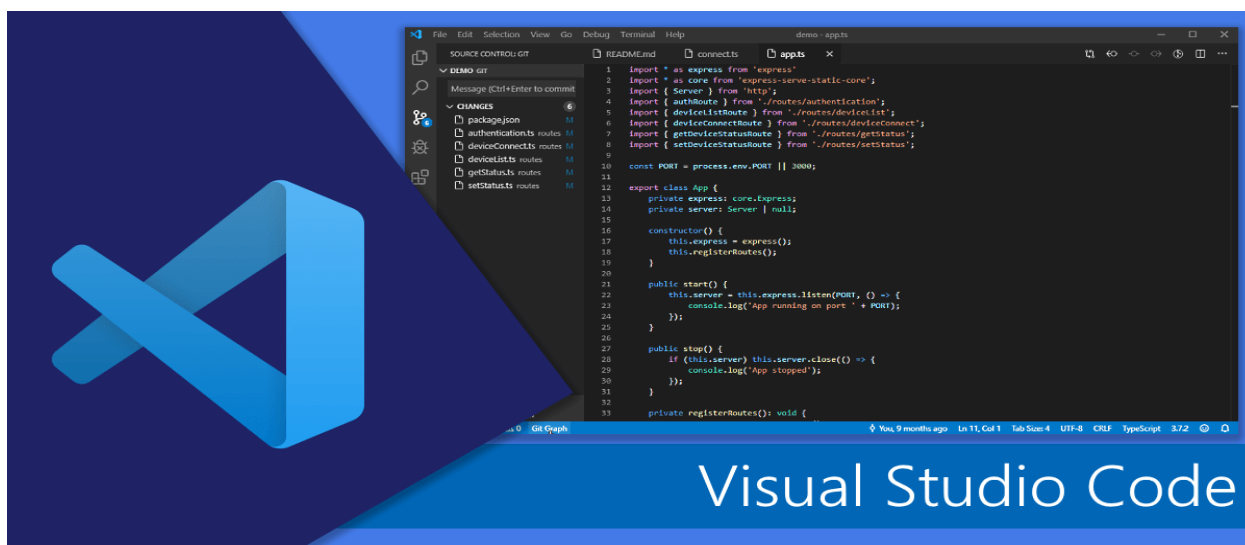


Рисунок 15 –Visual Studio Code

Позиционируется как «лёгкий» редактор кода для кроссплатформенной разработки веб- и облачных приложений. Включает в себя отладчик, инструменты для работы с Gi, подсветку синтаксиса, IntelliSense и средства для рефакторинга. Имеет широкие возможности для кастомизации: пользовательские темы, сочетания клавиш и файлы конфигурации. Распространяется бесплатно, разрабатывается как программное обеспечение с открытым исходным кодом, но готовые сборки распространяются под проприетарной лицензией [12].

Visual Studio Code основан на Electron и реализуется через веб-редактор Monaco, разработанный для Visual Studio Online [12].

5.3 Библиотеки

5.3.1 Tensorflow/Keras

Keras — это API высокого уровня платформы TensorFlow. Он предоставляет доступный и высокопроизводительный интерфейс для решения задач машинного обучения (ML) с упором на современное глубокое обучение. Keras охватывает каждый этап рабочего процесса машинного обучения: от обработки данных до настройки гиперпараметров и

развертывания. Он был разработан с упором на возможность быстрого экспериментирования (рисунок 16).



Рисунок 16 –TensorFlow + Keras (логотип)

С Keras у вас есть полный доступ к масштабируемости и кроссплатформенным возможностям TensorFlow. Вы можете запускать Keras на модуле TPU или больших кластерах графических процессоров, а также экспортировать модели Keras для запуска в браузере или на мобильных устройствах. Вы также можете обслуживать модели Keras через веб-API.

Keras предназначен для снижения когнитивной нагрузки за счет достижения следующих целей:

- он предлагает простые и согласованные интерфейсы,
- минимизирует количество действий, необходимых для распространенных случаев использования,
- предоставляет четкие и действенные сообщения об ошибках,
- он следует принципу постепенного раскрытия сложности: начать легко, и вы можете выполнять сложные рабочие процессы, обучаясь по ходу дела,
- помогает вам написать краткий, читаемый код [10].

5.3.2 Numpy

NumPy – это библиотека с открытым исходным кодом, некогда отделившаяся от проекта SciPy. NumPy является наследником Numeric и NumArray. Основан NumPy на библиотеке LAPACK, которая написана на Fortran. Не-python альтернативой для NumPy является Matlab (рисунок 17).



Рисунок 17 – NumPy (логотип)

В силу того, что NumPy базируется на Fortran это быстрая библиотека. А в силу того, что поддерживает векторные операции с многомерными массивами — крайне удобная [11].

6 Разработка системы

6.1 Анализ и обработка входных данных

В качестве входных данных используется датасет со следующими полями (таблица 1):

Таблица 1 – Входные данные

Наименование признака	Тип данных	Описание
Period	Datetime	Период, в котором приняты показания
AccountNumber	Integer	Номер лицевого счёта
AddressCode	Integer	Идентификатор МКД в системе
IsResidential	String	Помещение жилое/нежилое. Принимает значение «Yes» или «No»
Area	Float	Площадь помещения
Residents	Float	Количество проживающих

Продолжение таблицы 1

HeatingCounterStatus	String	Статус состояния прибора учета тепловой энергии и данных по показаниям за указанный период. Принимает следующие значения: <ul style="list-style-type: none"> – NoCounter – не оборудован прибором учёта, – ByCounter – были переданы показания, – ByMean – показания не передавались, объем определяется как среднее за 6 мес., – ByNormative – показания не передавались либо ИПУ вышел из строя, объем определяется как нормативный.
HeatingVolumeConsumed	Float	Потреблённый объём
WaterCounterStatus	String	Статус состояния прибора учета воды и данных по показаниям за указанный период. Описание признаков - см. HeatingCounterStatus
WaterVolumeConsumed	Float	Потреблённый объём

Загрузим данные показаний от 100 000 лицевых счетов за 4 года с информацией по приборам учёта из .csv файла. Данные представлены на рисунке 18

```
[ ] df = pd.read_csv('/content/drive/MyDrive/Мармстрепская/dataset.csv', delimiter=';', decimal=',')
```

```
[ ] df
```

	Period	AccountNumber	AddressCode	IsResidential	Area	Residents	HeatingCounterStatus	HeatingVolumeConsumed	WaterCounterStatus	WaterVolumeConsumed
0	31.01.2021 0:00	701021163	100	Yes	95.0	6	NoCounter	0.000	ByCounter	11.0
1	28.02.2021 0:00	701021163	100	Yes	95.0	6	NoCounter	0.000	ByCounter	6.0
2	31.03.2021 0:00	701021163	100	Yes	95.0	6	NoCounter	0.000	ByCounter	8.0
3	30.04.2021 0:00	701021163	100	Yes	95.0	6	NoCounter	0.000	ByCounter	7.0
4	31.05.2021 0:00	701021163	100	Yes	95.0	6	NoCounter	0.000	ByCounter	8.0
...	...	...	...	...	...	...	...	...	...	...
1028781	30.11.2024 0:00	720133086	200028	Yes	71.6	1	ByCounter	0.006	ByCounter	0.0
1028782	31.01.2025 0:00	720133086	200028	Yes	71.6	1	ByCounter	0.006	ByCounter	0.0
1028783	28.02.2025 0:00	720133086	200028	Yes	71.6	1	ByCounter	0.240	ByCounter	0.0
1028784	31.03.2025 0:00	720133086	200028	Yes	71.6	1	ByCounter	0.737	ByCounter	0.0
1028785	30.04.2025 0:00	720133086	200028	Yes	71.6	1	ByCounter	0.310	ByCounter	0.0

1028786 rows x 10 columns

Рисунок 18 – Входные данные

Проверим данные на наличие NaN. При проверке была найдена одна запись с NaN. Проверим данные по указанному индексу (рисунок 19):

```
df.iloc[666828]
```

	666828
Period	31.10.2021 0:00
AccountNumber	714039896
AddressCode	1415
IsResidential	Yes
Area	60.2
Residents	1
HeatingCounterStatus	NoCounter
HeatingVolumeConsumed	0.0
WaterCounterStatus	NaN
WaterVolumeConsumed	1.0

Рисунок 19 – Строка с NaN

Можно увидеть, что для данной записи отсутствует WaterCounterStatus. Обращаемся ко всем данным по указанному лицевому счёту. Полная выборка представлена на рисунке 20.

df.loc[df['AccountNumber'].isin([714039896])]

	Period	AccountNumber	AddressCode	IsResidential	Area	Residents	HeatingCounterStatus	HeatingVolumeConsumed	WaterCounterStatus	WaterVolumeConsumed
666825	31.03.2021 0:00	714039896	1415	Yes	60.2	1	NoCounter	0.0	ByCounter	0.663
666826	30.04.2021 0:00	714039896	1415	Yes	60.2	1	NoCounter	0.0	ByCounter	0.670
666827	30.09.2021 0:00	714039896	1415	Yes	60.2	1	NoCounter	0.0	ByCounter	5.000
666828	31.10.2021 0:00	714039896	1415	Yes	60.2	1	NoCounter	0.0	NaN	1.000
666829	28.02.2022 0:00	714039896	1415	Yes	60.2	1	NoCounter	0.0	ByCounter	1.000
666830	31.03.2022 0:00	714039896	1415	Yes	60.2	1	NoCounter	0.0	ByCounter	1.000
666831	31.05.2022 0:00	714039896	1415	Yes	60.2	1	NoCounter	0.0	ByCounter	1.000
666832	31.08.2022 0:00	714039896	1415	Yes	60.2	1	NoCounter	0.0	ByCounter	1.000
666833	31.10.2022 0:00	714039896	1415	Yes	60.2	1	NoCounter	0.0	ByCounter	1.000
666834	30.11.2022 0:00	714039896	1415	Yes	60.2	1	NoCounter	0.0	ByCounter	2.000
666835	31.12.2022 0:00	714039896	1415	Yes	60.2	1	NoCounter	0.0	ByCounter	3.000
666836	28.02.2023 0:00	714039896	1415	Yes	60.2	1	NoCounter	0.0	ByCounter	4.000
666837	30.04.2023 0:00	714039896	1415	Yes	60.2	1	NoCounter	0.0	ByCounter	3.500
666838	31.05.2023 0:00	714039896	1415	Yes	60.2	1	NoCounter	0.0	ByCounter	2.500
666839	30.06.2023 0:00	714039896	1415	Yes	60.2	1	NoCounter	0.0	ByCounter	5.000
666840	31.08.2023 0:00	714039896	1415	Yes	60.2	1	NoCounter	0.0	ByCounter	4.000
666841	30.09.2023 0:00	714039896	1415	Yes	60.2	1	NoCounter	0.0	ByCounter	2.000
666842	31.10.2023 0:00	714039896	1415	Yes	60.2	1	NoCounter	0.0	ByCounter	3.000
666843	31.12.2023 0:00	714039896	1415	Yes	60.2	1	NoCounter	0.0	ByCounter	5.000

Рисунок 20 – Данные по лицевому счёту

Можно видеть, что по указанному лицевому счёту ежемесячно передаются показания. Укажем для пропущенного значения соответствующий статус (рисунок 21).

df.iloc[666828]

Period	31.10.2021 0:00
AccountNumber	714039896
AddressCode	1415
IsResidential	Yes
Area	60.2
Residents	1
HeatingCounterStatus	NoCounter
HeatingVolumeConsumed	0.0
WaterCounterStatus	ByCounter
WaterVolumeConsumed	1.0

Рисунок 21 – Строка с замененным NaN

Проанализируем сами признаки на принадлежность к категориальным/числовым (рисунок 22):

```

from sklearn.compose import make_column_selector as selector

categorical_columns_selector = selector(dtype_include=object)
categorical_columns = categorical_columns_selector(df)
print('Категориальные признаки:', categorical_columns)

numerical_columns_selector = selector(dtype_include=[int, float])
numerical_columns = numerical_columns_selector(df)
print('Числовые признаки', numerical_columns)

```

Категориальные признаки: ['Period', 'IsResidential', 'HeatingCounterStatus', 'WaterCounterStatus']
Числовые признаки ['AccountNumber', 'AddressCode', 'Area', 'Residents', 'HeatingVolumeConsumed', 'WaterVolumeConsumed']

Рисунок 22 – Классификация признаков

Предварительно уберем из категориальных признаков Period – данный признак является временной меткой и не относится к категориальным признакам. Также уберем признаки AccountNumber и Address – они нужны для дальнейшей работы пользователей с указанными данными, и не должны быть модифицированы. После приведем категориальные признаки к числовым. Алгоритм преобразования и результаты представлены на рисунках 23-24.

```

[28] categorical_columns.remove('Period')
numerical_columns.remove('AccountNumber')
numerical_columns.remove('AddressCode')

```

```

from sklearn.preprocessing import OrdinalEncoder

df_ord = df.iloc[: -1].copy()

encoder = OrdinalEncoder(categories = [['Yes', 'No']])
encoder = encoder.fit(df_ord[['IsResidential']])
df_ord[['IsResidential']] = encoder.transform(df_ord[['IsResidential']])
print('IsResidential', encoder.categories_, df_ord[['IsResidential']].unique())

categories = [['NoCounter', 'ByCounter', 'ByMean', 'ByNormative']]
encoder = OrdinalEncoder(categories = categories)
encoder = encoder.fit(df_ord[['HeatingCounterStatus']])
df_ord[['HeatingCounterStatus']] = encoder.transform(df_ord[['HeatingCounterStatus']])
print('HeatingCounterStatus', encoder.categories_, df_ord[['HeatingCounterStatus']].unique())

encoder = OrdinalEncoder(categories = categories)
encoder = encoder.fit(df_ord[['WaterCounterStatus']])
df_ord[['WaterCounterStatus']] = encoder.transform(df_ord[['WaterCounterStatus']])
print('WaterCounterStatus', encoder.categories_, df_ord[['WaterCounterStatus']].unique())

```

IsResidential [array(['Yes', 'No'], dtype=object)] [0. 1.]
HeatingCounterStatus [array(['NoCounter', 'ByCounter', 'ByMean', 'ByNormative'], dtype=object)] [0. 1. 2. 3.]
WaterCounterStatus [array(['NoCounter', 'ByCounter', 'ByMean', 'ByNormative'], dtype=object)] [1. 3. 2. 0.]

Рисунок 23 – Приведение категориальных признаков к числовым

[] df_ord



	Period	AccountNumber	AddressCode	IsResidential	Area	Residents	HeatingCounterStatus	HeatingVolumeConsumed	WaterCounterStatus	WaterVolumeConsumed
0	31.01.2021 0:00	701021163	100	0.0	95.0	6	0.0	0.000	1.0	11.0
1	28.02.2021 0:00	701021163	100	0.0	95.0	6	0.0	0.000	1.0	6.0
2	31.03.2021 0:00	701021163	100	0.0	95.0	6	0.0	0.000	1.0	8.0
3	30.04.2021 0:00	701021163	100	0.0	95.0	6	0.0	0.000	1.0	7.0
4	31.05.2021 0:00	701021163	100	0.0	95.0	6	0.0	0.000	1.0	8.0
...	...	...	...	...	...	...	...	...	...	...
1028780	31.10.2024 0:00	720133086	200028	0.0	71.6	1	1.0	0.004	1.0	0.0
1028781	30.11.2024 0:00	720133086	200028	0.0	71.6	1	1.0	0.006	1.0	0.0
1028782	31.01.2025 0:00	720133086	200028	0.0	71.6	1	1.0	0.006	1.0	0.0
1028783	28.02.2025 0:00	720133086	200028	0.0	71.6	1	1.0	0.240	1.0	0.0
1028784	31.03.2025 0:00	720133086	200028	0.0	71.6	1	1.0	0.737	1.0	0.0

1028785 rows x 10 columns

Рисунок 24 – Датасет после обработки

На данном этапе предварительная подготовка данных закончена.

6.2 Обучение нейронной сети

Для обучения возьмём данные по одному лицевого счёту. В качестве примера используем ЛС №701021163 – по нему достаточно долго ведётся история, данных для прогнозирования достаточно (рисунок 25):

[] df_by_account_number = df_ord.groupby(['AccountNumber'], group_keys=True)[df.columns.drop('AccountNumber')]

[] df_by_account_number.get_group(701021163)



<ipython-input-333-be1a4af1384e>:1: FutureWarning: When grouping with a length-1 list-like, you will need to pass a length-1 tuple to get_group in a df_by_account_number.get_group(701021163)

	Period	AddressCode	IsResidential	Area	Residents	HeatingCounterStatus	HeatingVolumeConsumed	WaterCounterStatus	WaterVolumeConsumed
0	31.01.2021 0:00	100	0.0	95.0	6	0.0	0.0	1.0	11.0
1	28.02.2021 0:00	100	0.0	95.0	6	0.0	0.0	1.0	6.0
2	31.03.2021 0:00	100	0.0	95.0	6	0.0	0.0	1.0	8.0
3	30.04.2021 0:00	100	0.0	95.0	6	0.0	0.0	1.0	7.0
4	31.05.2021 0:00	100	0.0	95.0	6	0.0	0.0	1.0	8.0
5	30.06.2021 0:00	100	0.0	95.0	6	0.0	0.0	1.0	4.0
6	31.08.2021 0:00	100	0.0	95.0	6	0.0	0.0	1.0	2.0
7	31.10.2021 0:00	100	0.0	95.0	6	0.0	0.0	1.0	12.0
8	30.11.2021 0:00	100	0.0	95.0	6	0.0	0.0	1.0	4.0
9	31.12.2021 0:00	100	0.0	95.0	6	0.0	0.0	1.0	10.0
10	31.01.2022 0:00	100	0.0	95.0	6	0.0	0.0	1.0	6.0
11	28.02.2022 0:00	100	0.0	95.0	6	0.0	0.0	1.0	8.0
12	31.03.2022 0:00	100	0.0	95.0	6	0.0	0.0	1.0	6.0
13	30.04.2022 0:00	100	0.0	95.0	6	0.0	0.0	1.0	6.0
14	31.05.2022 0:00	100	0.0	95.0	6	0.0	0.0	1.0	4.0
15	30.06.2022 0:00	100	0.0	95.0	6	0.0	0.0	1.0	3.0
16	31.07.2022 0:00	100	0.0	95.0	6	0.0	0.0	1.0	1.0

Рисунок 25 – Выборка по ЛС № 701021163

В качестве архитектуры нейронной сети используем LSTM архитектуру. Обучим нейронную сеть для прогнозирования потребления по воде на определённый месяц относительно предыдущих 12 мес. потребления. Для обучения воспользуемся только объёмами – приведем данные к одномерному временному ряду (рисунки 26-27).



Рисунок 26 – Приводим данные к одномерному временному ряду

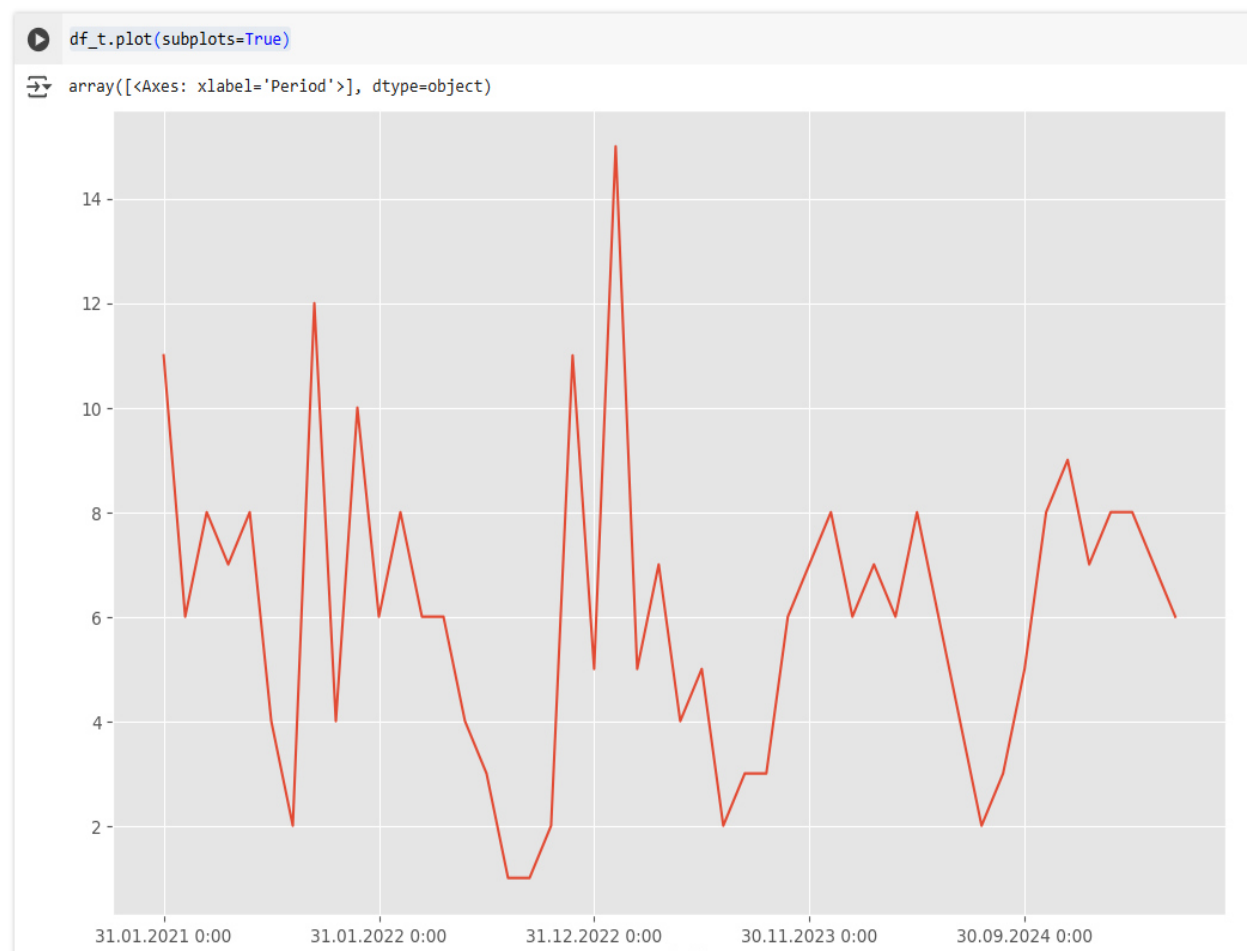


Рисунок 27 – График потребления

Подготовим данные для указанного типа прогноза: в качестве обучающей выборки используем значения за последние 3 года и проведем стандартизацию значений потребления (рисунки 28-30):

```
[ ] TRAIN_SPLIT = 36
```

```
df_t = df_t.values
```

```
[ ] uni_train_mean = df_t[:TRAIN_SPLIT].mean()  
    uni_train_std = df_t[:TRAIN_SPLIT].std()
```

```
[ ] df_t = (df_t-uni_train_mean)/uni_train_std
```

Рисунок 28 – Стандартизация данных

```
def univariate_data(dataset, start_index, end_index, history_size, target_size):  
    data = []  
    labels = []  
  
    start_index = start_index + history_size  
    if end_index is None:  
        end_index = len(dataset) - target_size  
  
    for i in range(start_index, end_index):  
        indices = range(i-history_size, i)  
        data.append(np.reshape(dataset[indices], (history_size, 1)))  
        labels.append(dataset[i+target_size])  
    return np.array(data), np.array(labels)
```

```
[ ] univariate_past_history = 6  
    univariate_future_target = 0  
  
    x_train_uni, y_train_uni = univariate_data(df_t, 0, TRAIN_SPLIT,  
                                                univariate_past_history,  
                                                univariate_future_target)  
    x_val_uni, y_val_uni = univariate_data(df_t, TRAIN_SPLIT, None,  
                                            univariate_past_history,  
                                            univariate_future_target)
```

Рисунок 29 – Подготовка данных

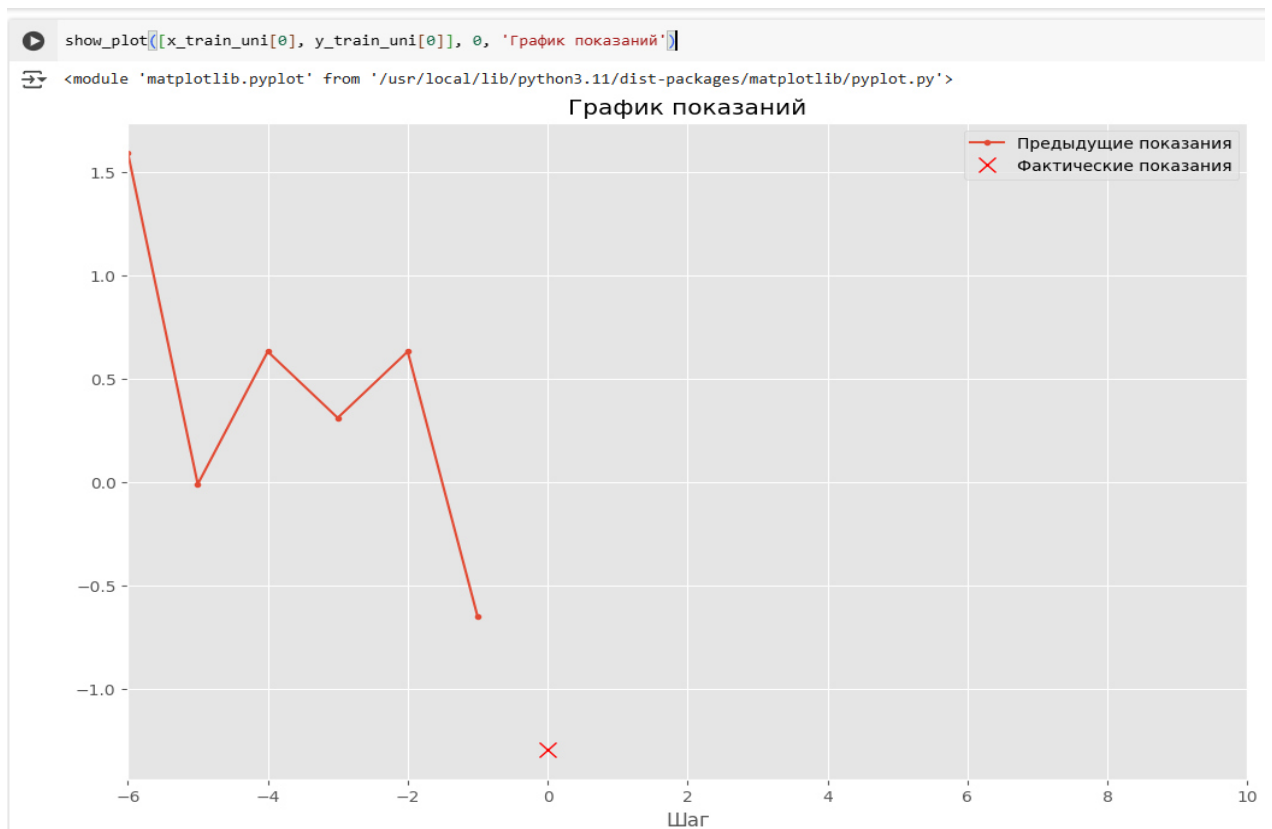


Рисунок 30 – График показаний (стандартизованный)

Подготовим тренировочный и валидационные датасеты, создадим модель и обучим ее. Тренировать будем 10 эпох с 50 итерациями в каждой (рисунки 31-33):

```
[58] import tensorflow as tf
```

```
[59] BATCH_SIZE = 256
      BUFFER_SIZE = 10000

      train_univariate = tf.data.Dataset.from_tensor_slices((x_train_uni, y_train_uni))
      train_univariate = train_univariate.cache().shuffle(BUFFER_SIZE).batch(BATCH_SIZE).repeat()

      val_univariate = tf.data.Dataset.from_tensor_slices((x_val_uni, y_val_uni))
      val_univariate = val_univariate.batch(BATCH_SIZE).repeat()
```

```
simple_lstm_model = tf.keras.models.Sequential([
    tf.keras.layers.LSTM(8, input_shape=x_train_uni.shape[-2:]),
    tf.keras.layers.Dense(1)
])

simple_lstm_model.compile(optimizer='adam', loss='mae')
```

Рисунок 31 – Подготовка тренировочного и валидационного датасетов и инициализация модели

```

▶ EVALUATION_INTERVAL = 50
  EPOCHS = 10

simple_lstm_model.fit(train_univariate, epochs=EPOCHS,
                      steps_per_epoch=EVALUATION_INTERVAL,
                      validation_data=val_univariate, validation_steps=50)

```

```

↔ Epoch 1/10
50/50 ————— 1s 12ms/step - loss: 0.4582 - val_loss: 0.3219
Epoch 2/10
50/50 ————— 1s 23ms/step - loss: 0.4262 - val_loss: 0.3130
Epoch 3/10
50/50 ————— 1s 13ms/step - loss: 0.4003 - val_loss: 0.2887
Epoch 4/10
50/50 ————— 1s 12ms/step - loss: 0.3763 - val_loss: 0.2536
Epoch 5/10
50/50 ————— 1s 15ms/step - loss: 0.3537 - val_loss: 0.2371
Epoch 6/10
50/50 ————— 1s 12ms/step - loss: 0.3319 - val_loss: 0.2352
Epoch 7/10
50/50 ————— 0s 6ms/step - loss: 0.3124 - val_loss: 0.2402
Epoch 8/10
50/50 ————— 0s 6ms/step - loss: 0.2956 - val_loss: 0.2476
Epoch 9/10
50/50 ————— 0s 6ms/step - loss: 0.2808 - val_loss: 0.2541
Epoch 10/10
50/50 ————— 0s 5ms/step - loss: 0.2689 - val_loss: 0.2521
<keras.src.callbacks.history.History at 0x7ab3848880d0>

```

Рисунок 32 – Результаты обучения

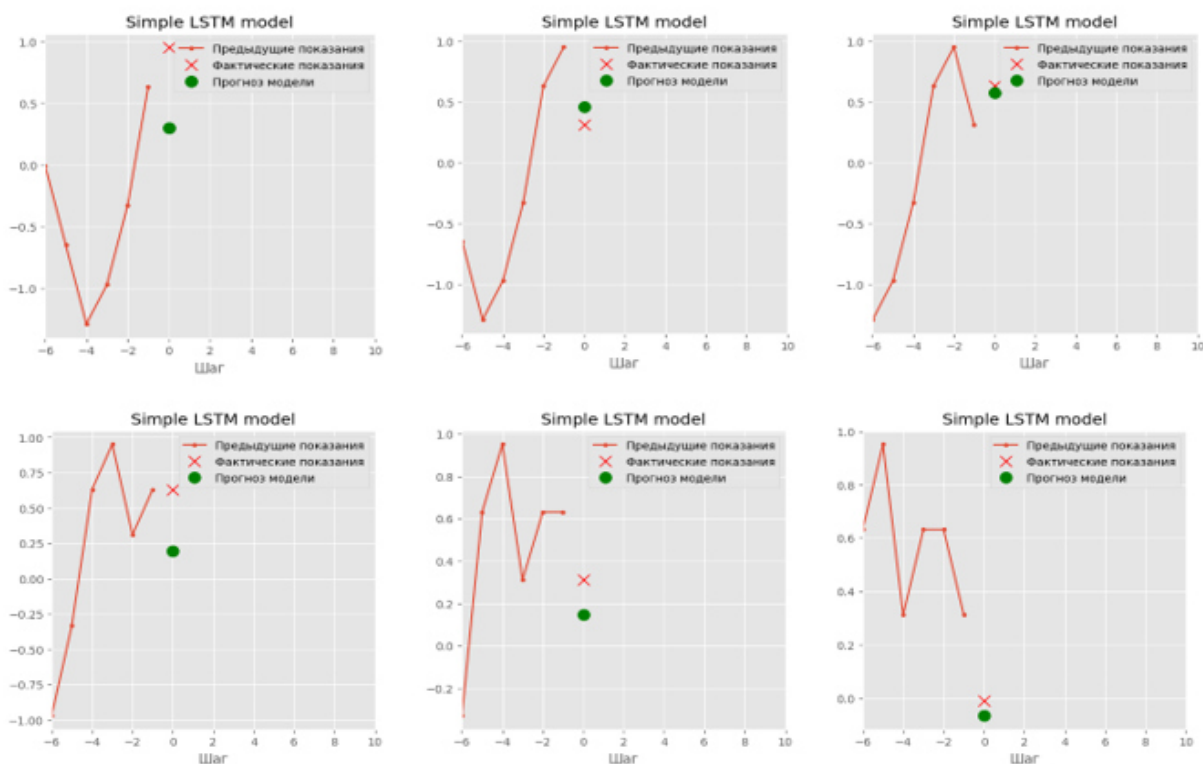


Рисунок 33 – Прогнозы

Организуем весь код для обучения массива нейронных сетей для прогнозирования расхода воды по одномерному временному ряду для прогноза на 1 месяц, для ускорения получения результатов и метрик обучения ограничимся 100 лицевыми счетами (рисунки 34-35):

```
for i in account_numbers[:100]:
    df_by_account = df_by_account_number.get_group((i, ))
    df_by_account = df_by_account[df_by_account.columns.drop(['HeatingCounterStatus', 'HeatingVolumeConsumed', 'IsResidential', 'AddressCode'])]

    df_t = df_by_account['WaterVolumeConsumed']
    print(i, df_t.shape)
    df_t.index = df_by_account['Period']
    df_t = df_t.values

    df_t_norm = df_t.copy()
    for j in reversed(range(1, df_t_norm.shape[0])):
        df_t_norm[j] = df_t_norm[j] - df_t_norm[j - 1]
    df_t_norm[0] = 0

    uni_train_mean = df_t_norm.mean()
    uni_train_std = df_t_norm.std()

    df_t_norm = (df_t_norm - uni_train_mean) / uni_train_std

    x_train_uni, y_train_uni = univariate_data(df_t_norm, 0, TRAIN_SPLIT,
                                                univariate_past_history,
                                                univariate_future_target)
    x_val_uni, y_val_uni = univariate_data(df_t_norm, TRAIN_SPLIT, None,
                                            univariate_past_history,
                                            univariate_future_target)

    train_univariate = tf.data.Dataset.from_tensor_slices((x_train_uni, y_train_uni))
    train_univariate = train_univariate.cache().shuffle(BUFFER_SIZE).batch(BATCH_SIZE).repeat()

    val_univariate = tf.data.Dataset.from_tensor_slices((x_val_uni, y_val_uni))
    val_univariate = val_univariate.batch(BATCH_SIZE).repeat()
```

Рисунок 34 – Блок обучения массива нейронных сетей (1 часть)

```
iter_count = 0;
stopwatch.start()
local_result = [];
while iter_count < 10:
    eps_add = 0.9 * ((iter_count + 1) / 10)
    simple_lstm_model = tf.keras.models.Sequential([
        tf.keras.layers.LSTM(1, input_shape=x_train_uni.shape[-2:]),
        tf.keras.layers.Dense(1)
    ])

    simple_lstm_model.compile(optimizer='adam', loss='mae', metrics=['accuracy', 'mean_squared_error', 'mean_absolute_error', 'r2_score'])
    simple_lstm_model.fit(train_univariate, epochs=EPOCHS,
                          steps_per_epoch=EVALUATION_INTERVAL,
                          validation_data=val_univariate, validation_steps=5,
                          callbacks=[tf.keras.callbacks.EarlyStopping(monitor="val_loss"), EarlyStoppingAtMinValLoss(min_loss=0.1+eps_add)])
    if (simple_lstm_model.metrics[0].result().numpy() < 0.1+eps_add): break
    iter_count += 1
    simple_lstm_model.save("/content/drive/MyDrive/Магистерская/models/{account_number}.keras".format(account_number=i));
stopwatch.stop()
results.loc[i, 'Elapsed Seconds'] = stopwatch.elapsed
results.loc[i, 'Global Iterations Count'] = iter_count
local_result.append(stopwatch.elapsed)
results.loc[i, simple_lstm_model.metrics[0].name] = simple_lstm_model.metrics[0].result().numpy()
for metric_result in simple_lstm_model.metrics[1].result():
    results.loc[i, metric_result] = simple_lstm_model.metrics[1].result()[metric_result].numpy()
stopwatch.reset()
```

Рисунок 35 – Блок обучения массива нейронных сетей (2 часть)

В качестве метрик возьмём значения функции потерь (loss), среднеквадратическую ошибку (MSE), среднюю абсолютную ошибку (MAE) и коэффициент детерминации (R^2). Метрики сформированы для каждого

лицевого, примененного в обучения, и собраны в .csv файл. Часть содержимого представлена в таблице 2:

Таблица 2 – Результаты обучения точечного прогнозирования

Account Number	Elapsed Seconds	loss	mse	mae	r2
701021163	175,3753	0,050424	0,003115	0,050424	-1,76041
701021167	41,62204	0,542569	0,453241	0,542569	-0,08154
701021174	72,99208	0,629497	0,923256	0,629497	0,076639
701021175	33,95675	0,782867	1,851112	0,782867	-0,43708
701021176	42,01731	1,891513	3,688522	1,891513	-50,9134
701021177	25,24174	0,570104	0,878225	0,570104	-0,04654
701021179	44,35128	0,481866	2,060411	0,481866	-0,13036
701021181	3,585719	0,027886	0,00112	0,027886	-0,8371
701021183	3,417298	0,014439	0,000334	0,014439	-0,48066
701021186	27,57453	0,602787	0,708668	0,602787	0,009701
701021163	147,0104	0,513574	2,529857	0,513574	-0,07153
701021167	26,94067	0,523973	0,886098	0,523973	-0,2978
701021174	71,72795	0,972363	3,422424	0,972363	-0,20869
701021175	33,74865	0,692051	0,500039	0,692051	-1,59055
701021176	34,45195	0,666058	0,822191	0,666058	-0,54559
701021177	3,001197	0,009997	0,000239	0,009997	-0,05166
701021179	76,74884	0,542726	1,682964	0,542726	-0,19752
701021181	28,73313	0,639425	0,667812	0,639425	-0,10772

Если рассмотреть результаты обучения, то:

- обучение каждой модели требует, в среднем, 40 – 60 секунд, что довольно длительно для сравнительно простых моделей и небольших объемов данных,

- критерий R^2 в большинстве своем является отрицательным, что может говорить о том, что прогнозы в большинстве своём случайны и могут

несколько некорректно выполнять прогноз (несмотря на достаточно малые значение MSE),

- метрики MSE и MAE довольно сильно отличаются для различных лицевых счетов: для одной части точность достаточно высокая, для других – слишком низкая.

Как итог: построение относительно корректных прогнозов возможно, но как минимум треть моделей будет давать погрешность.

Исходя из рассчитанного времени обучения, собранного в метриках, совокупное время обучения 100 моделей: 52 минуты, 56 секунд.

Для интервального прогнозирования несколько модифицируем функцию `univariate_data`, а также изменим конфигурацию нейронной сети (рисунки 36-37):

```
def univariate_data(dataset, start_index, end_index, history_size, target_size, single_step=True):
    data = []
    labels = []

    start_index = start_index + history_size
    if end_index is None:
        end_index = len(dataset) - target_size

    for i in range(start_index, end_index):
        indices = range(i-history_size, i)
        data.append(np.reshape(dataset[indices], (history_size, 1)))
        if single_step:
            labels.append(dataset[i+target_size])
        else:
            labels.append(dataset[i:i+target_size])
    return np.array(data), np.array(labels)
```

Рисунок 36 – функция `univariate_data` (модифицированная)

```
simple_lstm_model = tf.keras.models.Sequential([
    tf.keras.layers.LSTM(1, input_shape=x_train_uni.shape[-2:]),
    tf.keras.layers.Dense(3)
])

simple_lstm_model.compile(optimizer='adam', loss='mse', metrics=['accuracy', 'mean_squared_error', 'mean_absolute_error', 'r2_score'])
simple_lstm_model.fit(train_univariate, epochs=EPOCHS,
                    steps_per_epoch=EVALUATION_INTERVAL,
                    validation_data=val_univariate, validation_steps=5,
                    callbacks=[tf.keras.callbacks.EarlyStopping("val_accuracy"), EarlyStoppingAtMinValLoss(min_loss=0.1+eps_add)])
if (simple_lstm_model.metrics[0].result().numpy() < 0.1+eps_add): break
```

Рисунок 37 – правки конфигурации нейронной сети

В данную функцию был добавлен аргумент `single_step`, который указывает, берем ли мы прогнозируемое значение относительно входного

набора данных с некоторым шагом, либо берем интервал значений как прогнозируемые. Шаг задается аргументом target_size.

Рассмотрим результаты обучения интервального прогноза (таблица 3):

Таблица 3 – Результаты обучения интервального прогнозирования

AccNum	ES	loss	accuracy	mse	mae	r2
7010211	169,645	2,37454	0,3809523	2,3745400	0,4687107	-
63	5	0	88	91	21	0,0704871 79
7010211	21,0361	0,39233	0,6666666	0,3923364	0,4746979	-
67	2	6	87	58	77	0,0730772 84
7010211	57,8138	1,14735	0,3199999	1,1473557	0,7174554	-
74	1	5	93	95	47	0,2265847 92
7010211	57,9849	1,74270	0,2173912	1,7427034	0,8093231	-
75		3	97	38	32	0,4727151 69
7010211	57,0804	1,36054	0,3478260	1,3605487	1,0517994	-
76	2	8	93	35	17	2,2886779 31
7010211	48,9621	0,81833	0,3125	0,8183391	0,6084382	-
77	3	9		69	53	0,0196853 28
70	5	1	0,38	1,66	0,89	-
1021179	4,93943	,662017	8888896	2017465	1963184	0,0094585 61

Если рассмотреть результаты обучения, то:

- обучение каждой модели требует, в среднем, 30-40 секунд, что всё ещё довольно долго относительно точечного прогноза,

- критерий R^2 в большинстве своем является отрицательным, что может говорить о том, что прогнозы в большинстве своем случайны и могут несколько некорректно выполнять прогноз (несмотря на достаточно малые значение MSE),

- по метрикам MSE и MAE аналогично точечному прогнозу.

Как итог: построение относительно корректных прогнозов возможно, но как минимум треть моделей будет давать погрешность.

Исходя из рассчитанного времени обучения, собранного в метриках, совокупное время обучения 100 моделей: 1 час.

Далее приведен полный код переписанного блока обучения (рисунок 38):

```
x_train_uni, y_train_uni = univariate_data(df_t_norm, 0, TRAIN_SPLIT,
                                          univariate_past_history,
                                          univariate_future_target, single_step=False)
x_val_uni, y_val_uni = univariate_data(df_t_norm, TRAIN_SPLIT, None,
                                       univariate_past_history,
                                       univariate_future_target, single_step=False)

train_univariate = tf.data.Dataset.from_tensor_slices((x_train_uni, y_train_uni))
train_univariate = train_univariate.cache().shuffle(BUFFER_SIZE).batch(BATCH_SIZE).repeat()

val_univariate = tf.data.Dataset.from_tensor_slices((x_val_uni, y_val_uni))
val_univariate = val_univariate.batch(BATCH_SIZE).repeat()

iter_count = 0;
stopwatch.start()
local_result = [];
while iter_count < 10:
    eps_add = 0.9*((iter_count + 1)/10)
    simple_lstm_model = tf.keras.models.Sequential([
        tf.keras.layers.LSTM(1, input_shape=x_train_uni.shape[-2:]),
        tf.keras.layers.Dense(3)
    ])

    simple_lstm_model.compile(optimizer='adam', loss='mse', metrics=['accuracy', 'mean_squared_error', 'mean_absolute_error', 'r2_score'])
    simple_lstm_model.fit(train_univariate, epochs=EPOCHS,
                        steps_per_epoch=EVALUATION_INTERVAL,
                        validation_data=val_univariate, validation_steps=5,
                        callbacks=[tf.keras.callbacks.EarlyStopping("val_accuracy"), EarlyStoppingAtMinValLoss(min_loss=0.1+eps_add)])
    if (simple_lstm_model.metrics[0].result().numpy() < 0.1+eps_add): break
    iter_count += 1
simple_lstm_model.save("/content/drive/MyDrive/Мастерская/models/{account_number}_3.keras".format(account_number=i));
stopwatch.stop()
results.loc[i, 'Elapsed Seconds'] = stopwatch.elapsed
results.loc[i, 'Global Iterations Count'] = iter_count
local_result.append(stopwatch.elapsed)
results.loc[i, simple_lstm_model.metrics[0].name] = simple_lstm_model.metrics[0].result().numpy()
for metric_result in simple_lstm_model.metrics[1].result():
    results.loc[i, metric_result] = simple_lstm_model.metrics[1].result()[metric_result].numpy()
stopwatch.reset()
```

Рисунок 38 – Блок обучения массива нейронных сетей для интервального прогноза

6.3 Проектирования взаимодействия между клиентом и сервером

После обработки и очистки данных, а также обучения моделей, требуется разработать само приложение. Так как требуемая система

реализуется в виде двух отдельных модулей согласно клиент-серверной архитектуре, первым шагом в разработке является проектирование схемы их взаимодействия. Рассмотрим диаграмму последовательностей, описывающую взаимодействие пользователя и системой (рисунок 39):



Рисунок 39 – Диаграмма последовательности – взаимодействие с системой

По схеме видно, что изначально клиент направляет запрос к серверу с целью проверки работоспособности серверной части системы. В случае отсутствия подключения возможность отправки запроса блокируется, в ином случае со стороны сервера на клиент выгружается перечень доступных для прогнозирования лицевых счетов.

После успешного подключения к системе пользователь, указав все данные, может выполнить запрос. При получении запроса с клиента сервер подгружает в оперативную память модель, обученную под указанный лицевой счёт, и выполняет запрос. После выполнения расчёта сервер возвращает клиенту результаты прогноза.

Так как приложение будет реализовано в архитектуре «клиент-сервер», где клиент – веб-сайт, достаточно очевидно, что взаимодействие будет

проходить по протоколу HTTP, то есть с клиента уходит Request на endpoint сервера, сервер возвращает Response с данными либо ошибкой (например, 404 Not Found).

Для проверки работоспособности сервера добавим на его стороне endpoint /check, который просто возвращает 200 (ОК), в любом ином случае взаимодействие с клиентом будет ограничено. Опрос работоспособности сервера со стороны клиента будет проводиться каждые 5 секунд в асинхронном режиме.

Данные запроса прогноза отправляются на endpoint /prognose в следующем виде (описание полей приведено в таблице 4):

```
{  
  "accountNumber": "701021163",  
  "serviceType": 1,  
  "prognoseType": 2  
}
```

Таблица 4 – Описание полей запроса

Наименование поля	Тип поля	Описание поля
accountNumber	String	Номер лицевого счёта, по которому выполняется прогноз
serviceType	Integer	Тип услуги, для которой ведётся прогноз. Может принимать следующие значения: 0 – вода, 1 – отопление.

Продолжение таблицы 4

prognoseType	Integer	<p>Тип выполняемого прогноза. Может принимать следующие значения:</p> <p>0 - на следующий месяц относительно последнего месяца в базе,</p> <p>1 - на следующие 3 месяца,</p> <p>2 - на следующие полгода.</p>
--------------	---------	---

После выполнения прогноза пользователь может скачать результаты в виде json файла, либо в виде изображения.

Приведём диаграмму вариантов использования (use case), описывающую структуру работы клиента (рисунок 40):

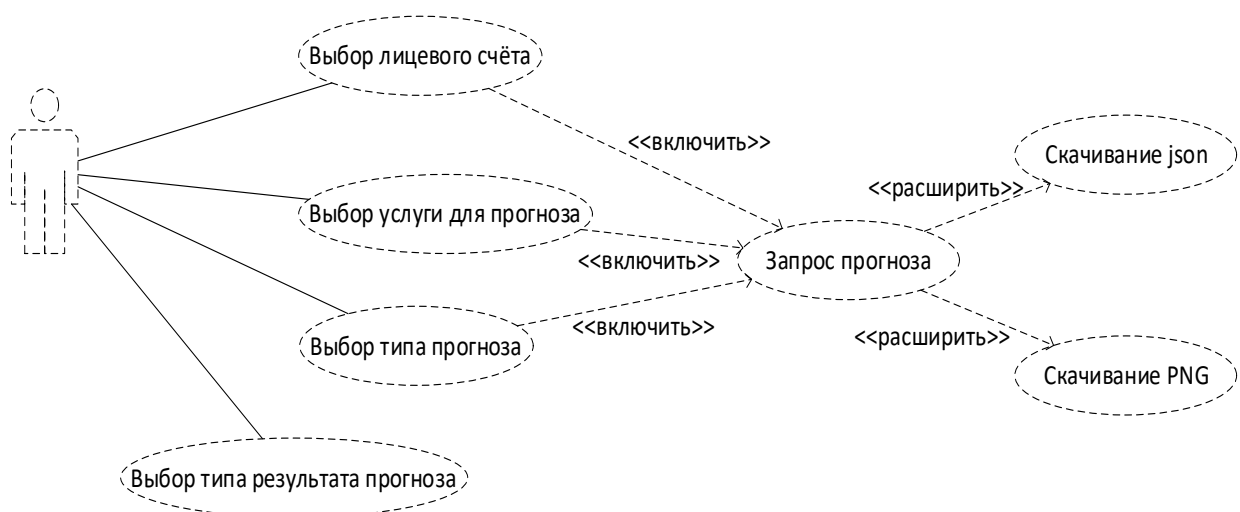


Рисунок 40 – Диаграмма вариантов использования работы пользователя с клиентом.

По схеме видно, что для запроса прогноза пользователь через клиент (веб-страницу) должен указать лицевой счёт, услугу для прогноза и срок прогноза. Выбор типа результата прогноза просто меняет отображения результатов прогноза (в частности сумм), не влияя на логику запроса самого прогноза. После выполнения прогноза можно скачать результаты в формате JSON либо график прогноза в формате PNG.

7 Тестирование системы

При открытии приложения нас встречает форма для ввода, в которой указываются номер лицевого счёта, по которому выполняется прогноз, услуга, по которой выполняется прогноз, тип прогноза и какое результирующее значение должно быть выведено в поле результатов прогноза. Внешний вид окна представлен на рисунке 41.

The screenshot shows a web form titled "Прогнозирование расхода коммунальных ресурсов". It contains the following fields and controls:

- A text input field labeled "Введите номер лицевого счёта:".
- A dropdown menu labeled "Выберите услугу для прогноза:" with "Холодная вода" selected.
- Two radio buttons for "Выберите результаты прогноза:": "Объём потребления" (selected) and "Сумма начислений".
- A dropdown menu labeled "Выберите тип прогноза:" with "Прогноз на следующий месяц" selected.
- A blue button labeled "Вывести прогноз".

Рисунок 41 – Исходное состояние интерфейса

Можно рассчитать расход по двум видам услуг: потреблению ресурса на отопление либо потреблению холодной воды. Из доступных вариантов прогнозов: прогноз на следующий месяц, прогноз на следующие 3 месяца, прогноз на следующие 6 месяцев. На рисунке 42 показана форма выбора прогноза.

This screenshot is identical to Figure 41, but the dropdown menu for "Выберите тип прогноза:" is open, showing three options:

- Прогноз на следующий месяц (highlighted)
- Прогноз на следующие 3 месяца
- Прогноз на следующие 6 месяцев

Рисунок 42 – Варианты прогноза

При нажатии на кнопку «Вывести прогноз» формируется график показаний, где синими точками указаны фактические значения, а красными – спрогнозированные (рисунки 43-44).

Прогнозирование расхода коммунальных ресурсов

Введите номер лицевого счёта:

Выберите услугу для прогноза:

Холодная вода

Выберите результаты

☒ Объём потребления

☐ Сумма начислений

прогноза:

Выберите тип прогноза:

Прогноз на следующий месяц

Вывести прогноз

Суммарный результат расчёта:

4.2 куб. м.

График прогноза



Скачать PNG

Скачать JSON

Рисунок 43 – Результаты прогноза (с суммарным потреблением в куб. метрах)

Прогнозирование расхода коммунальных ресурсов

Введите номер лицевого счёта:

Выберите услугу для прогноза:

Холодная вода

Выберите результаты



Объём потребления



Сумма начислений

прогноза:

Выберите тип прогноза:

Прогноз на следующий месяц

Вывести прогноз

Суммарный результат
расчёта:

140.2 руб.

График прогноза



Скачать PNG

Скачать JSON

Рисунок 44 – Результаты прогноза (с суммарным потреблением в рублях)

При необходимости можно сохранить результаты в виде JSON файла либо PNG-изображения (только график, рисунок 45)

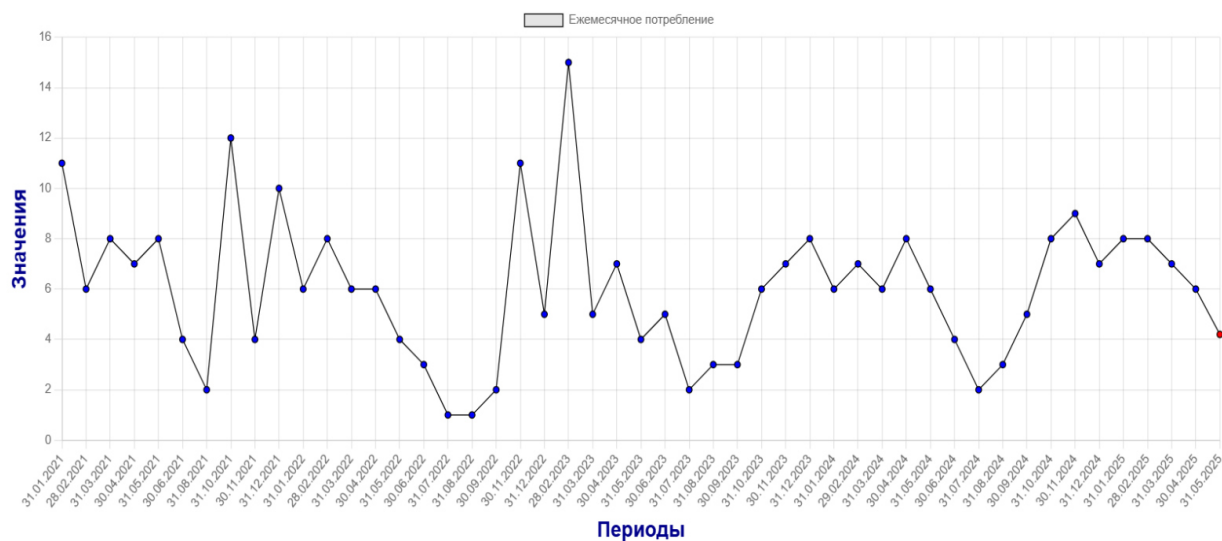


Рисунок 45 – Прогноз в виде PNG-изображения

```
{  
  "accountNumber": "701021163",  
  "coounterType": "HotWater",  
  "prognozeType": "ToNextMonth",  
  "results": [ ...  
],  
  "summaryVolume": "4.2 куб. м.",  
  "summaryAmount": "140.2 руб."
```

Рисунок 46 – Прогноз в виде JSON-файла

ЗАКЛЮЧЕНИЕ

В рамках данной магистерской диссертации были реализованы методы и средства прогнозирования расхода и оплаты коммунальных услуг. В качестве пользовательского интерфейса была разработана простая интерактивная веб-страница. В качестве метода прогнозирования были использованы модификации рекуррентных нейронных сетей – LSTM – модели. В качестве средств прогнозирования был использован массив искусственных нейронных сетей, каждая отдельная сеть которого была обучена на работу под отдельный лицевой счёт, отдельный интервал прогноза и тип услуги. Таким образом, можно считать, что задача, определенная в рамках магистерской диссертации, была выполнена.

К особенностям проекта можно отнести саму схему ее работы – для прогнозирования используется набор нейронных сетей, обученные под определенные входные данные.

Полученные результаты обладают научной ценностью. Ценность заключается в перспективе дальнейших исследований в области интеграции экспертных систем в область ЖКХ Российской Федерации.

Дальнейшее развитие разработанной системы может включать в себя интеграцию данной системы с реальной системой биллинга и расчёта коммунальных услуг.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Чучуева, И. А. Модель прогнозирования временных рядов по выборке максимального подобия, диссертация... канд. тех. наук / Московский государственный технический университет им. Н.Э. Баумана. Москва, 2012.
2. Лебедева И.М., Федорова А.Ю., Макроэкономическое планирование и прогнозирование / И.М. Лебедева, А.Ю. Федорова; под ред. А.Ю. Федоровой. – СПб: Университет ИТМО, 2016. – 54 с.
3. Жилищно-коммунальное хозяйство / [Электронный ресурс] // <https://ru.wikipedia.org/> : [сайт]. — URL: https://ru.wikipedia.org/wiki/%D0%96%D0%B8%D0%BB%D0%B8%D1%89%D0%BD%D0%BE-%D0%BA%D0%BE%D0%BC%D0%BC%D1%83%D0%BD%D0%B0%D0%BB%D1%8C%D0%BD%D0%BE%D0%B5_%D1%85%D0%BE%D0%B7%D1%8F%D0%B9%D1%81%D1%82%D0%B2%D0%BE (дата обращения: 05.07.2024).
4. Прогноз / [Электронный ресурс] // <https://ru.wikipedia.org/> : [сайт]. — URL: <https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%B3%D0%BD%D0%BE%D0%B7> (дата обращения: 05.07.2024).
5. JSON / [Электронный ресурс] // <https://ru.wikipedia.org/> : [сайт]. — URL: <https://ru.wikipedia.org/wiki/JSON> (дата обращения: 04.07.2024).
6. Комличенко, В. Н. Сравнительный анализ различных архитектур нейронных сетей для задач регрессии / В. Н. Комличенко, В. А. Федосенко, А. С. Купрейчик. — Текст : непосредственный // экономика и качество систем связи. — 2025. — № 35. — С. 110-120.
7. Постановление Правительства РФ от 06.05.2011 N 354 (ред. от 29.08.2024) «О предоставлении коммунальных услуг собственникам и пользователям помещений в многоквартирных домах и жилых домов» // СПС КонсультантПлюс.

8. LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., & Jackel, L. D. Backpropagation Applied to Handwritten Zip Code Recognition. [Текст] / LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., & Jackel, L. D. // Neural Computation. — 1989. — № 1(4). — С. 541-551.
9. Рекуррентная нейронная сеть / [Электронный ресурс] // Википедия : [сайт]. — URL: https://ru.wikipedia.org/wiki/%D0%94%D0%BE%D0%BB%D0%B3%D0%B0%D1%8F_%D0%BA%D1%80%D0%B0%D1%82%D0%BA%D0%BE%D1%81%D1%80%D0%BE%D1%87%D0%BD%D0%B0%D1%8F_%D0%BF%D0%B0%D0%BC%D1%8F%D1%82%D1%8C (дата обращения: 4.06.2025).
10. Keras: API высокого уровня для TensorFlow. / [Электронный ресурс] // TensorFlow : [сайт]. — URL: <https://www.tensorflow.org/guide/keras?hl=ru> (дата обращения: 10.06.2025).
11. Нескучный tutorial по NumPy / [Электронный ресурс] // Хабр : [сайт]. — URL: <https://habr.com/ru/articles/469355/> (дата обращения: 07.06.2025).
12. Visual Studio Code / [Электронный ресурс] // Википедия : [сайт]. — URL: https://ru.wikipedia.org/wiki/Visual_Studio_Code (дата обращения: 19.06.2025).

ПРИЛОЖЕНИЕ А

(обязательное)

**Разработка метода и средств прогнозирования расхода энергоносителя и
оплаты коммунальных услуг**

Листинг программы

ПРИЛОЖЕНИЕ А.1

Файл «index.html»

```
<!DOCTYPE html>
<head>
  <title>Прогнозирование расхода энергоресурсов</title>
  <link
href="node_modules/bootstrap/dist/css/bootstrap.min.css"
rel="stylesheet"/>
  <script
src="node_modules/jquery/dist/jquery.min.js"></script>
  <script
src="node_modules/bootstrap/dist/js/bootstrap.min.js"></script>
  <script
src="https://cdn.jsdelivrivr.net/npm/chart.js"></script>
  <script type="text/javascript" src="index.js"></script>
</head>
<body>
  <div class="container mt-5 card">
    <div class="row mt-2 mb-5">
      <h1 class="text-center align-middle">Прогнозирование
расхода коммунальных ресурсов</h1>
    </div>
    <div class="row">
      <div class="col-md-3">
        <label class="align-middle">Введите номер
лицевого счёта:</label>
      </div>
      <div class="col-md-3">
        <input class="form-control form-control-sm"
type="number"/>
      </div>
    </div>
    <div class="row mt-3">
      <div class="col-md-6">
        <div class="row">
          <div class="col-md-6">
            <label class="align-middle">Выберите
услугу для прогноза:</label>
          </div>
          <div class="col-md-6">
            <select class="form-select">
              <option selected value="0">Холодная
вода</option>
              <option value="1">Отопление</option>
            </select>
          </div>
        </div>
      </div>
      <div class="col-md-6">
        <div class="row">
          <div class="col-md-4">
```

```

        <label class="align-middle">Выберите
результаты прогноза:</label>
    </div>
    <div class="col-md-4 form-check">
        <input class="form-check-input"
type="radio" id="volumeResult" checked>
        <label class="form-check-label"
for="volumeResult">Объём потребления</label>
    </div>
    <div class="col-md-4 form-check">
        <input class="form-check-input"
type="radio" id="amountResult">
        <label class="form-check-label"
for="amountResult">Сумма начислений</label>
    </div>
</div>
</div>
<div class="row mt-3 mb-2">
    <div class="col-md-9">
        <div class="row">
            <div class="col-md-3">
                <label class="align-middle">Выберите тип
прогноза:</label>
            </div>
            <div class="col-md-9">
                <select class="form-select">
                    <option selected value="0">Прогноз
на следующий месяц</option>
                    <option value="1">Прогноз на
следующие 3 месяца</option>
                    <option value="2">Прогноз на
следующие 6 месяцев</option>
                </select>
            </div>
        </div>
        <div class="col-md-3">
            <button class="btn btn-primary w-100" id="show-
prognose">Вывести прогноз</button>
        </div>
    </div>
    <div id="prognoseResults" hidden>
        <div class="row mt-3 mb-2">
            <div class="col-md-9">
                <div class="row">
                    <div class="col-md-3">
                        <label class="align-
middle">Суммарный результат расчёта:</label>
                    </div>
                    <div class="col-md-3">
                        <input type="text" class="form-
control" disabled value="140.2 руб."/>

```

```

        </div>
    </div>
</div>
<div class="row mt-3 mb-2">
    <div class="row mt-1">
        <h1 class="align-middle text-center">График
прогноза</h1>
    </div>
    <div class="row mt-3 pr-3">
        <canvas id="prognoseChart" width="100%"
height="40%" style="border: 2px solid gainsboro; margin-left:
1%;">
            </canvas>
        </div>
    </div>
<div class="row mt-3 mb-2">
    <div class="row mt-1">
        <div class="col-md-6"></div>
        <div class="col-md-3">
            <button class="btn btn-primary w-
100">Скачать PNG</button>
        </div>
        <div class="col-md-3">
            <button class="btn btn-primary w-
100">Скачать JSON</button>
        </div>
    </div>
</div>
</div>
</body>
</html>

```

ПРИЛОЖЕНИЕ А.2

Файл «NetworkLearning.ipynb»

```
#Импорт библиотек и базовая конфигурация, а так же объявление
функций
pip install python_stopwatch
pip install openpyxl
import pandas as pd

import numpy as np
import seaborn as sns

import matplotlib.pyplot as plt
import matplotlib.mlab as mlab
import matplotlib
plt.style.use('ggplot')
from matplotlib.pyplot import figure

import tensorflow as tf
import time
from stopwatch import Stopwatch, profile

%matplotlib inline
matplotlib.rcParams['figure.figsize'] = (12,8)

pd.options.mode.chained_assignment = None
def univariate_data(dataset, start_index, end_index,
history_size, target_size, single_step=True):
    data = []
    labels = []

    start_index = start_index + history_size
    if end_index is None:
        end_index = len(dataset) - target_size

    for i in range(start_index, end_index):
        indices = range(i-history_size, i)
        data.append(np.reshape(dataset[indices], (history_size, 1)))
        if single_step:
            labels.append(dataset[i+target_size])
        else:
            labels.append(dataset[i:i+target_size])
    return np.array(data), np.array(labels)

#Загрузка и обработка данных
df =
pd.read_csv('/content/drive/MyDrive/Маристерская/dataset_volumeP
rogression.csv', delimiter=';', decimal=',')
df
df[df.isnull().any(axis=1)]
df[df.isnull().any(axis=1)].index
df.iloc[666828]
df.loc[df['AccountNumber'].isin([714039896])]
df.iloc[666828] = df.iloc[666828].fillna('ByCounter')
```

```

df.iloc[666828]
#Приведение категориальных признаков к числовым

from datetime import datetime
from sklearn.compose import make_column_selector as selector

categorical_columns_selector = selector(dtype_include=object)
categorical_columns = categorical_columns_selector(df)
print('Категориальные признаки:', categorical_columns)

numerical_columns_selector = selector(dtype_include=[int,
float])
numerical_columns = numerical_columns_selector(df)
print('Числовые признаки', numerical_columns)
categorical_columns.remove('Period')
numerical_columns.remove('AccountNumber')
numerical_columns.remove('AddressCode')
from sklearn.preprocessing import OrdinalEncoder

df_ord = df.iloc[:-1].copy()

encoder = OrdinalEncoder(categories = [['Yes', 'No']])
encoder = encoder.fit(df_ord[['IsResidential']])
df_ord[['IsResidential']] =
encoder.transform(df_ord[['IsResidential']])
print('IsResidential', encoder.categories_,
df_ord['IsResidential'].unique())

categories = [['NoCounter', 'ByCounter', 'ByMean',
'ByNormative']]
encoder = OrdinalEncoder(categories = categories)
encoder = encoder.fit(df_ord[['HeatingCounterStatus']])
df_ord[['HeatingCounterStatus']] =
encoder.transform(df_ord[['HeatingCounterStatus']])
print('HeatingCounterStatus', encoder.categories_,
df_ord['HeatingCounterStatus'].unique())

encoder = OrdinalEncoder(categories = categories)
encoder = encoder.fit(df_ord[['WaterCounterStatus']])
df_ord[['WaterCounterStatus']] =
encoder.transform(df_ord[['WaterCounterStatus']])
print('WaterCounterStatus', encoder.categories_,
df_ord['WaterCounterStatus'].unique())
df_ord
#Подготовка к обучению
df_by_account_number = df_ord.groupby(['AccountNumber'],
group_keys=True)[df.columns.drop('AccountNumber')]
account_numbers = df['AccountNumber'].drop_duplicates()
account_numbers
def univariate_data(dataset, start_index, end_index,
history_size, target_size, single_step=True):
    data = []
    labels = []

```

```

start_index = start_index + history_size
if end_index is None:
    end_index = len(dataset) - target_size

for i in range(start_index, end_index):
    indices = range(i-history_size, i)
    data.append(np.reshape(dataset[indices], (history_size, 1)))
    if single_step:
        labels.append(dataset[i+target_size])
    else:
        labels.append(dataset[i:i+target_size])
return np.array(data), np.array(labels)
TRAIN_SPLIT = 18
univariate_past_history = 6
univariate_future_target = 3

BATCH_SIZE = 64
BUFFER_SIZE = 100

EVALUATION_INTERVAL = 120
EPOCHS = 50
df_ord_counts_by_account_number =
df_ord.groupby(['AccountNumber'],
group_keys=True)['WaterVolumeConsumed'].count()
mask = df_ord_counts_by_account_number.where(lambda x: x >
40).isnull()
account_numbers = df_ord_counts_by_account_number[~mask].index
stopwatch = Stopwatch()
results = pd.DataFrame();
class EarlyStoppingAtMinValLoss(tf.keras.callbacks.Callback):
    def __init__(self, patience=0, min_loss=0):
        super().__init__()
        self.patience = patience
        self.min_loss = min_loss
        self.best_weights = None

    def on_train_begin(self, logs=None):
        self.wait = 0
        self.stopped_epoch = 0
        self.best = np.inf

    def on_epoch_end(self, epoch, logs=None):
        current = logs.get("val_loss")
        if np.less(current, self.best):
            self.best = current
            self.wait = 0
            self.best_weights = self.model.get_weights()
        else:
            self.wait += 1
            if self.wait >= self.patience:
                self.stopped_epoch = epoch
                self.model.stop_training = True

```

```

        self.model.set_weights(self.best_weights)

        if np.less(current, self.min_loss):
            self.stopped_epoch = epoch
            self.model.stop_training = True
            self.best = current
df_by_account = df_by_account_number.get_group((701021163, ))
df_by_account =
df_by_account[df_by_account.columns.drop(['HeatingCounterStatus'
, 'HeatingVolumeConsumed', 'IsResidential', 'AddressCode'])]
df_t = df_by_account['WaterVolumeConsumed']
df_t.index = df_by_account['Period']
df_t = df_t.values
df_t_norm = df_t.copy()
for j in reversed(range(1, df_t_norm.shape[0])):
    df_t_norm[j] = df_t_norm[j] - df_t_norm[j - 1]
df_t_norm[0] = 0
uni_train_mean = df_t_norm.mean()
uni_train_std = df_t_norm.std()
df_t_norm = (df_t_norm-uni_train_mean)/uni_train_std

past_history = 18
future_target = 3
STEP = 1

x_train_multi, y_train_multi = univariate_data(df_t_norm, 0,
TRAIN_SPLIT,

univariate_past_history,

univariate_future_target, single_step=False)
x_val_multi, y_val_multi = univariate_data(df_t_norm,
TRAIN_SPLIT, None,

univariate_past_history,
univariate_future_target,

single_step=False)
print ('Single window of past history :
{}'.format(x_train_multi[0].shape))
print ('\n Target temperature to predict :
{}'.format(y_train_multi[0].shape))
train_data_multi =
tf.data.Dataset.from_tensor_slices((x_train_multi,
y_train_multi))
train_data_multi =
train_data_multi.cache().shuffle(BUFFER_SIZE).batch(BATCH_SIZE).
repeat()

val_data_multi =
tf.data.Dataset.from_tensor_slices((x_val_multi, y_val_multi))
val_data_multi = val_data_multi.batch(BATCH_SIZE).repeat()
#Конфигурация ИНС и обучение
multi_step_model = tf.keras.models.Sequential([

```

```

        tf.keras.layers.LSTM(1, input_shape=x_val_multi.shape[-
2:]),
        tf.keras.layers.Dense(3)
    ])

multi_step_model.compile(optimizer=tf.keras.optimizers.RMSprop(c
lipvalue=1.0), loss='mse', metrics=['accuracy', 'r2_score'])
x_train_multi.shape[-1:]
for i in account_numbers[:100]:
    df_by_account = df_by_account_number.get_group((i, ))
    df_by_account =
df_by_account[df_by_account.columns.drop(['HeatingCounterStatus'
, 'HeatingVolumeConsumed', 'IsResidential', 'AddressCode'])]

    df_t = df_by_account['WaterVolumeConsumed']
    print(i, df_t.shape)
    df_t.index = df_by_account['Period']
    df_t = df_t.values

    df_t_norm = df_t.copy()
    for j in reversed(range(1, df_t_norm.shape[0])):
        df_t_norm[j] = df_t_norm[j] - df_t_norm[j - 1]
    df_t_norm[0] = 0

    uni_train_mean = df_t_norm.mean()
    uni_train_std = df_t_norm.std()

    df_t_norm = (df_t_norm-uni_train_mean)/uni_train_std

    x_train_uni, y_train_uni = univariate_data(df_t_norm, 0,
TRAIN_SPLIT,

univariate_past_history,

univariate_future_target, single_step=False)
    x_val_uni, y_val_uni = univariate_data(df_t_norm, TRAIN_SPLIT,
None,

univariate_past_history,
univariate_future_target,

single_step=False)

    train_univariate =
tf.data.Dataset.from_tensor_slices((x_train_uni, y_train_uni))
    train_univariate =
train_univariate.cache().shuffle(BUFFER_SIZE).batch(BATCH_SIZE).
repeat()

    val_univariate =
tf.data.Dataset.from_tensor_slices((x_val_uni, y_val_uni))
    val_univariate = val_univariate.batch(BATCH_SIZE).repeat()

    iter_count = 0;
    stopwatch.start()

```

```

local_result = [];
while iter_count < 10:
    eps_add = 0.9*((iter_count + 1)/10)
    simple_lstm_model = tf.keras.models.Sequential([
        tf.keras.layers.LSTM(1, input_shape=x_train_uni.shape[-
2:]),
        tf.keras.layers.Dense(3)
    ])

    simple_lstm_model.compile(optimizer='adam', loss='mse',
metrics=['accuracy', 'mean_squared_error',
'mean_absolute_error', 'r2_score'])
    simple_lstm_model.fit(train_univariate, epochs=EPOCHS,
                        steps_per_epoch=EVALUATION_INTERVAL,
                        validation_data=val_univariate,
validation_steps=5,

callbacks=[tf.keras.callbacks.EarlyStopping("val_accuracy"),
EarlyStoppingAtMinValLoss(min_loss=0.1+eps_add)])
    if (simple_lstm_model.metrics[0].result().numpy() <
0.1+eps_add): break
    iter_count += 1

simple_lstm_model.save("/content/drive/MyDrive/Магистерская/models/{account_number}_3.keras".format(account_number=i));
stopwatch.stop()
results.loc[i, 'Elapsed Seconds'] = stopwatch.elapsed
results.loc[i, 'Global Iterations Count'] = iter_count
local_result.append(stopwatch.elapsed)
results.loc[i, simple_lstm_model.metrics[0].name] =
simple_lstm_model.metrics[0].result().numpy()
for metric_result in simple_lstm_model.metrics[1].result():
    results.loc[i, metric_result] =
simple_lstm_model.metrics[1].result()[metric_result].numpy()
stopwatch.reset()
#Сохранение метрик обучения
results.to_excel('/content/drive/MyDrive/Магистерская/Статистика
по обучению_Обучение на 3 месяца.xlsx')

```

ПРИЛОЖЕНИЕ Б

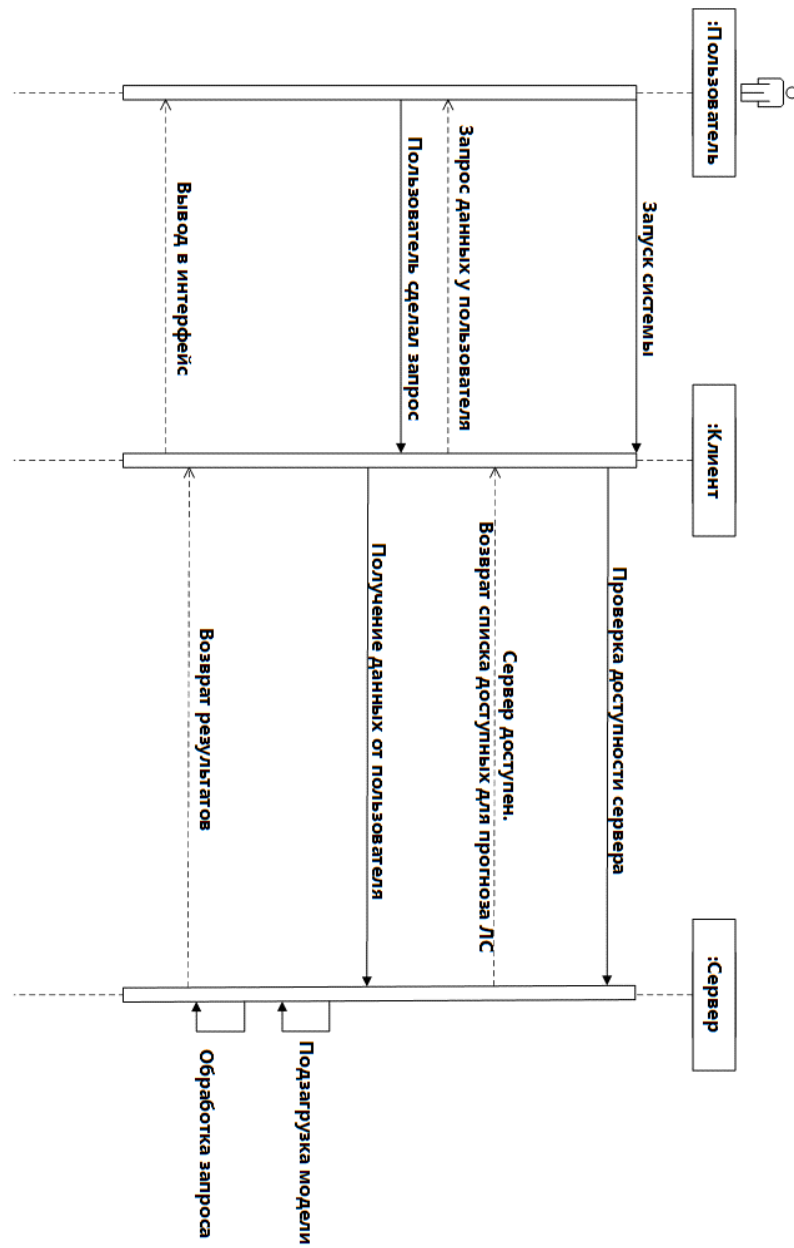
(обязательное)

**Разработка метода и средств прогнозирования расхода энергоносителя и
оплаты коммунальных услуг**

Диаграммы

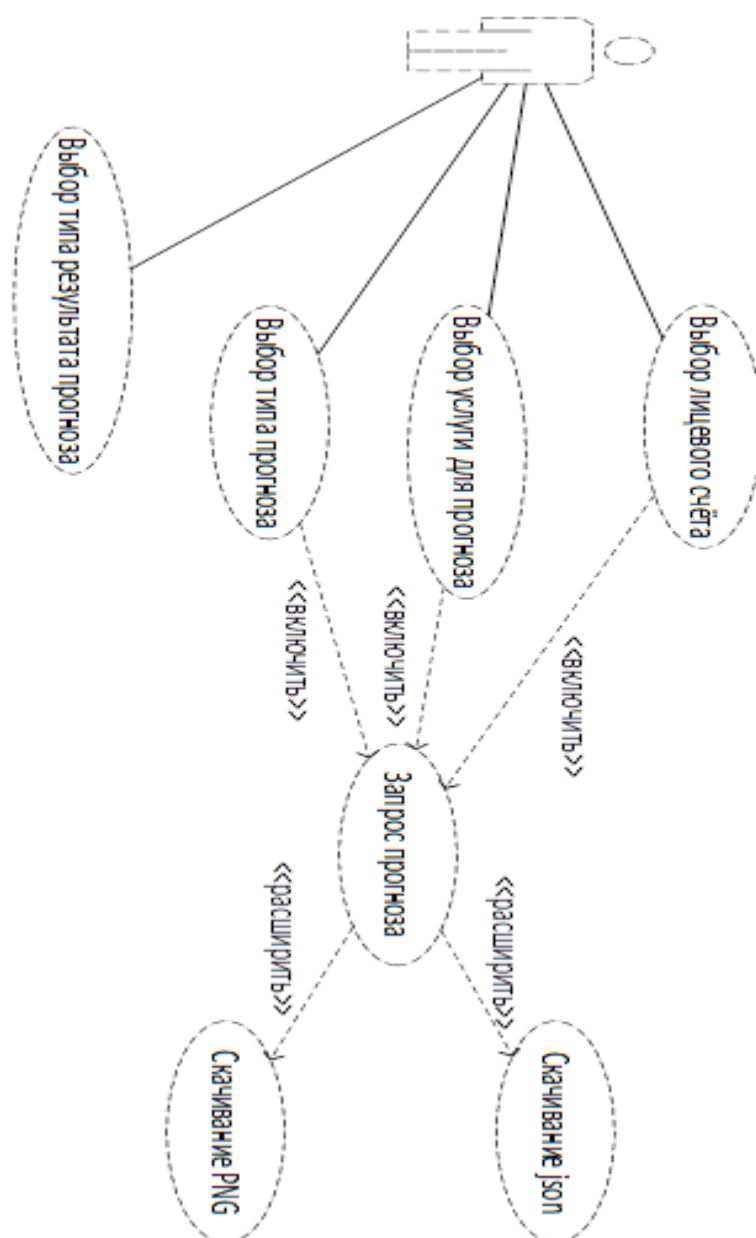
ПРИЛОЖЕНИЕ Б.1

Диаграмма последовательности – взаимодействие с системой



ПРИЛОЖЕНИЕ Б.2

Диаграмма прецедентов - работа пользователя с клиентом.



ПРИЛОЖЕНИЕ В

(обязательное)

**Разработка метода и средств прогнозирования расхода энергоносителя и
оплаты коммунальных услуг**

Презентация

РАЗРАБОТКА МЕТОДА И СРЕДСТВ ПРОГНОЗИРОВАНИЯ РАСХОДА ЭНЕРГОНОСИТЕЛЯ И ОПЛАТЫ КОММУНАЛЬНЫХ УСЛУГ

Автор работы: магистрант группы 23ВВИМ1 Попов А. Ф.,

Научный руководитель: д.т.н., профессор Финогеев А. Г.

Пенза, 2025

Цель и объект работы

Цель работы: разработать методы и модели, позволяющие спрогнозировать потребление энергоресурсов исходя из истории показаний собственников

Объект работы: система прогнозирования, реализующая следующие возможности:

- взаимодействие с пользователем посредством вебинтерфейса;
- возможность выполнить прогноз по лицевому счету из перечня доступных;
- возможность сохранения результатов прогноза.

Жилищно-коммунальное хозяйство

Жилищно-коммунальное хозяйство (ЖКХ) — отрасль экономики, обеспечивающая функционирование жилых зданий (домов), сетей снабжения водой, теплом и так далее, и сооружений, создающих безопасное, удобное и комфортабельное проживание и нахождение в них людей.



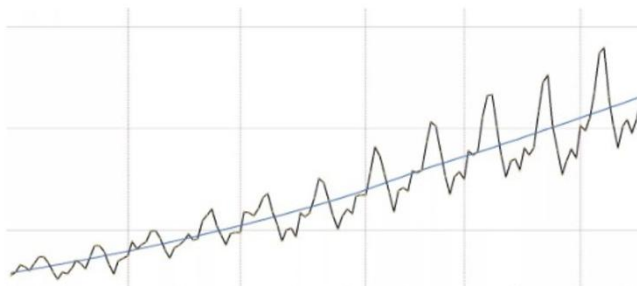
3

Прогноз

Прогно́з (от греч. πρόβωσις «предвидение, предсказание») — это научно обоснованное суждение о возможных состояниях объекта в будущем и (или) об альтернативных путях и сроках их осуществления. В узком смысле, это вероятностное суждение о будущем состоянии объекта исследования.

Существующие методы прогнозирования:

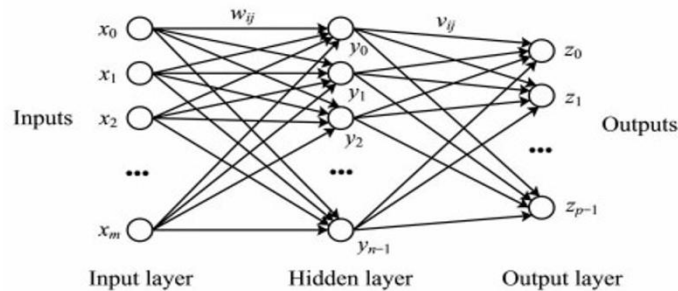
- фактологические;
- фактографические;
- интуитивные;
- экспертные;
- индивидуальные;
- коллективные;
- логические;
- математические



4

Искусственные нейронные сети

- Искусственные нейронные сети (ИНС) - это математические модели, созданные на основе биологических нейронных сетей, существующих в человеческом мозге. Они относятся к направлению искусственного интеллекта (ИИ)



5

Итоговая схема работы с системой



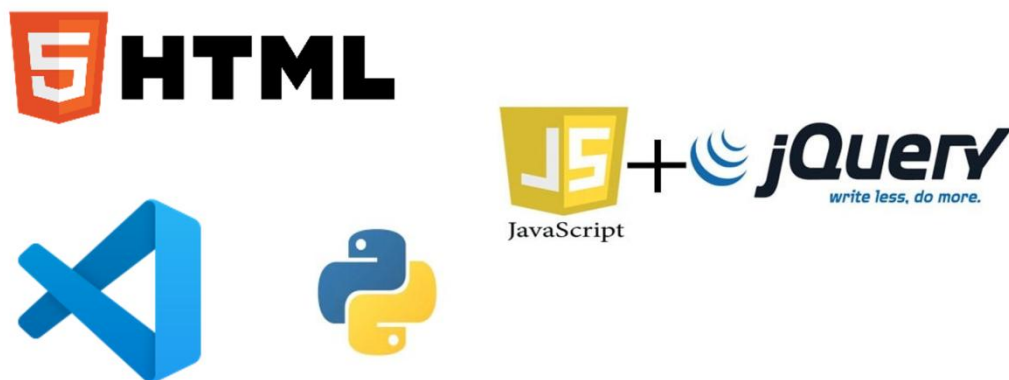
6

Стек технологий для разработки ИИС



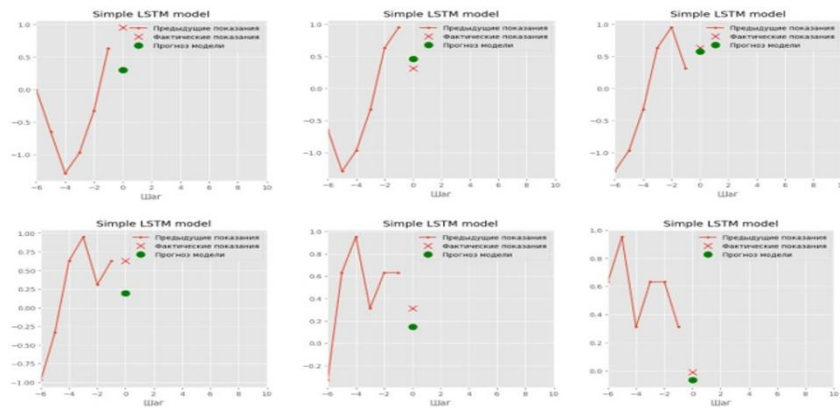
7

Стек технологий для клиент-серверного приложения



8

Результаты обучения



9

Клиент. Заполнение данных

Прогнозирование расхода коммунальных ресурсов

Введите номер лицевого счёта:

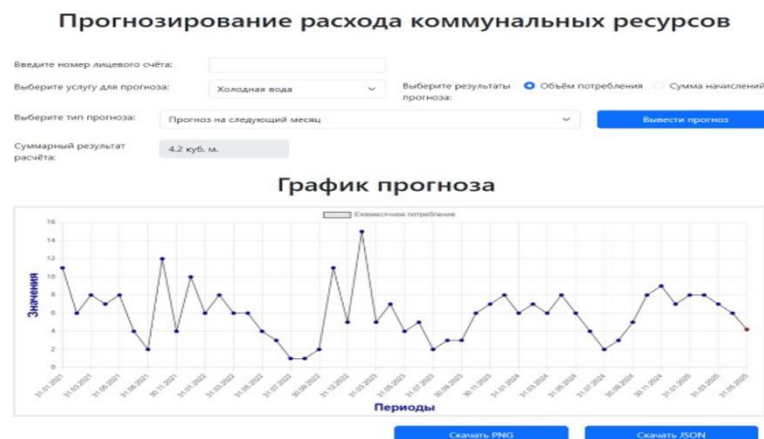
Выберите услугу для прогноза:

Выберите тип прогноза:

Выберите результаты прогноза: ☒ Объём потребления ☐ Сумма начислений

10

Клиент. Вывод прогноза



11

Заключение

В рамках данной магистерской диссертации были реализованы методы и средства прогнозирования расхода и оплаты коммунальных услуг. В качестве пользовательского интерфейса была разработана простая интерактивная веб-страница. В качестве метода прогнозирования были использованы модификации рекуррентных нейронных сетей LSTM – модели. В качестве средств прогнозирования был использован массив искусственных нейронных сетей, каждая отдельная сеть которого была обучена на работу под отдельный лицевой счёт, отдельный интервал прогноза и тип услуги. Таким образом, можно считать, что задача, определенная в рамках магистерской диссертации, была выполнена.

- К особенностям проекта можно отнести саму схему ее работы для прогнозирования используется набор нейронных сетей, обученные под определенные входные данные.
- Полученные результаты обладают научной ценностью. Ценность заключается в перспективе дальнейших исследований в области интеграции экспертных систем в область ЖКХ Российской Федерации.
- Дальнейшее развитие разработанной системы может включать в себя интеграцию данной системы с реальной системой биллинга и расчёта коммунальных услуг.

12

Спасибо за внимание!