МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ФАКУЛЬТЕТ ПРИКЛАДНОЙ МАТЕМАТИКИ И ИНФОРМАТИКИ

Кафедра компьютерных технологий и систем

ШАПЕЛЬ
Никита Ростиславович

АНАЛИЗ ТОЧНОСТИ ПРИМЕНЕНИЯ НЕЙРОННЫХ СЕТЕЙ В ЗАДАЧАХ ПРОГНОЗИРОВАНИЯ ВРЕМЕННЫХ РЯДОВ

Дипломная работа

Научный руководитель:
кандидат физико-математических наук,
доцент Е.С. Чеб

Допущена к защите

«__» _____2025 г.

Зав. кафедрой компьютерных технологий и систем
доктор педагогических наук, кандидат физико-математических наук, профессор В.В. Казаченок

Минск, 2025

РЕФЕРАТ

Дипломная работа, 47 страниц, 28 иллюстраций, 6 таблиц, 39 формул, 12 источников.

Ключевые слова: НЕЙРОННЫЕ СЕТИ, LSTM, GRU, ВРЕМЕННЫЕ РЯДЫ, ПРОГНОЗИРОВАНИЕ ВРЕМЕННЫХ РЯДОВ.

Объект исследования – модели прогнозирования временных рядов.

Предмет исследования – применение моделей прогнозирования к временным рядам с различной структурой.

Цель работы – анализ точности прогнозирования временных рядов с различной структурой с помощью алгоритмов и моделей, используемых для прогнозирования временных рядов, а также определение наиболее предпочтительной модели прогнозирования в зависимости от структуры исследуемого временного ряда.

Методы исследования – а) теоретические: изучение литературы, связанной с прогнозированием временных рядов и нейронными сетями; б) практические: генерация временных рядов со сложной структурой, прогнозирование сгенерированных временных рядов с помощью классических и нейросетевых моделей.

В процессе работы проведен анализ методов и программных систем прогнозирования временных рядов. Были рассмотрены различные алгоритмы и модели, используемые для прогнозирования временных рядов, включая классические модели, рекуррентные нейронные сети и сверточные нейронные сети. Для проведения исследования были сгенерированы временные ряды с нетипичной структурой, с целью выявления наиболее предпочтительного метода прогнозирования временных рядов с такой конфигурацией. Также были обучены нейросетевые модели и подобраны коэффициенты для традиционной модели авторегрессии, был проведен анализ результатов прогнозирования временных рядов с помощью рассматриваемых моделей, в зависимости от структуры прогнозируемого временного ряда. Было изучено влияние параметров временного ряда на качество прогноза каждой из рассматриваемых моделей. В результате исследования сделан вывод о том, что для прогнозирования временных рядов со сложной структурой лучше подходят нейросетевые модели, в частности рекуррентные нейронные сети.

РЭФЕРАТ

Дыпломная праца, 47 старонак, 28 ілюстрацый, 6 табліц, 39 формул, 12 крыніц.

Ключавыя словы: НЕЙРОНАВЫЯ СЕТКІ, LSTM, GRU, ЧАСОВЫЯ ШЭРАГІ, ПРАГНАЗАВАННЕ ЧАСОВЫХ ШЭРАГАЎ.

Аб'ект даследавання – мадэлі прагназавання часовых шэрагаў.

Прадмет даследавання – прымяненне мадэляў прагназавання да часовых шэрагаў з рознай структурай.

Мэта працы – аналіз дакладнасці прагназавання часовых шэрагаў з рознай структурай з дапамогай алгарытмаў і мадэляў, якія выкарыстоўваюцца для прагназавання часовых шэрагаў, а таксама вызначэнне найбольш пераважнай мадэлі прагназавання ў залежнасці ад структуры даследнага часовага шэрагу.

Метады даследавання – а) тэарэтычныя: вывучэнне літаратуры, звязанай з прагназаваннем часовых шэрагаў і нейронавымі сеткамі; б) практычныя: генерацыя часовых шэрагаў са складанай структурай, прагназаванне згенераваных часовых шэрагаў з дапамогай класічных і нейрасеткавых мадэляў.

У працэсе работы праведзены аналіз метадаў і праграмных сістэм прагназавання часовых шэрагаў. Былі разгледжаны розныя алгарытмы і мадэлі, якія выкарыстоўваюцца для прагназавання часовых шэрагаў, уключаючы класічныя мадэлі, рэкурэнтныя нейронавыя сеткі і канвалюцыйныя нейронавыя сеткі. Для правядзення даследавання былі згенераваныя часовыя шэрагі з нетыповай структурай, з мэтай выяўлення найбольш пераважнага метаду прагназавання часовых шэрагаў з такой канфігурацыяй. Таксама былі навучаны нейрасеткавыя мадэлі і падабраныя каэфіцыенты для традыцыйнай мадэлі аўтарэгрэсіі, быў праведзены аналіз вынікаў прагназавання часовых шэрагаў з дапамогай разгляданых мадэляў, у залежнасці ад структуры прагназуемага часовага шэрагу. Быў вывучаны ўплыў параметраў часовага шэрагу на якасць прагнозу кожнай з разгляданых мадэляў. У выніку даследавання зроблена выснова аб тым, што для прагназавання часовых шэрагаў са складанай структурай лепш падыходзяць нейрасеткавыя мадэлі, у прыватнасці рэкурэнтныя нейронавыя сеткі.

ABSTRACT

Diploma work, 47 pages, 28 illustrations, 6 tables, 39 formulas, 12 sources.

Keywords: NEURAL NETWORKS, TIME SERIES, LSTM, GRU, TIME SERIES FORECASTING.

The object of research is time series forecasting models.

The subject of the research is the application of forecasting models to time series with different structures.

The purpose of the work is to analyze the accuracy of forecasting time series with different structures using algorithms and models used to predict time series, as well as to determine the most preferred forecasting model depending on the structure of the time series under study.

The research methods are a) theoretical: the study of literature related to time series forecasting and neural networks; b) practical: the generation of time series with a complex structure, forecasting the generated time series using classical and neural network models.

In the process of work, the analysis of methods and software systems for forecasting time series was carried out. Various algorithms and models used to predict time series were considered, including classical models, recurrent neural networks, and convolutional neural networks. To conduct the study, time series with an atypical structure were generated in order to identify the most preferred method for predicting time series with this configuration. Neural network models were also trained and coefficients were selected for the traditional autoregressive model, and the results of time series forecasting were analyzed using the models under consideration, depending on the structure of the predicted time series. The influence of time series parameters on the forecast quality of each of the considered models was studied. As a result of the study, it was concluded that neural network models, in particular recurrent neural networks, are better suited for predicting time series with a complex structure.

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Прогнозирование временных рядов является очень активной темой исследований в области науки и техники [12]. По мере увеличения спроса на средства непрерывного мониторинга и сбора данных потребность в инструментах анализа временных рядов с использованием как статистических, так и машинных методов обучения постоянно увеличивается. В действительности, наиболее перспективные модели основаны на обеих технологиях [2, с. 25].

Одним из направлений, вызвавших особый интерес, стало использование нейронных сетей. Эти методы показали хорошие результаты при работе с зависимостями, которые трудно обнаружить с помощью более простых алгоритмов. Благодаря возможности обрабатывать большие объемы информации и выявлять полезные закономерности, нейронные сети активно применяются для анализа и прогноза временных рядов различной сложности.

Цель данной дипломной работы — рассмотреть, насколько точно нейронные сети справляются с задачей прогнозирования временных рядов. Основное сравнение проводится между результатами прогнозов, полученных с помощью традиционных моделей, таких как ARIMA и линейная регрессия, и нейросетевых моделей, представленных моделями LSTM и GRU.

Для оценки качества результатов прогнозирования используются числовые показатели, которые позволяют сравнивать точность прогнозов моделей между собой. В данном случае это среднеквадратическая ошибка и средняя абсолютная ошибка.

В первых двух главах дипломной работы освещается теоретическая часть, связанная с временными рядами, а также с моделями и инструментами для их прогнозирования. В третьей главе приводятся описание и полученные результаты проведенного экспериментального исследования. В частности, в данной главе приведены методы, использованные при прогнозировании сгенерированных временных рядов, а также представлены результаты оценки качества прогнозирования каждой из исследуемых моделей вместе с графическим изображением полученных прогнозов.

Актуальность данной работы связана с тем, что с каждым годом многие предприятия из различных отраслей сталкиваются с тем, что им приходится как можно быстрее реагировать на поведение финансового и потребительского рынков. В таких условиях правильно составленный прогноз может не только принести компании прибыль, но и помочь занять лидирующее место на потребительском рынке. Кроме того, внедрение хорошо обученных моделей прогнозирования позволяет компаниям заранее выявлять

тенденции и подстраиваться под изменения, вызванные различными факторами.

ГЛАВА 1. АНАЛИЗ МЕТОДОВ И ПРОГРАММНЫХ СРЕДСТВ ПРОГНОЗИРОВАНИЯ ВРЕМЕННЫХ РЯДОВ

1.1 Теория временных рядов

Под временным рядом понимаются статистические наблюдения над одним и тем же объектом в динамике, дополнительно упорядоченные во времени. Иначе говоря, временной ряд – это случайная функция

$$x = x(t) = x(\omega, t) \in R, \tag{1.1}$$

где t ∈ T ⊆ R – называется временем.[1]

Как правило, временные ряды состоят из различных компонент, которые могут характеризовать их поведение во времени. Посредством анализа данных компонент можно достоверно исследовать временные ряды и строить более точные модели для их прогнозирования. Чаще всего выделяют следующие компоненты:

- тренд;
- сезонность;
- шум;
- циклы.

Тренд. Тренды во временных рядах показывают общее направление, то есть долгосрочную тенденцию данных. Они могут указывать как на неизменность, так и на рост или снижение значений ряда на протяжении длительного временного отрезка.[1]

Чаще всего выделяются следующие виды трендов:

- линейный;
- параболический;
- экспоненциальный;
- гиперболический;
- логарифмический;
- логистический.[1]

Уравнение линейного тренда представимо в виде:

$$y_i = a + bt_i, \tag{1.2}$$

где y_i – уровень тренда в момент времени t_i;

a – свободный член уравнения, численно равный уровню ряда, принятого за начало отсчета;

b – средняя величина изменения уровней ряда за логарифм единицы времени.

Параболический тренд представим в виде:

$$y_i = a + bt_i + \mathrm{c}t_i{}^2, \tag{1.3}$$

где y_i – уровни тренда, на момент времени t_i;
a – свободный член уравнения, численно равный уровню ряда, принятого за начало отсчета;
b – величина изменения уровней ряда, которая изменяется равномерно со средним ускорением, равным $2\mathrm{c}$.[1]

Экспоненциальный тренд представим в виде:

$$y_i = ak^{t_i}, \tag{1.4}$$

где y_i – уровни тренда, на момент времени t_i;
a – свободный член уравнения, численно равный уровню ряда, принятого за начало отсчета;
k – темп изменения уровней.[1]

Уравнение гиперболического тренда имеет вид:

$$y_i = a + \frac{b}{t_i}, \tag{1.5}$$

где y_i – уровни тренда, на момент времени t_i,
a – число, к которому стремится тренд;
b – параметр гиперболы.[1]

Логарифмический тренд описывается уравнением вида:

$$y_i = a + b \ln t_i, \tag{1.6}$$

где y_i – уровни тренда, на момент времени t_i;
a – свободный член уравнения, численно равный уровню ряда, принятого за начало отсчета;
b – средняя величина изменения уровней ряда за логарифм единицы времени.

Наконец, логистический тренд можно представить в виде:

$$y_i = \frac{y_{max} - y_{min}}{e^{a_0 + a_1 t_i + 1}} + y_{min}\ , \tag{1.7}$$

где y_i – уровни тренда, на момент времени t_i;
y_{max} и y_{min} – верхняя и нижняя границы значений y_i;
a_0, a_1 – параметры логистического уравнения.[1]

Сезонность. Под сезонностью подразумеваются предсказуемые закономерности, которые повторяются регулярно. Как пример можно привести рост уровня продаж школьной формы перед началом учебного года или наборов конфет перед Новым годом. Сезонные компоненты демонстрируют колебания, фиксированные по направлению, величине и времени. [1]

Чаще всего сезонность разделяют на аддитивную и мультипликативную.

При аддитивной сезонности она включается в уравнение модели в качестве слагаемого, если же сезонность мультипликативная, то в уравнение она включается уже как множитель при тренде.[1]

Если сезонность аддитивная, то уравнение модели имеет вид:

$$Y = T + S + E, \tag{1.8}$$

где Y – прогнозируемая переменная;

T – уравнение тренда;

S – сезонные поправки;

E – случайные ошибки.[1]

Для мультипликативной сезонности уравнение модели имеет следующий вид:

$$Y = T * S + E, \tag{1.9}$$

где Y – прогнозируемая переменная,

T – уравнение тренда;

S – сезонные поправки;

E – случайные ошибки.[1]

Как правило, в случае мультипликативной сезонности уравнение модели записывается в виде:

$$Y = T * S * E, \tag{1.10}$$

При такой записи не составляет труда осуществить переход от уравнения с мультипликативной сезонностью к уравнению модели с аддитивной сезонностью, прологарифмировав прогнозируемую переменную.[1]

Шум. Шум является остаточной составляющей временных рядов, не поддающейся объяснению с помощью других компонент. Как правило, это могут быть случайные компоненты, затрудняющие исследование и прогнозирование временных рядов.

Цикл. Циклы представляют собой колебания, не имеющие фиксированной частоты во времени. Зачастую они обусловлены экономическими условиями и длятся не менее года, не имея постоянной продолжительности.

При анализе временных рядов необходимо учитывать структуру исследуемого ряда, так как алгоритмы исследования и прогнозирования отличаются в зависимости от того, является ли временной ряд стационарным или нет. К стационарным относятся временные ряды, у которых такие компоненты как дисперсия, математическое ожидание и ковариация не зависят от времени, то есть постоянны. Нестационарные временные ряды, соответственно, не обладают таким свойством, то есть их математическое ожидание, дисперсия и ковариация зачастую изменяются со временем.

1.2 Классические методы прогнозирования временных рядов

Авторегрессия. Авторегрессионные методы основываются на регрессионных методах прогнозирования временных рядов. На основании временного ряда составляется система, после оценивания параметров которой получается модель, имеющая вид [7]

$$x_{p+i} = \theta_0 + \theta_1 x_{p+i-1} + \cdots + \theta_p x_i. \quad (1.12)$$

Как правило, дополнительно на параметры θ_i могут накладываться такие условия, при которых временной ряд будет стационарным, то есть будет иметь постоянные математическое ожидание и дисперсию. В частности, если p = 1 то таким условием является $|\theta_1| < 1$. В общем же случае рассматривают следующий многочлен [7]

$$1 - \theta_1 z - \cdots - \theta_p z^p, \quad (1.13)$$

временной ряд будет являтся стационарным, если выполняется условие, что $|z_i| < 1$, i = $\overline{1, p}$, где $z_i -$ корни этого многочлена..[7]

Модель скользящего среднего. Данная модель представима следующей формулой:

$$x_{q+t} = c + \varepsilon_{q+t} + \theta_1 \varepsilon_{q+t-1} + \cdots + \theta_q \varepsilon_t, \text{ t} = \overline{1, n-q}. \quad (1.14)$$

В этом случае вектор $\{\varepsilon_i\}$ будет рассматриваться как ошибки при прогнозировании предыдущих значений временного ряда, используемые при прогнозировании будущих значений.

Также имеется возможность перейти от авторегрессионной модели к модели скользящего среднего.[7]

Модель ARIMA. Модель ARIMA объединяет в себе сразу 2 модели: скользящего среднего и авторегрессионную. В ARIMA также используются разности временных рядов и в общем виде эта модель может быть представлена при помощи следующей формулы:

$$x'_t = c + \varphi_1 x'_{t-1} + \cdots + \varphi_p x'_{t-p} + \theta_1 \varepsilon_{t-1} + \cdots + \theta_q \varepsilon_{t-q} + \varepsilon_1, \quad (1.15)$$

где $x'_t = x_t - x_{t-1}$.[7]

Можно также заметить, что часто в ARIMA вместо вышеупомянутых разностей используются разности второго порядка исходного временного ряда. Так, в моделях, представленных в библиотеках языка Python, используется именно такое представление. Зачастую, такие модели обозначают как ARIMA(p, d, q), где p – параметр авторегрессии, d – порядок разности, а q – параметр модели скользящего среднего. К таким моделям также предъявляются требования стационарности, относящиеся к параметрам авторегрессии, а также условия применимости преобразований, касающиеся модели скользящего среднего.[7]

Модель ARIMA позволяет учитывать такие характеристики временных рядов, как тренды и сезонность, а также линейные зависимости данных. Данная модель очень распространена при прогнозировании временных рядов в различных областях.[7]

Внешние параметры модели ARIMA могут быть подобраны, например, при помощи алгоритма Гайндмана-Хандакара.[7]

Экспоненциальное сглаживание. Простое экспоненциальное сглаживание зачастую применяется к временным рядам, у которых отсутствуют такие компоненты как тренд и сезонность. Экспоненциальное сглаживание может быть выполнено по следующей формуле:

$$X_{T+1|T} = ax_T + a(1-a)x_{T-1} + \cdots + a(1-a)^k x_{T-k} + \cdots, \quad (1.16)$$

где $a \in (0,1)$ – параметр сглаживания.

Сумма всех коэффициентов в формуле равна единице, а начиная с некоторого k коэффициенты при предыдущих значениях временного ряда становятся пренебрежимо малыми. Соответственно, чем больше значение a,

тем меньше слагаемых из формулы необходимо использовать для получения прогноза с заданной точностью.[7]

Применяя экспоненциальное сглаживание можно получить новый временной ряд, представляющий собой сглаженный исходный ряд.[7]

Для подбора оптимального значения параметра сглаживания α используются различные методы, так, например, можно использовать метод наименьших квадратов или анализ ошибок прогноза. Также существуют модификации простого экспоненциального сглаживания, такие как двойное экспоненциальное сглаживание и тройное экспоненциальное сглаживание, учитывающие тренды и сезонность временных рядов.

Модель Хольта с линейным трендом. Данная модель позволяет использовать экспоненциальное сглаживание для прогнозирования временных рядов, у которых есть тренд. Такая модель характеризуется следующими уравнениями:

$$\begin{cases} x_{t+h|t} = l_t + hb_t, \\ l_t = ax_t + (1-a)(l_{t-1} + b_{t-1}), \\ b_t = \beta(l_t - l_{t-1}) + (1-\beta)b_{t-1}, \end{cases} \quad (1.17)$$

где l_t – оценка уровня в момент времени t для рассматриваемого временного ряда;
b_t – оценка тренда в момент t;
$a \in (0,1)$ – параметр сглаживания для уровня;
$\beta \in (0,1)$ – параметр сглаживания для тренда. [7]

Прогнозируемые значения, получаемые с использованием данной модели, являются линейной функцией от h. Но модель Хольта с линейным трендом не может использоваться для слишком больших значений h, так как часто линейный тренд прогнозных значений не сохраняется в долгосрочной перспективе. Для устранения данного недостатка часто используют модель Хольта с демпфированным трендом. [7]

1.3 Современные методы прогнозирования временных рядов

Современные методы прогнозирования временных рядов включают различные алгоритмы и модели, которые способны учитывать более сложные структуры и особенности временных рядов.

SARIMA. Сезонная авторегрессионная интегрированная скользящая средняя, SARIMA или Сезонная ARIMA, является расширением ARIMA,

которое явно поддерживает одномерные данные временных рядов с сезонным компонентом.

Она добавляет три новых гиперпараметра для определения авторегрессии (AR), разности (I) и скользящего среднего (MA) для сезонной составляющей ряда, а также дополнительный параметр для периода сезонности.

Сезонная часть модели состоит из компонент, которые аналогичны несезонным компонентам модели, но включают обратные сдвиги сезонного периода.

Сезонный компонент моделей SARIMA добавляет следующие три компонента: сезонную авторегрессию, сезонный интеграл и сезонную скользящую среднюю.

Сезонная авторегрессия отражает взаимосвязь между текущим значением ряда и его прошлыми значениями, особенно при сезонных задержках.

Аналогично несезонным разностям, сезонный интеграл учитывает разницу, необходимую для устранения сезонности в ряду.

Сезонная скользящая средняя моделирует зависимость между текущим значением и остаточными ошибками предыдущих прогнозов при сезонных задержках.

Градиентный бустинг. Градиентный бустинг – это метод группового машинного обучения, суть которого заключается в том, чтобы последовательно объединять прогнозы нескольких слабых обучаемых, которыми, как правило, выступают деревья решений. Он направлен на повышение общей эффективности прогнозирования путем оптимизации весовых коэффициентов модели на основе ошибок предыдущих итераций, постепенного уменьшения ошибок прогнозирования и повышения точности модели.

Если рассматривать градиентный бустинг как алгоритм, то у него можно выделить следующие этапы:

- Первым делом выбирается базовая модель. Базовую модель еще называют слабой, а используется она в основном для того, чтобы получить первичный прогноз. Часто в качестве базовой модели используется дерево решений. Сутью дальнейших этапов алгоритма, собственно говоря, является улучшение полученного первичного прогноза.
- На втором этапе подбирается функция, задача которой – вычисление разности между прогнозными и реальными значениями. В задачах регрессии используется среднеквадратичная ошибка, а в задачах классификации, например, – логарифмическая функция потерь.

- Далее необходимо найти градиент функции потерь. С его помощью определяется направление наибольшего убывания функции потерь. На каждом шаге градиентного бустинга задача заключается в том, чтобы модель приблизилась к минимуму функции потерь, двигаясь в направлении, противоположному градиенту.
- Для того, чтобы исправить ошибки и улучшить базовую модель, к ней добавляется еще одна. Основная задача новой модели – минимизация ошибок прогнозирования, оставшихся после предыдущих моделей. Данный процесс повторяется несколько раз, пока не будет достигнута заданная точность или не будет выполнено определенное количество итераций цикла.
- На финальном этапе алгоритма построенные модели объединяются в ансамбль, каждая со своим весом. Прогнозные значения получаются путем комбинирования предсказаний всех моделей с соответствующими весами.

Если рассматривать реализации градиентного бустинга, то наиболее популярными являются такие модели как XGBoost, LightGBM и CatBoost. Данные модели содержат различные оптимизации и улучшения, которые улучшают производительность и удобство использования.

1.4 Программные системы прогнозирования временных рядов

Существует большое количество программных систем для прогнозирования временных рядов, среди которых есть как языки программирования, так и специализированные программные пакеты. В рамках данной дипломной работы рассматриваются наиболее распространенные программные системы для прогнозирования временных рядов. В данном разделе будут рассмотрены четыре языка программирования, которые являются наиболее распространенными и эффективными в области анализа и прогнозирования временных рядов.

Язык Python. Python – это интерпретируемый, объектно-ориентированный язык программирования, который широко используется, в том числе, в области прогнозирования временных рядов. Благодаря своему простому и легко воспринимаемому синтаксису Python является одним из самых популярных языков программирования в мире. Интерпретатор Python, а также его обширная стандартная библиотека, доступны в исходном коде бесплатно для всех платформ, более того, они могут свободно распространяться. Не удивительно, что в Python существуют и библиотеки для анализа и прогнозирования временных рядов.

Pandas – это, наверное, самая популярная библиотека в Python, широко используемая для работы с большими наборами данных. Она предоставляет различные функции индексирования временных рядов, повторной выборки и выравнивания данных. Pandas также предлагает удобные методы для обработки недостающих данных, агрегирования данных и расчетов на основании времени.[10, с. 3]

Еще одной широко используемой библиотекой в Python является NumPy. Данная библиотека предоставляет объекты многомерных массивов, матрицы и многое другое. Библиотека включает в себя широкий набор функций для манипуляций с массивами: математические и логические вычисления, сортировки, выборки, изменение формы массивов, дискретные преобразования Фурье, линейная алгебра, генерация случайных чисел.

Очень полезным в прикладных задачах прогнозирования временных рядов является модуль Statsmodels, предоставляющий классы и функции для оценки множества различных статистических моделей, а также для проведения статистических тестов и анализа данных. Данная библиотека содержит большое количество моделей, включая модели временных рядов, такие как ARIMA и SARIMA. В Statsmodels также имеются инструменты для оценки точности моделей, тестирования гипотез, и диагностической проверки.[10, с. 4]

Язык R. R – это одновременно и среда, и язык, который предназначен для статистических вычислений и графики. В нем представлены коллекции пакетов, которые были специально разработаны для манипуляций с данными временных рядов. В языке R выделяют следующие наиболее используемые пакеты: Forecast, Tseries, Zoo.

Пакет Forecast предоставляет функции и методы для анализа и отображения прогнозов временных рядов, включая автоматическое моделирование ARIMA, а также экспоненциальное сглаживание с помощью моделей пространства состояний.

Также как и Forecast, пакет tseries обладает набором инструментов для анализа и прогнозирования временных рядов. В этом пакете собраны такие функции, как тесты на сезонность, статистические тесты. Tseries также предоставляет функции для декомпозиции временных рядов и обнаружения тренда.

Zoo в первую очередь предназначен для анализа данных временных рядов с неравномерными интервалами. С помощью инструментов, собранных в данном пакете, можно манипулировать временными рядами, создавать их подмножества и объединять в один ряд. Zoo также включает в себя функции для выполнения агрегирования данных за нерегулярные промежутки времени и обработки пропущенных значений.

Язык MATLAB. MATLAB является незаменимым инструментом при проведении научных исследований или инженерных расчетов, а не просто высокоуровневым языком программирования. В MATLAB также имеется пакет программ и интегрированная среда для разработки, выполнения инженерных и математических расчетов, а также работы с матричными базами данных и визуализации.

В MATLAB присутствуют специальные структуры данных, которые способствуют эффективной и точной работе с временными рядами. Такие объекты позволяют легко манипулировать данными временных рядов, визуализировать и индексировать их.

В MATLAB есть набор инструментов для обработки сигналов, который содержит функции для различных задач анализа временных рядов, включая такие, как спектральный анализ, фильтрация и частотно-временной анализ. С помощью MATLAB также можно выполнять анализ Фурье и цифровую фильтрацию временных рядов.

Также необходимо отметить набор инструментов для эконометрики, содержащий большое количество функций и моделей для эконометрического анализа временных рядов. Этот набор включает в себя функции для оценки и прогнозирования временных рядов, выполнения тестов на единичный корень, анализа данных и реализации многомерного анализа временных рядов.

Что касается финансовых временных рядов, то MATLAB предоставляет функции и модели, специально разработанные для анализа временных рядов такого типа. Он содержит функции для анализа и моделирования данных финансового рынка, оптимизации портфеля и расчета показателей риска, которые приходятся очень кстати при прогнозировании финансовых временных рядов. Вообще говоря, финансовый инструментарий является одним из самых полезных и наиболее часто используемых пакетов для работы с временными рядами.

Wolfram Mathematica. Можно сказать, что Wolfram Language является уникальным проблемно-ориентированным языком программирования. В нем имеется большое количество встроенных функций, которые охватывают практически все области знания. Также Wolfram Mathematica поддерживает интеграцию с другими языками программирования, такими как Java, C, MATLAB, R, что позволяет использовать функции из различных областей знаний для решения комплексных задач. Необходимо заметить, что Wolfram Mathematica предлагает большой набор инструментов и для работы с временными рядами.

В модуле Wolfram Mathematica Time Series представлены функции для анализа и прогнозирования временных рядов, которые значительно упрощают процесс прогнозирования. Time Series также помогает оценить адекватность и

применимость выбранной модели с помощью различных статических процедур. Исследуемый пакет позволяет применять математические преобразования для подготовки данных к моделированию, чтобы представлять временные ряды в удобной форме для последующего анализа. Модуль Time Series дает возможность исследовать временные ряды посредством построения графиков автокорреляции и частичной автокорреляции, не говоря уже о визуализации самих временных рядов. Также модулем поддерживается анализ в частотной и временной областях, включая спектральный анализ на основе преобразований Фурье.

1.5 Оценки качества прогнозирования временных рядов

При прогнозировании временных рядов, решающую роль в качестве прогноза, безусловно, играет выбранная модель и правильность ее обучения. Однако, для того, чтобы правильно выбрать адекватную и подходящую для прогнозирования модель из имеющихся, необходимо оценить качество прогноза каждой из них и выбрать лучшую. С данной задачей помогают справится оценки качества прогнозирования.

В данном разделе как раз рассматриваются наиболее распространенные метрики и подходы, позволяющие оценить точность и стабильность прогнозов временных рядов, при выборе модели для прогнозирования.

Mean Absolute Error. Средняя абсолютная ошибка является одним из самых, если не самым простым и интуитивно понятным показателем оценки для модели прогнозирования временных рядов. Она измеряет среднюю величину абсолютных ошибок между фактическими и прогнозируемыми значениями [11, с. 29]. MAE рассчитывается по следующей формуле:

$$MAE = \frac{1}{n}\sum\left|y_{true} - y_{pred}\right|, \tag{1.18}$$

где n – число наблюдений;
y_{true} – фактическое значение;
y_{pred} – прогнозируемое значение.

MAE легко интерпретируема, поскольку она показывает, насколько в среднем полученные прогнозы отличаются от фактических значений исследуемого ряда. Однако стоит учитывать тот факт, что MAE нечувствительна к выбросам и не учитывает масштаб данных.[11, с. 29]

Root Mean Squared Error. Среднеквадратичная ошибка также широко используется при оценке качества прогнозирования значений временного ряда. Она измеряет среднюю разницу между фактическими значениями ряда

и прогнозируемыми значениями. Эта средняя разница возводится в квадрат, дабы избежать игнорирования положительных и отрицательных значений при их суммировании. RMSE рассчитывается как:

$$RMSE = \sqrt{\frac{1}{n}\sum(y_{true} - y_{pred})^2}, \quad (1.19)$$

где n – число наблюдений;
y_{true} – фактическое значение;
y_{pred} – прогнозируемое значение.

RMSE похожа на MAE, но она придает больший вес большим ошибкам, поскольку они возводятся в квадрат перед усреднением. Это необходимо учитывать при выборе оценки качества прогнозирования. Также RMSE более чувствительна к выбросам и вариациям в данных, чем MAE.

Mean Absolute Percentage Error. MAPE — еще один из наиболее часто используемых показателей для оценки точности прогностических моделей. Данный показатель представляет собой процент средней абсолютной ошибки и показывает, насколько в среднем прогнозы отклоняются от реальных значений. MAPE позволяет определить среднюю относительную ошибку модели, выраженную в процентах, что может облегчить интерпретацию результатов [11, с. 29]. MAPE можно найти по следующей формуле:

$$MAPE = \frac{100}{n}\sum\left|\frac{y_{true} - y_{pred}}{y_{true}}\right|, \quad (1.20)$$

где n – число наблюдений;
y_{true} – фактическое значение;
y_{pred} – прогнозируемое значение.

MAPE зачастую полезна при сравнении производительности моделей, которые имеют разные масштабы или единицы измерения. Также MAPE легко интерпретируема, так как она показывает, насколько прогнозы отличаются от фактических значений в процентном выражении. Однако MAPE имеет и ограничения, такие как, например, неопределенность, когда фактическое значение равно нулю или чувствительность к выбросам и сезонности.

Symmetric Mean Absolute Percentage Error. Симметричная средняя абсолютная процентная ошибка является модифицированной версией MAPE, которая устраняет некоторые из ее недостатков. SMAPE измеряет средний процент абсолютных ошибок между реальными и прогнозируемыми значениями относительно среднего значения фактических и прогнозируемых значений. SMAPE можно рассчитать по формуле:

$$SMAPE = \frac{100}{n}\sum\left|\frac{y_{true} - y_{pred}}{(y_{true} - y_{pred})/2}\right|, \quad (1.21)$$

где n – число наблюдений;
y_{true} – фактическое значение;
y_{pred} – прогнозируемое значение.

SMAPE отличается тем, что одинаково учитывает как переоценку, так и недооценку прогнозов. В отличие от MAPE, она остается определённой даже в случаях, когда фактическое значение равно нулю, если предсказанное значение не равно нулю. Однако SMAPE не лишена недостатков. Например, для нее свойственна склонность к высоким значениям, чувствительность к выбросам и сезонности, а также максимальное значение в 200%.

Mean Absolute Scaled Error. Средняя абсолютная масштабированная ошибка – это относительный оценочный показатель для модели прогнозирования временных рядов, который сравнивает производительность модели с исходными данными [11, с. 30]. Она измеряет среднюю величину абсолютных ошибок между фактическими и прогнозируемыми значениями, умноженную на среднюю величину абсолютных ошибок наивного прогноза. MASE рассчитывается следующим образом:

$$MASE = \frac{\frac{1}{n}\sum|y_{true}-y_{pred}|}{1/(n-1)}\sum\left|y_{true} - y_{true\_lag1}\right|, \quad (1.22)$$

где n – число наблюдений;
y_{true} – фактическое значение;
y_{pred} – прогнозируемое значение;
y_{true_lag1} – фактическое значение на предыдущем шаге.

MASE полезна, когда необходимо оценить, насколько модель лучше, чем простое предположение. MASE также устойчива к выбросам, сезонности и масштабу. Значение MASE меньше единицы указывает на то, что модель лучше наивного прогноза, в то время как значение больше единицы указывает на то, что модель хуже.

ГЛАВА 2. ПРИМЕНЕНИЕ НЕЙРОННЫХ СЕТЕЙ В ЗАДАЧАХ ПРОГНОЗИРОВАНИЯ

2.1 Архитектура нейронных сетей

Нейронные сети представляют собой структуры, являющиеся комбинациями дифференцируемых функций, составленные из блоков для нахождения приблизительных значений неизвестных функций с помощью имеющихся данных. Часто можно встретить представление нейронной сети в виде ориентированного ациклического графа. Между узлами данного графа расположены ребра, соединяющие данные узлы и содержащие веса, значения которых основываются на данных. Основой любой нейронной сети является нейрон. Каждый нейрон включает в себя входные параметры, линейную комбинацию с настраиваемыми коэффициентами и нелинейную функцию активации. В различных типах нейронных сетей эти компоненты расположены по-разному.[11]

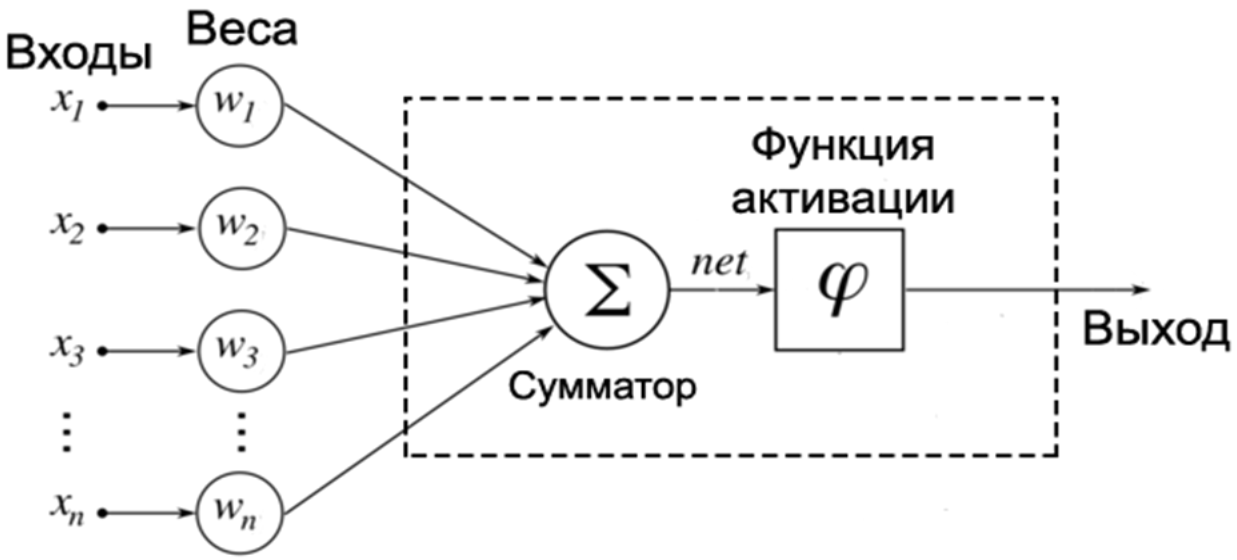

Рисунок 2.1 – Схема узла в нейронной сети

Многослойный персептрон. В многослойных персептронах, или, как их еще называют, нейронные сети с прямой связью, слои нейронов накладываются друг на друга с целью изучения более сложных представлений данных. Нейронные сети с прямой связью состоят из входных и выходных слоев, а также промежуточных или скрытых слоев. Учитывая тот факт, что на каждом уровне сети каждый узел связан со всеми узлами на предыдущем уровне, можно сказать, что каждый слой представляет собой полносвязный двудольный граф. Скрытые слои позволяют сети строить сложные

нелинейные представления исходных данных, которые затем анализируются выходным слоем для получения прогнозных значений. [6, с. 223]

Если говорить про недостатки многослойных персептронов, то из основных ограничений можно выделить то, что они не используют структуру, характерную для данных в областях обработки естественного языка, компьютерного зрения и прогнозирования. В добавок, количество входных и выходных данных является фиксированным, из-за этого они не могут применяться в задачах с разным количеством входных и выходных данных. Далее описываются более сложные архитектуры, способные преодолевать перечисленные выше ограничения, и для которых многослойный персептрон выступает в качестве основы.

Сверточные нейронные сети. Сверточные нейронные сети – это отдельный класс нейронных сетей, который испоользуется в случаях, когда известна порядковая структура входных данных. Чаще всего данные нейросетевые модели используются для анализа видео и изображений. Сверточным нейронным сетям присуще использование сверточных нейронных слоев, посредством применения функции свертки к меньшим областям входных данных. Этот метод позволяет выделять важные зависимости во входных данных.

Вместе со сверточными слоями также может использоваться объединяющий слой с целью уменьшения размерности пространства признаков и увеличения их устойчивости. Например, обычно используемый слой максимального объединения, который применяется к выходным данным сверточных слоев, извлекает максимальное значение объектов в заданной окрестности. Операция объединения выполняется схожим образом с операцией свертки: соответствующий фильтр перемещается в пределах мелких окрестностей входных данных. И все же свертка и объединяющий слой считаются одним слоем, так как последний не имеет инструментов для вычисления весов

При прогнозировании одними из важнейших элементов являются причинно-следственные связи, определяемые как

$$h_j = \sum_{d \in D} w_d \alpha_{j-d}, \qquad (2.1)$$

где h_j – выход скрытого узла;
α обозначает входные данные;
$D = \{1, 2, \ldots n\}$ для некоторого n.
Иначе говоря, причинно-следственные связи – это взвешенные скользящие средние, которые учитывают только те входные данные, которые находятся перед j.[7]

Рекуррентные нейронные сети. Рекуррентные нейронные сети или RNN – это вид нейронных сетей, специально разработанных для работы с последовательными данными. Они могут быть особенно полезными при работе с временными рядами, при обработке естественного языка и распознавании речи. Основная идея данной структуры состоит в том, чтобы с некоторым периодом времени скрытые блоки нейронной сети подключались обратно к самим себе с временной задержкой. Благодаря возможности скрытых модулей запоминать функциональные представления входных данных, передача их обратно самим себе представляется как сеть динамической памяти. Важной деталью при таком подходе является то, что для всех временных шагов используется одна и та же сеть, то есть веса сети распределяются по временным шагам. Эта идея распределения веса аналогична той, что используется в сверточных нейронных сетях, где используется один и тот же фильтр в разных частях входных данных. Такое устройство позволяет рекуррентной нейронной сети не только обрабатывать последовательности различной длины во время обучения, но и обобщать на последовательности, длина которых не видна во время обучения.

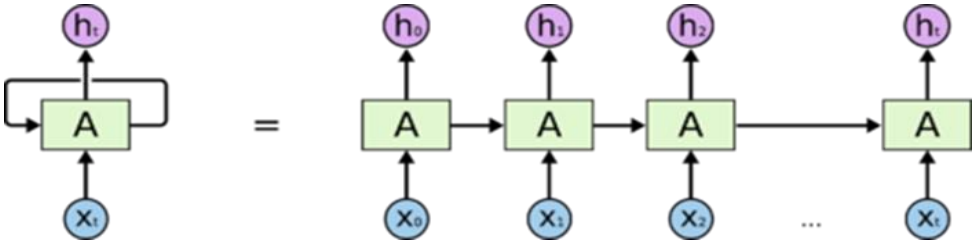

Рисунок 2.2 – Устройство рекуррентной нейронной сети

2.2 Модель LSTM

Традиционные рекуррентные нейронные сети сталкиваются с проблемой исчезающего градиента, которая затрудняет сети идентификацию долгосрочных зависимостей в последовательных данных. Однако LSTM (долгая краткосрочная память) элегантно решает эту проблему, поскольку включает специализированные ячейки памяти и механизмы стробирования, которые сохраняют и контролируют поток градиентов в расширенных последовательностях. Это позволяет сети более эффективно фиксировать долгосрочные зависимости и значительно повышает ее способность к обучению на основе последовательных данных. LSTM имеет три элемента управления (ввод, забвение и вывод) и отлично справляется с фиксацией долгосрочных зависимостей.

Алгоритм работы сверточных нейронных сетей следующий. Элементы ввода LSTM (2.2) и забывания (2.3) управляют состоянием ячейки (2.5), которая и является долговременной памятью. Элемент вывода (2.4), в свою очередь, формирует выходной вектор или скрытое состояние (2.6), которое является памятью, предназначенной для использования. Такая система памяти позволяет нейронной сети запоминать данные в течение длительного времени, чего очень не хватает обычным рекуррентным нейронным сетям.

$$i_t = sigmoid(W_i x_t + U_i h_{t-1} + b_i), \tag{2.2}$$

$$f_t = sigmoid\big(W_f x_t + U_f h_{t-1} + b_f\big), \tag{2.3}$$

$$o_t = sigmoid(W_o x_t + U_o h_{t-1} + b_o), \tag{2.4}$$

$$c_t = f_t \odot c_{t-1} + i_t \odot tanh(W_c x_t + U_c h_{t-1} + b_c), \tag{2.5}$$

$$h_t = o_t \odot \tanh(c_t), \tag{2.6}$$

где x_t – входной вектор;
h_t – выходной вектор;
c_t – вектор расстояния;
W, U – матрицы параметров;
f, i, o – векторы вентилей.[3]

Если сравнивать между собой обычные рекуррентные нейронные сети и LSTM, то основное различие между их архитектурами будет заключаться в том, что скрытое состояние LSTM формируется внутри специально организованной ячейки памяти. Эта ячейка состоит из четырех слоев, которые, в свою очередь, взаимодействуют таким образом, чтобы управлять потоком информации. Данные слои решают: какие данные сохранить, какие забыть и какие передать дальше. В результате формируются два выхода — скрытое состояние и состояние ячейки, которые передаются на следующий шаг.

В отличие от стандартной RNN, в которой используется только один слой с функцией активации, архитектура LSTM включает три логистических входа — входные, забывающие и выходные ворота, а также слой с функцией активации для обновления содержимого ячейки. Эти ворота управляют потоком информации внутри ячейки, определяя, какие данные оставить, какие забыть и какие передать дальше. Каждые из ворот выдают значения от 0 до 1, где 0 означает полное блокирование информации, а 1 — полную передачу.

Информация, которая больше не нужна в состоянии ячейки, удаляется с помощью забывающего элемента. На этом этапе на вход поступают данные

текущего шага и результат предыдущего шага. Они преобразуются с использованием весов и смещений, после чего проходят через сигмоидную функцию активации, которая выдает значения от 0 до 1. Значение, близкое к 0, означает, что соответствующая информация будет забыта, а значение, близкое к 1, — что информация будет сохранена.[3]

Добавление новой информации в ячейку происходит с помощью входного механизма. Сначала данные текущего и предыдущего шагов проходят через сигмоидную функцию, которая определяет, какие значения стоит запомнить. Затем те же входные данные обрабатываются функцией гиперболического тангенса, создающей набор новых кандидатных значений в диапазоне от -1 до +1. После этого эти два результата перемножаются и только важные элементы из кандидатных значений попадают в состояние ячейки.

Формирование выходных данных для LSTM осуществляется с помощью выходного элемента. То есть сначала на основе текущего состояния ячейки формируется вектор, проходящий через функцию активации — это и есть основа будущего выхода. Далее, как и в предыдущих случаях, данные текущего и предыдущего шагов проходят через сигмоидную функцию, которая решает, какие части информации нужно пропустить дальше. После прохождения через сигмоидную функцию выходной вектор фильтруется уже с учетом принятых решений и передается на следующий шаг.

С ростом популярности LSTM в ее традиционную архитектуру были внесены различные модификации, направленные на упрощение внутреннего устройства ячеек, повышение эффективности и снижение вычислительной нагрузки. Одно из усовершенствований — добавление так называемых «петлевых» соединений, которые позволяют управляющим элементам напрямую учитывать текущее состояние ячейки при принятии решений. В некоторых вариантах архитектуры также объединяются входной и забывающий элементы в один, что позволяет одновременно решать, какие данные сохранить, а какие — удалить, упрощая логику работы ячейки.

Альтернативным направлением развития рекуррентных нейронных сетей стало внедрение упрощенного рекуррентного блока (GRU), в котором снижена общая сложность модели за счет уменьшения числа компонентов. В GRU объединены скрытое состояние и состояние ячейки, а также отсутствует отдельный выходной элемент. Входной и забывающий элементы сливаются в единый обновляющий механизм, который регулирует как сохранение информации, так и ее обновление, что делает архитектуру более компактной и быстрее обучаемой при сохранении конкурентной точности.

2.3 Модель GRU

GRU – это, по сути, упрощенная версия LSTM. Он выполняет точно такую же роль в сети. Основное отличие заключается в количестве элементов и весовых коэффициентах — GRU несколько проще. В данной конфигурации рекуррентной нейронной сети всего 2 элемента. Поскольку у GRU отсутствует выходной элемент, нет никакого контроля над содержимым памяти. Модуль обновления из формулы (2.8) управляет потоком информации о предыдущей активации, а также добавлением новой информации (2.9). В то же время модуль сброса по формуле (2.7) вставляется в возможную активацию. В целом, архитектура GRU довольно похожа на LSTM. Поэтому, исходя из одних только различий, трудно сказать, какой из них является лучшим выбором.

Вообще говоря, считается, что основное преимущество GRU заключается в том, что она объединяет в себе преимущества LSTM, но при этом использует меньше параметров, так как не имеет отдельного состояния ячейки. Данная особенность делает ее подходящей для задач, где важны скорость и ресурсоэффективность.[3]

Обновление состояния GRU выполняется по следующим формулам:

$$r_t = \sigma(W_{xr}x_t + W_{hr}h_{t-1} + b_r), \tag{2.7}$$

$$z_t = \sigma(W_{xz}x_t + W_{hz}h_{t-1} + b_z), \tag{2.8}$$

$$h'_t = \tanh(W_{xh}x_t + W_{hh}(r_t \odot h_t) + b_h), \tag{2.9}$$

$$h_t = z_t \odot h_{t-1} + (1 - z_t) \odot h'_t, \tag{2.10}$$

где x_t – входной вектор на текущем шаге времени;
h_{t-1} – скрытое состояние с предыдущего шага;
r_t – вектор сброса;
z_t – вектор обновления;
h'_t – кандидаты нового состояния;
h_t – итоговое скрытое состояние;
σ – сигмоидальная функция активации;
W – обучаемые веса;
b – смещение.[3]

Исходя из представленных выше формул, нетрудно сделать вывод, что каждый из управляющих элементов выполняет важную роль в механизме работы GRU. Сбросный элемент определяет, насколько сильно скрытое состояние из предыдущего шага влияет на вычисления на текущем шаге. Если

вектор сброса близок к нулю, то модель «забывает» предыдущую информацию, что может быть полезно при обработке шумных или неинформативных данных. Обновляющий элемент, в свою очередь, определяет, следует ли сохранить старую информацию или заменить ее новым значением. Такое поведение позволяет данной нейронной сети адаптироваться к как краткосрочным, так и долгосрочным зависимостям в данных.[3]

GRU эффективно решает проблему исчезающего градиента, характерную для классических рекуррентных нейронных сетей. Так как GRU имеет меньшее число параметров по сравнению с LSTM, ее обучение и быстрее, и выгоднее в плане затраты ресурсов. Это особенно актуально в задачах, где доступна ограниченная вычислительная мощность или требуется обработка больших объемов данных в реальном времени. Например, в задачах потоковой обработки аудиосигналов или при развертывании моделей на мобильных устройствах.

В современных системах обработки речи GRU демонстрирует высокую эффективность, особенно в задачах идентификации говорящего. Речевая информация, представленная в виде спектрограммы, содержит как временные, так и частотные особенности, уникальные для каждого человека. Благодаря своей способности учитывать временные зависимости, GRU успешно извлекает голосовые паттерны из последовательностей признаков, таких как логарифмическая энергия мел-частотных фильтров.

Если говорить о GRU как о составной компоненте более сложной нейронной сети, то она хорошо работает в связке с другими архитектурами, такими как, например, те же самые сверточные нейронные сети, рассмотренные выше. Такая комбинация — CNN для пространственного анализа и GRU для временного анализа — широко применяется в современных моделях, в том числе в системах биометрической аутентификации и голосового управления.

2.4 Способы улучшения прогнозов нейронных сетей

Тщательная обработка входных данных является важным компонентом моделей глубокого обучения в целом и моделей глубокого прогнозирования в частности. Если говорить о моделях глубокого прогнозирования, то чаще всего они используются в качестве глобальных моделей, что означает, что веса нейронных сетей обучаются по всей панели временных рядов. Поэтому очень важно, чтобы масштаб исходных данных был сопоставимым. При прогнозировании применяются такие стандартные методы, как, например, вычисление средней дисперсии. На практике важно избегать утечки будущих

значений в схемах нормализации, чтобы среднее значение и дисперсия учитывались в прошлых окнах.

Для решение вышеописанной проблемы зачастую применяются различные преобразования входных данных. Как правило, в литературе чаще всего используется преобразование вида

$$h = \frac{z^{\lambda}-1}{\lambda}, \tag{2.11}$$

где где z – входные данные преобразования;
h – выходные данные;
λ – свободный параметр.

Преобразование Бокса-Кокса – это эвристика, позволяющая получить входные данные, близкие к гауссову распределению. Преобразование Бокса-Кокса может быть легко интегрировано в нейронные сети, при этом свободный параметр λ оптимизируется в рамках процесса обучения совместно с другими параметрами сети.

Еще одним стандартным методом преобразования входных данных является их дискретизация по категориальным значениям или ячейкам. Этот подход позволяет преобразовывать непрерывные признаки в дискретные интервалы, что упрощает дальнейшую обработку и анализ данных. Данные методы очень распространены при предварительной обработке данных для задач классификации. Особенно это актуально, когда необходимо снизить чувствительность модели к масштабированию признаков.

Стоит также отметить, что любое преобразование входных данных должно быть обращено вспять, чтобы получить значения в интересующей области. Обычно, выбор того, когда стоит применить преобразовани, ложится на плечи непосредственного разработчика нейросетевой модели. Как вариант, можно преобразовать входные и выходные данные полностью за пределами нейронной сети или использовать входные преобразования как часть нейронной сети и, следовательно, подвергать их обучению.

Часто в литературе в качестве функции преобразования можно встретить функцию гаусса. Она позволяет в том числе нормализовать входные данные, а также, в некоторых случаях, осуществить обратный переход к данным до преобразования. На рисунке 2.3 для гауссова распределения функция плотности f находится на левой панели, соответствующая функция кумулятивной плотности F (первичный интеграл от f) – на центральной панели, а квантильная функция F^{-1} – на правой панели.

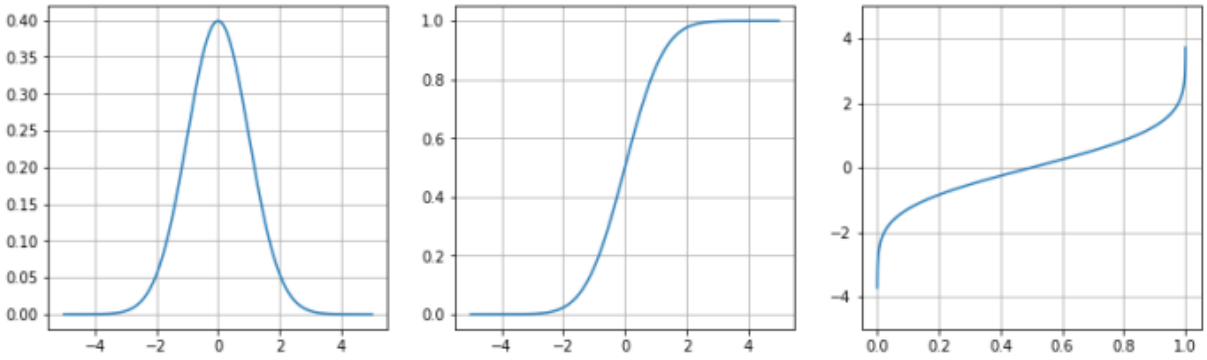

Рисунок 2.3 – Функции плотности для гауссова распределения

Если продолжать говорить о существующих способах улучшения прогноза нейронной сети, то нельзя не упомянуть изменение ее гиперпараметров. Настройка гиперпараметров нейросетевой модели представляет собой процесс выбора оптимальных значений для гиперпараметров модели машинного обучения. Гиперпараметры, в свою очередь, – это параметры, управляющие процессом обучения модели. К ним можно отнести скорость обучения, количество нейронов в нейронной сети или размер ядра в методе опорных векторов. Гиперпараметры настраивают с той целью, чтобы найти такие их значения, которые бы обеспечивали наилучшую производительность при решении задачи.

Если говорить про гиперпараметры в контексте нейронных сетей и машинного обучения, то под гиперпараметрами подразумеваются переменные конфигурации, которые задаются перед началом процесса обучения модели. Эти переменные управляют самим процессом обучения, а не извлекаются из данных. Гиперпараметры постоянно используются для настройки модели, и они оказывают решающее влияние на точность ее прогноза, а также производительность.

Далее выделены пять самых важных гиперпараметров, которые напрямую влияют на производительность и качество прогноза именно нейросетевых моделей:

- *Скорость обучения*. Скорость обучения определяет размер шага, который выполняет оптимизатор во время каждой итерации обучения. Слишком малая скорость обучения может привести к медленной конвергенции, в то время как слишком большая скорость обучения может привести к нестабильности и расхождениям.
- *Эпохи*. Данный гиперпараметр отражает то количество раз, когда весь обучающий набор данных проходит через модель во время обучения. Увеличение количества эпох может повысить производительность модели, но также может привести к переобучению, что является очень распространенной проблемой в области машинного обучения.

- *Количество слоев*. Количество слоев определяет глубину модели. Это оказывает существенное влияние как на ее сложность, так и на способность к обучению.
- *Количество нейронов на слое*. Это, пожалуй, самый важный и наиболее часто конфигурируемый гиперпараметр нейронной сети. Увеличение или уменьшение нейронов на скрытом слое напрямую влияет на точность и скорость получения прогноза.
- *Функция активации.* Данная функция определяет входной и выходной сигналы, а также привносит нелинейность в модель, что позволяет нейронным сетям обучаться на сложных наборах данных. Самыми распространенными функциями активации являются сигмовидная форма и гиперболический тангенс.

ГЛАВА 3. РЕЗУЛЬТАТЫ ЭКСПЕРИМЕНТАЛЬНОГО ИССЛЕДОВАНИЯ КАЧЕСТВА ПРОГНОЗИРОВАНИЯ

Для успешного прогнозирования временных рядов необходимо выбирать подходящую модель, которая будет учитывать компоненты временного ряда и давать наиболее точный прогноз. Если обучать и тестировать модель на довольно простых и «классических» временных рядах, например, с минимальным уровнем шума или без смены режимов, то при прогнозировании временных рядов в реальных задачах выбранная модель может составить прогноз, далекий от реальности. Это обусловлено тем, что, как правило, временные ряды из реальных задач обладают более сложной, часто непредсказуемой структурой.

Таким образом, в данной главе будет исследовано, как традиционные (линейная регрессия и авторегрессия) и нейросетевые (LSTM, GRU) модели прогнозирования справятся с прогнозированием временных рядов, обладающих нетривиальной структурой. Такой подход позволит оценить не только универсальность моделей, но и выяснить, где традиционные методы дают сбой, а где сложные нейросетевые архитектуры теряют преимущество.

3.1 Подбор гиперпараметров нейронной сети

В рамках данной работы в качестве нейронной сети для прогнозирования временных рядов используются долгая краткосрочная память (LSTM) – особая разновидность архитектуры рекуррентных нейронных сетей, способная к обучению долговременным зависимостям, а также GRU - похож на долговременную кратковременную память (LSTM) с механизмом управления для ввода или забывания определенных функций, но не имеет контекстного вектора или выходного шлюза, что приводит к меньшему количеству параметров, чем LSTM. Реализация данных нейронных сетей выбрана из библиотеки Keras для языка программирования Python. Данные нейронный сети обладают следующими гиперпараметрами:

- количество нейронов на входном слое;
- количество нейронов на скрытом слое;
- количество эпох обучения.

Далее осуществлялась подборка оптимальных гиперпараметров. Для каждого из наборов гиперпараметров осуществлялось обучение нейронных сетей и оценивалась норма вектора отклонений прогноза нейронной сети от фактических значений временного ряда с помощью MAE:

$$MAE = \frac{1}{n}\sum|y_{true} - y_{pred}|, \quad (3.1)$$

где n – число наблюдений;
y_{true} – фактическое значение;
y_{pred} – прогнозируемое значение.

В таблице 3.1 приведены результаты эксперимента по подбору оптимального набора гиперпараметров сетей для временного ряда с нелинейным трендом.

Таблица 3.1 – Среднее значение MAE в зависимости от набора гиперпараметров

Количество нейронов на скрытом слое	Количество эпох обучения	Средняя абсолютная ошибка (MAE)
50	15	6.08
50	20	4.78
50	25	5.23
100	15	4.10
100	20	4.28
100	25	3.82
150	15	4.16
150	20	3.76
150	25	3.78

Таким образом, при 50 нейронах на срытом слое, имела место тенденция на улучшение результата при увеличении количества эпох обучения до 20, при увеличении количества эпох до 25 точность прогноза снова упала. В дальнейшем, при увеличении количества эпох обучения, точность прогноза нейронной сети возрастала. При 150 нейронах на скрытом слое, разность в точности прогноза между нейронной сетью с 20 эпохами и нейронной сетью с 25 эпохами незначительна, поэтому было принято решение прекратить подбор гиперпараметров и выбрать оптимальный набор из уже имеющихся.

Исходя из результатов, отображенных в таблице 3.1, было принято решение в дальнейшем прогнозировать временные ряды с помощью нейронных сетей со 150 нейронами на скрытом слое.

3.2 Обзор прогнозируемых временных рядов

В данном разделе приведены формулы и графики временных рядов, сгенерированных при помощи инструментов языка программирования Python для дальнейшего прогнозирования с помощью моделей, рассматриваемых в рамках данной дипломной работы (Линейная регрессия, ARIMA, LSTM, GRU).

1. Временной ряд с модулированной сезонностью, шумом и трендом.

$$y_t = a * t + \left(\frac{1}{1+e^{-0.05\left(t-\frac{n}{2}\right)}}\right) * 3 * \sin\left(\frac{2\pi t}{s}\right) + \varepsilon_t\,, \qquad (3.2)$$

где $a * t$ – линейный тренд;
синусоида модулируется сигмоидой;
$\varepsilon_t \sim N(0,\ \sigma_t^2)$.

Для данного временного ряда характерно постепенное усиление сезонности, обусловленное ростом сигмоидной функции с течением времени. Это приводит к тому, что амплитуда колебаний в начале ряда близка к нулю и постепенно увеличивается, достигая насыщения ближе к концу ряда. Таким образом, модель имитирует ситуацию, в которой периодические колебания становятся все более выраженными на фоне общего линейного роста.

Такой ряд может использоваться для моделирования процессов с нарастающей цикличностью, например, спроса на продукт с сезонным характером, усиливающимся в результате маркетинговой активности, роста узнаваемости или внешнего тренда. Также он полезен для тестирования устойчивости моделей прогнозирования к динамически усиливающейся сезонности.

График рассматриваемого ряда представлен на рисунке 3.1.

Рисунок 3.1 – График временного ряда с модулированной сезонностью, трендом и шумом

2. Псевдослучайный временной ряд.

$$y_t = 2 * \sin(0.01 * t^2) + \varepsilon_t\,. \qquad (3.3)$$

В данном случае синусоида имеет нелинейно растущую частоту. Данный временной ряд может быть полезен для тестирования алгоритмов, способных выявлять неочевидные и нелинейные зависимости в данных,

особенно на ранних этапах анализа. Он представляет интерес в тех задачах, где структура ряда постепенно усложняется и требует от модели способности к адаптивному обучению. Примеры применения — распознавание динамически меняющихся сигналов, предсказание событий в нестационарных процессах, а также в задачах анализа частотно-модулированных колебаний, таких как радиосигналы или геофизические данные.

График временного ряда представлен на рисунке 3.2.

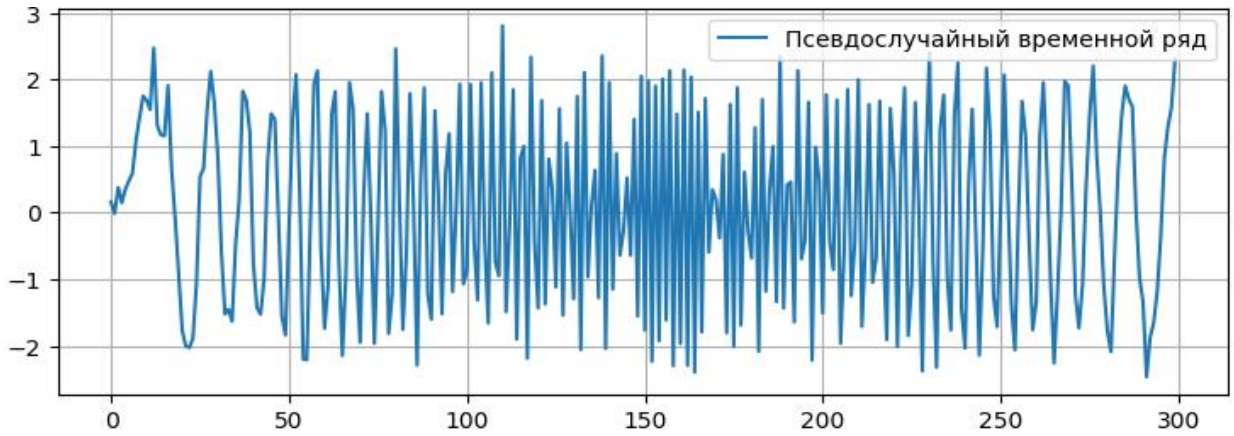

Рисунок 3.2– График псевдослучайного временного ряда

3. Гетероскедастичный временной ряд.

$$y_t = 2 * \sin\left(\frac{2\pi t}{25}\right) + \varepsilon_t\,, \tag{3.4}$$

где $\varepsilon_t \sim N(0,\ \sigma_t^2)$;
$\sigma_t = 0.2 + 0.02t$.

Для данного временного ряда характерно увеличение дисперсии со временем. С его помощью можно проверить модель на устойчивость к изменяющейся дисперсии. График данного временного ряда изображен на рисунке 3.3.

Рисунок 3.3 – График временного ряда с возрастающей дисперсией

4. Временной ряд с поэтапной сменой поведения.

$$y_t = \begin{cases} \sin\left(\frac{2\pi t}{20}\right) + \varepsilon_t, & t < \frac{n}{3} \\ 5 + \varepsilon_t, & \frac{n}{3} \le t \le \frac{2n}{3} \\ -2 + 0.1 * t + \varepsilon_t, & t > \frac{2n}{3} \end{cases}. \quad (3.5)$$

Данный временной ряд моделирует резкие изменения в поведении системы, посредством перехода от гармонического колебания к константе, а затем — к ряду с линейным трендом. Он представляет собой пример многорежимной структуры, где каждый сегмент отражает различный характер динамики: периодичность, стабильность и рост.

Представленная выше структура может использоваться для тестирования устойчивости моделей прогнозирования в условиях смены режимов работы системы, например, при переходе от нормального состояния к аномальному или при переключении между операционными фазами.

График данного временного ряда представлен на рисунке 3.4.

Рисунок 3.4 – График временного ряда с поэтапной сменой поведения

5. Хаотический временной ряд.

$$y_{t+1} = r * y_t * (1 - y_t), \quad (3.6)$$

где r = 3.8.

Данный временной ряд моделирует детерминированный хаос, то есть поведение непредсказуемо при высоких r. При значении параметра r = 3.8 система входит в область, где колебания становятся хаотичными: ряд остается полностью детерминированным (то есть заданным строго определенной формулой), однако его поведение становится крайне чувствительным к

начальному условию и практически непредсказуемым в долгосрочной перспективе. График представлен на рисунке 3.5.

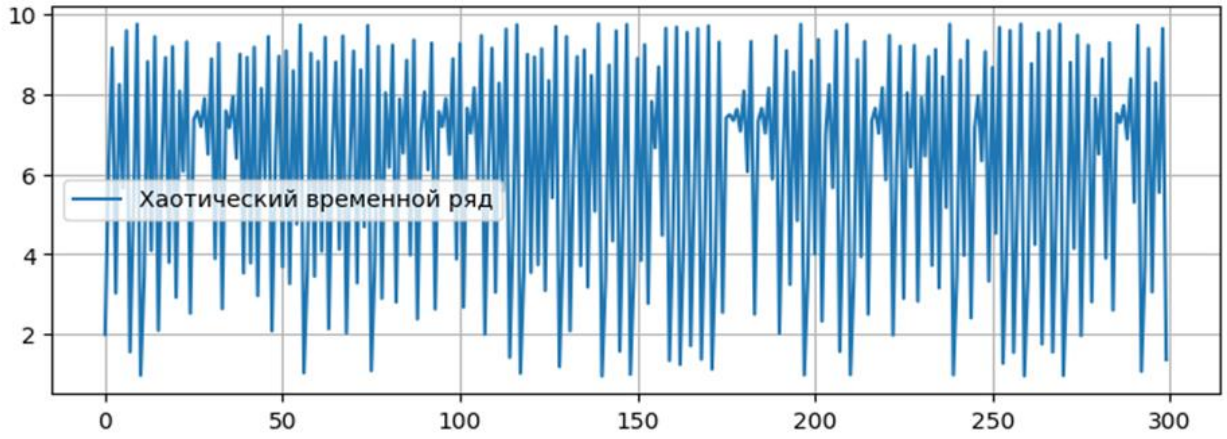

Рисунок 3.5 – график хаотического временного ряда

3.3 Сравнительный анализ классических и нейросетевых моделей прогнозирования временных рядов

В данном разделе приведены результаты сравнения точности прогнозов нейронных сетей и классических моделей прогнозирования временных рядов, рассмотренных в предыдущем пункте работы. В качестве показателей точности прогноза использовались значения MAE. В следующих таблицах в первых колонках цифры соответствуют следующим временным рядам: 1 – временной ряд с модулированной сезонностью, шумом и трендом, 2 – псевдослучайный временной ряд, 3 – временной ряд с поэтапной сменой поведения, 4 – гетероскедастичный временной ряд, 5 – хаотический временной ряд.

Перед тем как привести таблицы с результатами для остальных моделей, сначала будут представлены результаты прогнозирования с использованием линейной регрессии. Поскольку данная модель не требует подбора гиперпараметров для исследуемых временных рядов и является базовым методом, целесообразно выделить ее результаты в отдельную таблицу. Это позволит наглядно оценить качество прогноза классического подхода и использовать его в дальнейшем для сравнения с более сложными моделями машинного обучения. В таблице 3.2 приведены результаты прогнозов с помощью линейной регрессии по каждому из рассматриваемых временных рядов.

Таблица 3.2 – Результаты прогноза с помощью линейной регрессии

Временной ряд	MAE
1	1.0778
2	1.4749
3	0.5237
4	5.3871
5	2.1033

В таблице 3.3 приведены результаты прогнозов временных рядов с помощью модели ARIMA в зависимости от набора гиперпараметров.

Таблица 3.3 – Результаты прогноза с помощью ARIMA

Временной ряд	Гиперпараметры (p, d, q)					
	(0, 1, 0)	(2, 1, 2)	(3, 2, 3)	(4, 3, 4)	(5, 3, 5)	(4, 2, 3)
1	2.1758	3.7348	2.2382	2.1611	1.9948	2.4044
2	1.4908	1.3005	1.2449	1.2196	1.5445	1.3208
3	5.7769	5.3113	0.8500	12.5512	11.0109	0.7826
4	4.4903	4.4763	3.9642	5.4525	6.0558	4.1670
5	4.9663	2.6981	2.6982	2.7459	2.6930	2.6968

В таблице 3.4 приведены результаты прогнозов временных рядов с помощью нейросетевой модели LSTM в зависимости от гиперпараметров.

Таблица 3.4 – Результаты прогноза с помощью LSTM

Временной ряд	Нейроны на скрытом слое и количество скрытых слоев					
	(20, 1)	(30, 1)	(20, 2)	(30, 2)	(20, 3)	(30, 3)
1	1.5280	1.1689	1.9231	3.5822	4.2350	3.4815
2	1.1041	0.6846	0.9957	0.7542	1.2691	1.3138
3	1.7078	0.4523	0.4528	0.6462	3.9668	5.2318
4	4.7465	4.2121	4.2184	5.1613	5.5715	4.5772
5	2.2103	2.2746	2.7322	0.2786	2.7829	2.7429

В таблице 3.5 приведены результаты прогнозов временных рядов с помощью нейросетевой модели GRU в зависимости от гиперпараметров.

Таблица 3.5 – Результаты прогноза с помощью GRU

Временной ряд	Нейроны на скрытом слое и количество скрытых слоев					
	(20, 1)	(30, 1)	(20, 2)	(30, 2)	(20, 3)	(30, 3)
1	1.1688	1.2265	1.2414	1.2494	1.6635	1.4669
2	0.5784	0.5288	0.5302	0.4618	0.8052	0.7099
3	0.7321	0.4751	0.8251	1.5943	0.4670	0.6506
4	6.3255	5.9041	4.8260	4.8837	5.6458	5.5447
5	1.8906	0.1182	0.2012	0.2388	2.7295	2.9792

На рисунках 3.6 – 3.9 приведены графики прогнозов каждой из моделей для временного ряда с модулированной сезонностью, шумом и трендом.

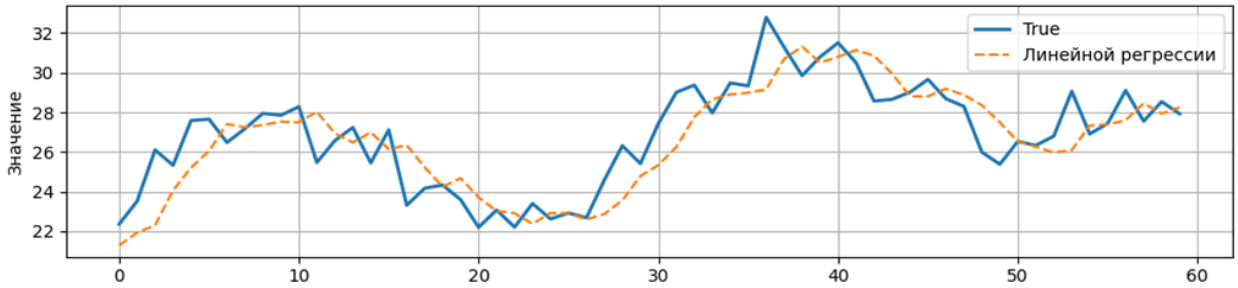

Рисунок 3.6 – График прогноза линейной регрессии для временного ряда с модулированной сезонностью, шумом и трендом

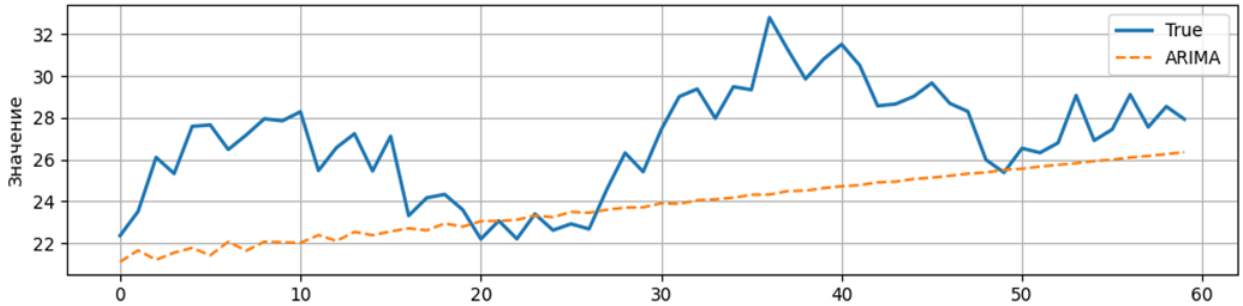

Рисунок 3.7 – График прогноза ARIMA для временного ряда с модулированной сезонностью, шумом и трендом

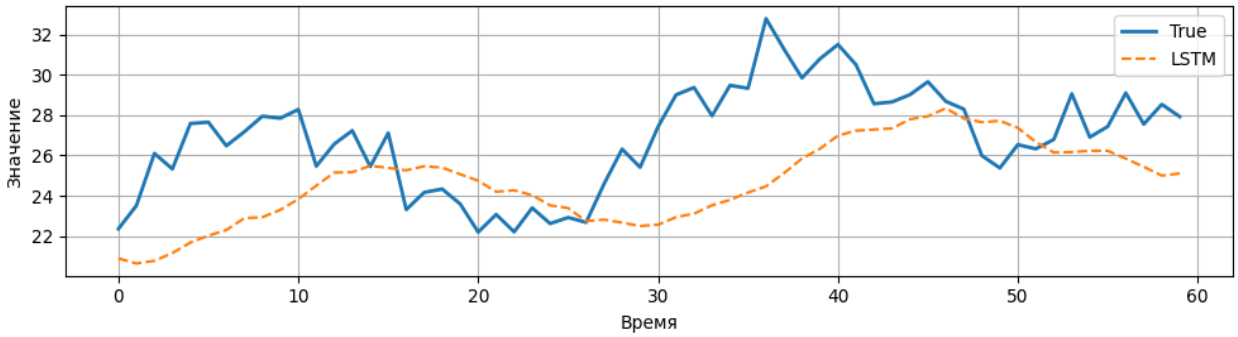

Рисунок 3.8 – График прогноза LSTM для временного ряда с модулированной сезонностью, шумом и трендом

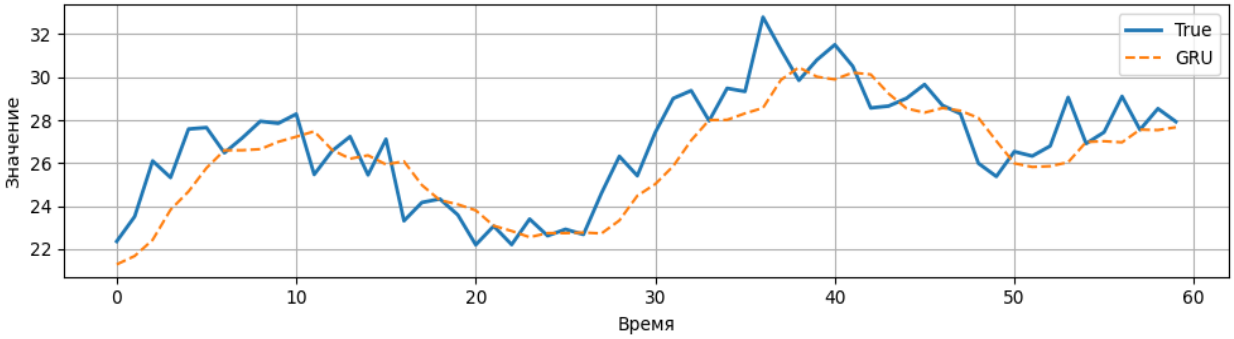

Рисунок 3.9 – График прогноза GRU для временного ряда с модулированной сезонностью, шумом и трендом

На рисунках 3.10 – 3.14 приведены графики прогнозов каждой из рассматриваемых моделей для псевдослучайного временного ряда.

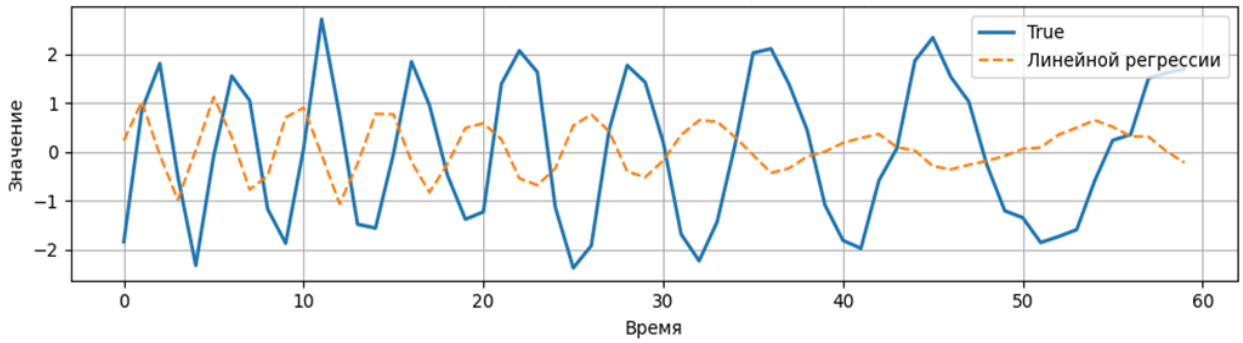

Рисунок 3.10 – График прогноза линейной регрессии для псевдослучайного временного ряда

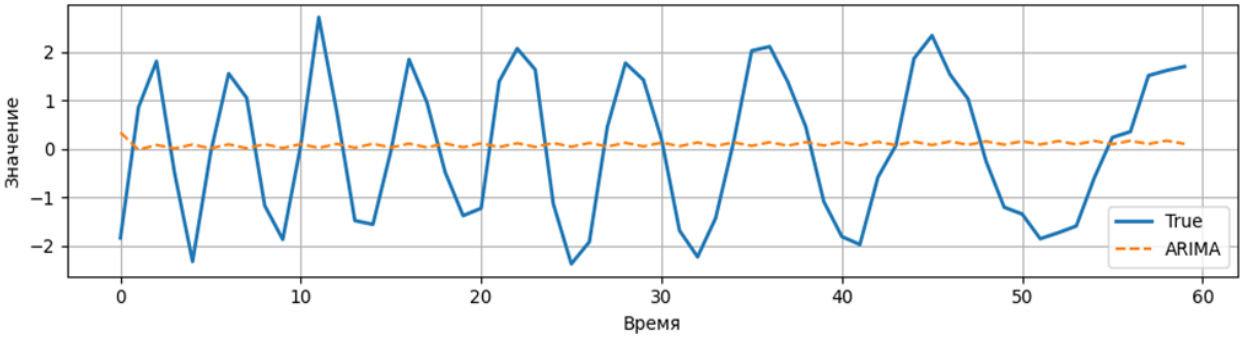

Рисунок 3.11 – График прогноза ARIMA для псевдослучайного временного ряда

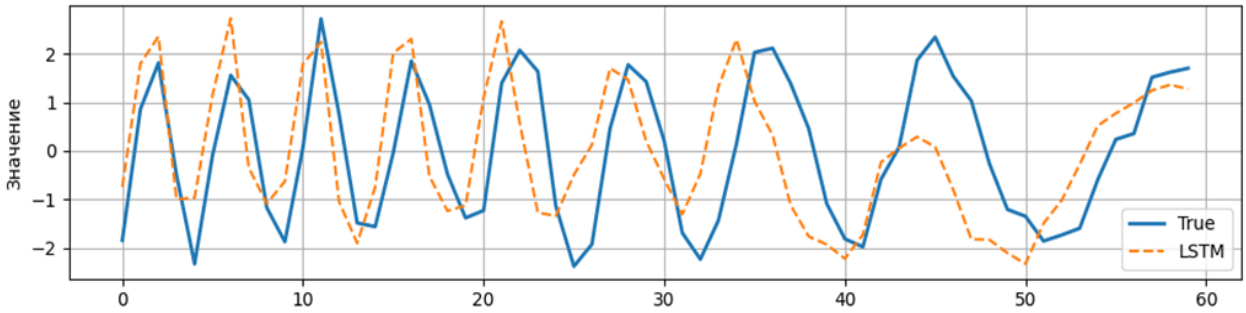

Рисунок 3.12 – График прогноза LSTM для псевдослучайного временного ряда

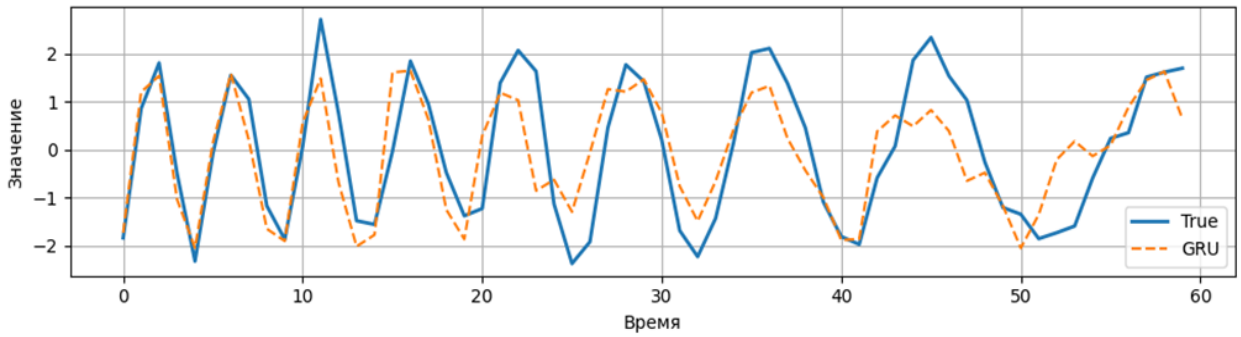

Рисунок 3.13 – График прогноза GRU для псевдослучайного временного ряда

На рисунках 3.14 – 3.17 приведены графики полученных прогнозов для моделей, обученных на данных исследуемого временного ряда с поэтапной сменой поведения.

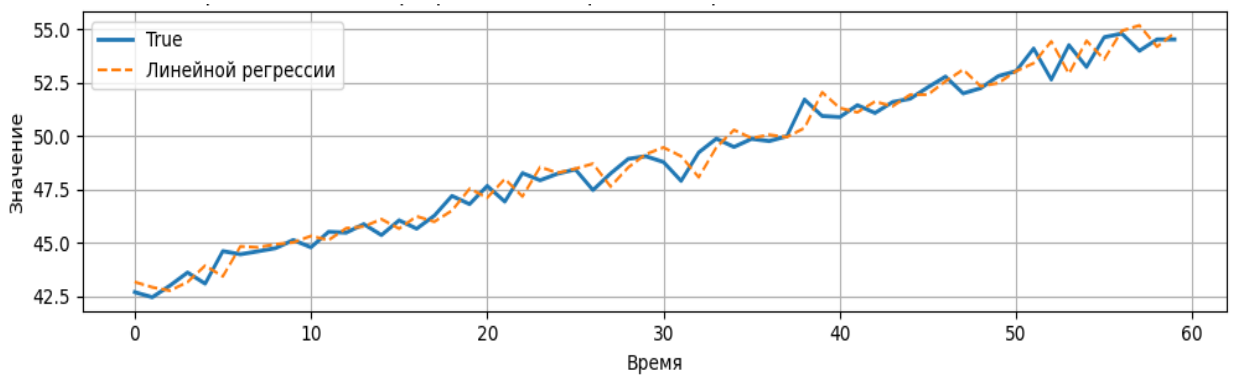

Рисунок 3.14 – График прогноза линейной регрессии для временного ряда с поэтапной сменой поведения

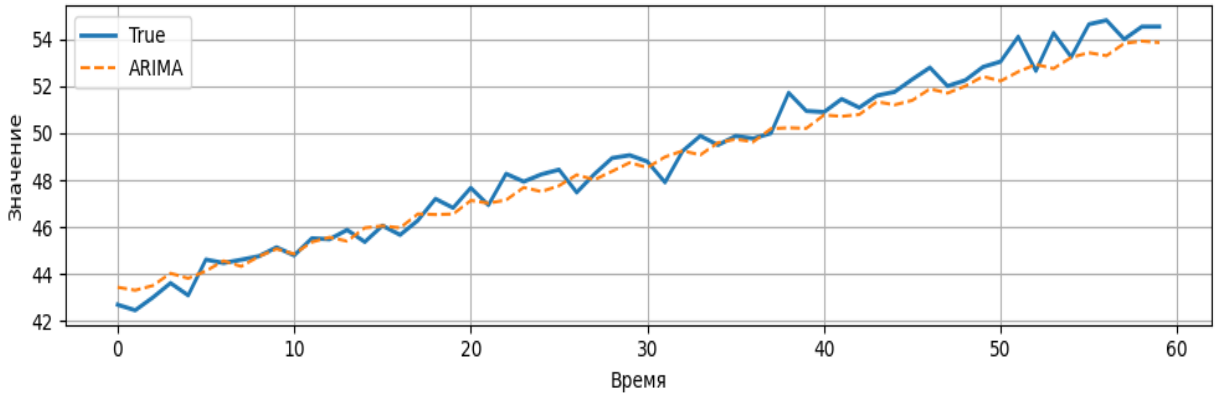

Рисунок 3.15 – График прогноза ARIMA для временного ряда с поэтапной сменой поведения

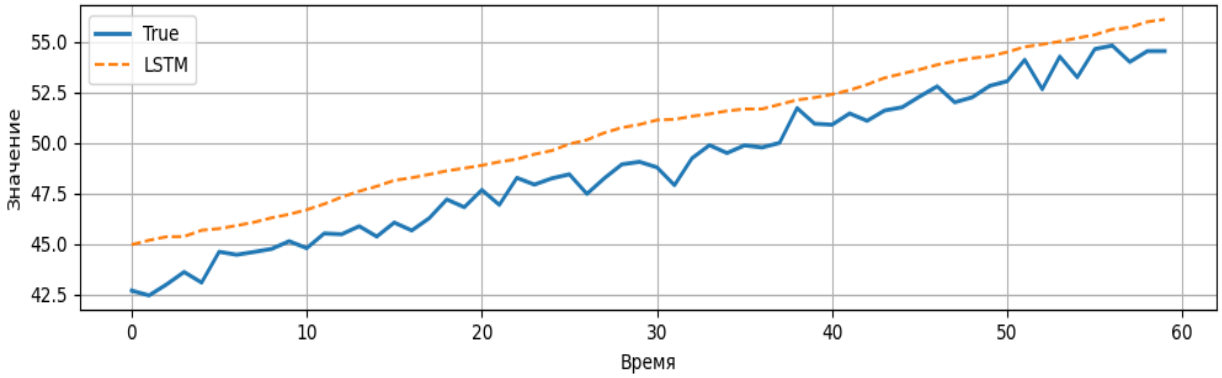

Рисунок 3.16 – График прогноза LSTM для временного ряда с поэтапной сменой поведения

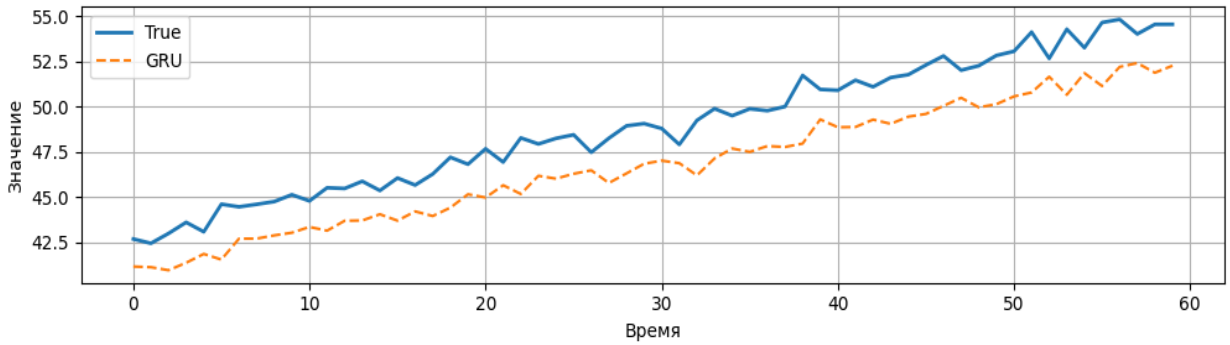

Рисунок 3.17 – График прогноза GRU для временного ряда с поэтапной сменой поведения

На рисунках 3.18 – 3.21 отображены прогнозные значения, полученные с помощью прогностических моделей (линейной регрессии, ARIMA, LSTM, GRU), для гетероскедастичного временного ряда.

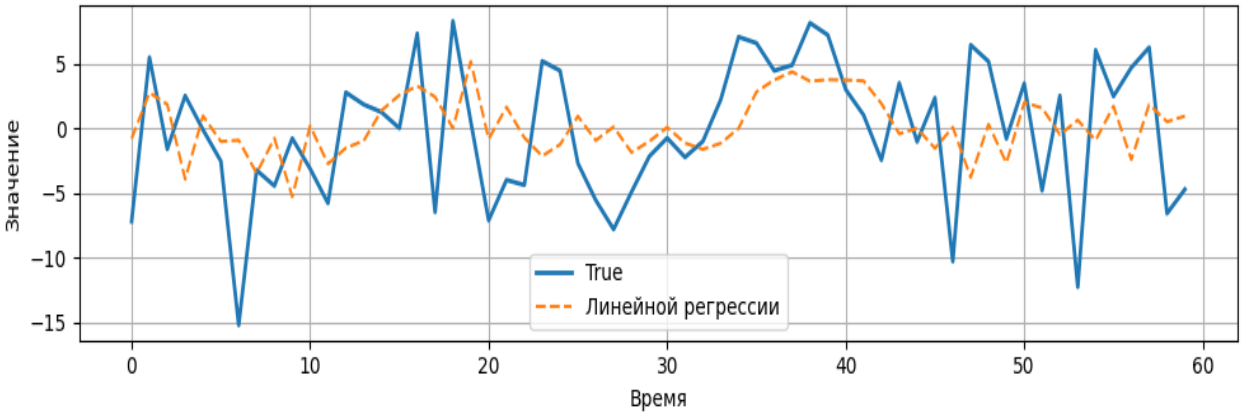

Рисунок 3.18 – График прогноза линейной регрессии для гетероскедастичного временного ряда

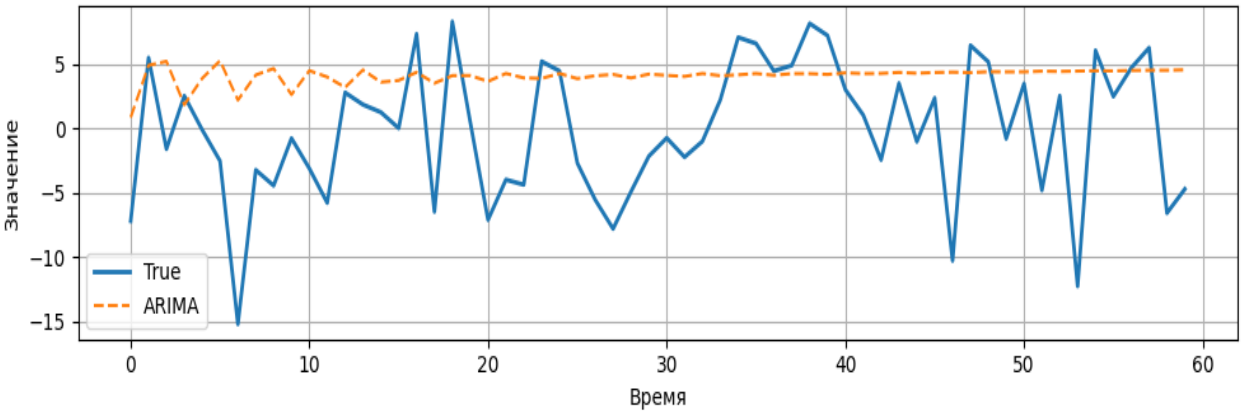

Рисунок 3.19 – График прогноза ARIMA для гетероскедастичного временного ряда

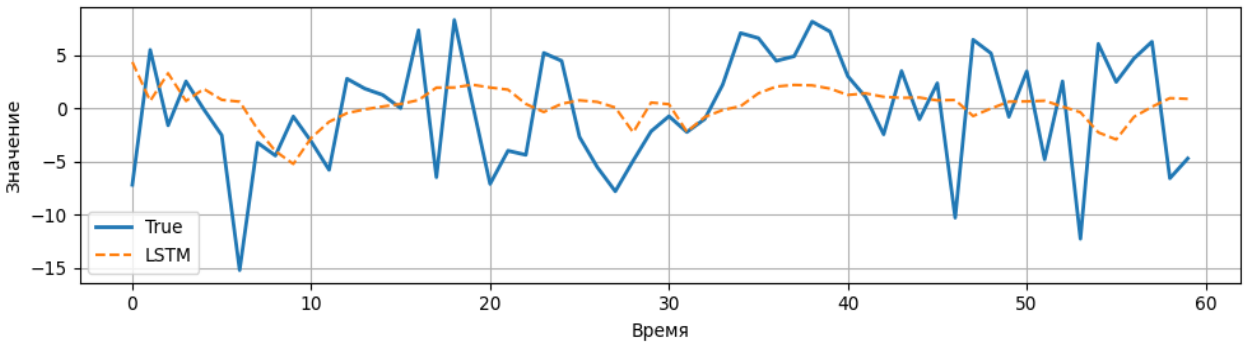

Рисунок 3.20 – График прогноза LSTM для гетероскедастичного временного ряда

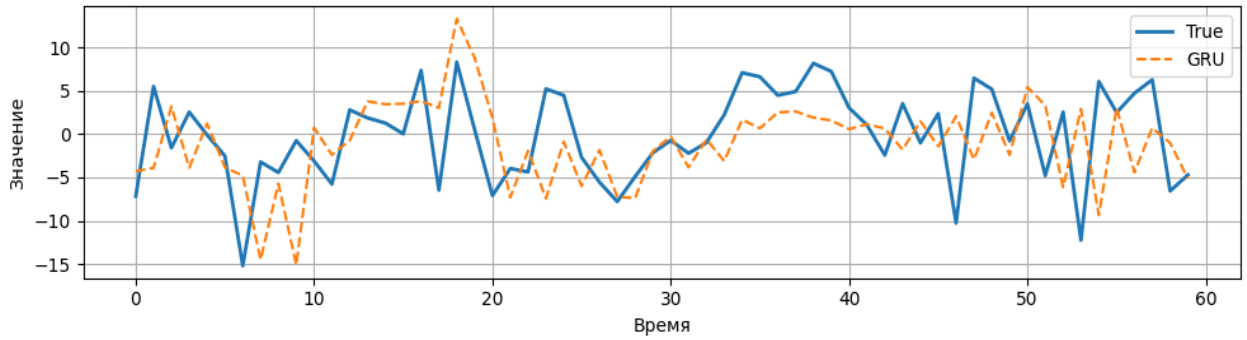

Рисунок 3.21 – График прогноза GRU для гетероскедастичного временного ряда

Далее приведены графики прогнозов исследуемых прогностических моделей для хаотического временного ряда (рис. 3.22 – рис. 3.25).

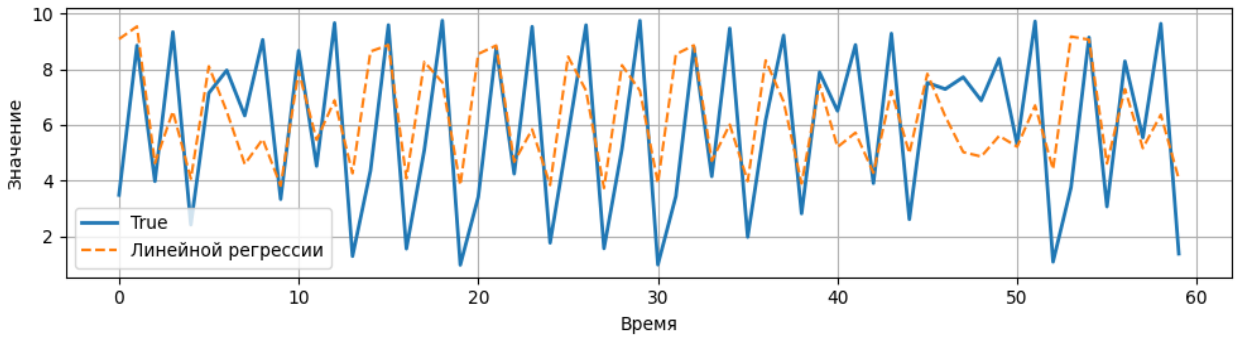

Рисунок 3.22 – График прогноза линейной регрессии для хаотического временного ряда

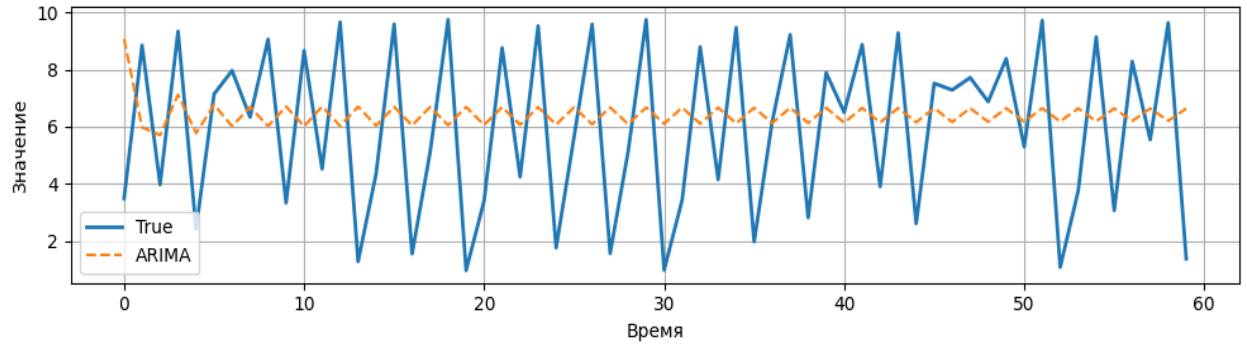

Рисунок 3.23 – График прогноза ARIMA для хаотического временного ряда

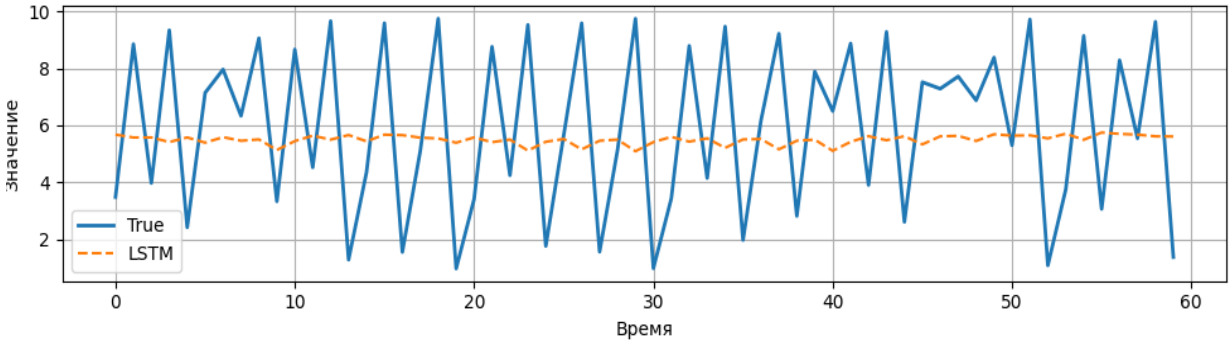

Рисунок 3.24 – График прогноза LSTM для хаотического временного ряда

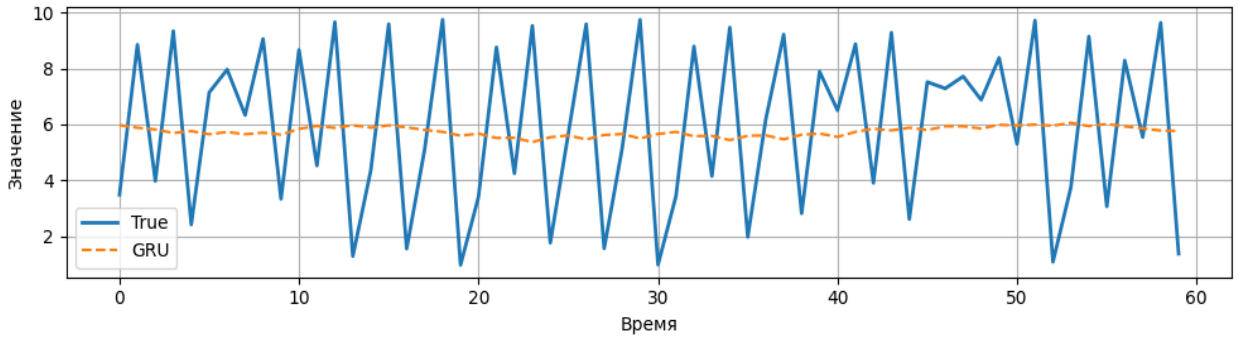

Рисунок 3.25 – График прогноза GRU для хаотического временного ряда

В таблице 3.6 приведены лучшие результаты MAE для каждой из моделей при прогнозировании исследуемых временных рядов.

Таблица 3.6 – Лучшие результаты прогностических моделей

Временной ряд	Прогностическая модель			
	Линейная регрессия	ARIMA	LSTM	GRU
1	1.0778	1.9948	1.1689	1.1688
2	1.4749	1.2196	0.6846	0.4618
3	0.5237	0.7826	0.4523	0.4670
4	5.3871	3.9642	4.2121	4.8260
5	2.1033	2.6930	0.2786	0.1182

Таким образом, исходя из полученных результатов, приведенных в таблице 3.6, можно сделать вывод, что лучше всего себя показали нейросетевые модели. В частности GRU была немного лучше LSTM в прогнозировании псевдослучайного и хаотического временных рядов(0.4618 против 0.6846 и 0.1182 против 0.2786), а LSTM оказалась точнее при прогнозировании гетероскедастичного временного ряда и временного ряда с поэтапной сменой поведения (0.4523 против 0.4670 и 4.2121 против 4.8260). Тем не менее, в случае с временным рядом, у которого дисперсия увеличивалась со временем, ARIMA показала лучший результат (3.9642). А в

случае с временным рядом с модулированной сезонностью, шумом и трендом лучше всего себя удалось показать линейной регрессии (1.0778). Необходимо также заметить, что лучшие результаты при прогнозировании различных по структуре временных рядов, модели показали с различными наборами параметров. Так, например, у GRU лучший результат при прогнозировании хаотического временного ряда удалось достичь при 30 эпохах обучения и 1 скрытом слое, а лучший результат при прогнозировании псевдослучайного временного ряда был достигнут при 30 эпохах и 2-х скрытых слоях.

ЗАКЛЮЧЕНИЕ

В ходе работы были выполнены как обзорные, так и экспериментальные действия. Их цель — понять, как параметры временных рядов влияют на точность прогнозов с помощью разных моделей: как классических, так и основанных на нейросетях. Основной задачей было сравнение работы исследуемых моделей при прогнозировании временных рядов с нетипичной структурой.

Для построения и анализа моделей использовались инструменты на языке Python, а именно библиотеки statsmodels, scikit-learn, keras и tensorflow. С их помощью были реализованы все этапы экспериментального исследования:

- создание моделей;
- настройка моделей;
- обучение моделей на тестовых выборках;
- оценка качества прогнозирования.

Для оценки точности использовались два показателя: MAE и RMSE.

В рамках практической части дипломной работы были сгенерированы следующие временные ряды с различной структурой: с увеличивающейся дисперсией, с меняющейся сезонностью, шумом, общим направлением и с хаотичными элементами. По результатам проведенного эксперимента был сделан вывод, что, к сожалению, не существует ультимативной модели прогнозирования, которую можно было бы одинаково эффективно использовать при прогнозирования различных по структуре временных рядов, и что в зависимости от временного ряда предпочтительными могут быть совершенно разные прогностические модели.

В большинстве случаев нейросетевые модели GRU и LSTM показали лучшие результаты. Модель ARIMA оказалась предпочтительнее при работе с рядами, где дисперсия со временем увеличивалась, но в остальном структура ряда оставалась такой же. Линейная регрессия хорошо показала себя в случае с временным рядом с выраженными сезонностью, трендом и шумом.

Также немаловажен тот факт, что результат прогнозирования напрямую зависел и от того, каковы были настройки гиперпараметров у модели. Например, при прогнозировании хаотичных рядов лучше показали себя сети с более сложные архитектуры, в то время как при работе с сезонными или псевдослучайными рядами хорошие результаты давали модели с меньшим количеством скрытых слоев.

Результаты данной работы могут быть применимы при прогнозировании данных в таких сферах, как экономика, транспорт, медицина, энергетика и IT.

Описанные методы и выводы могут служить основой для дальнейших исследований и практического применения при анализе временных данных.

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1. Зосимов, С. А. Анализ временных рядов на примере ведущего экономического индекса : дипломная работа / С. А. Зосимов ; БГУ. – 2017. – 49 с.
2. Лукашин, Ю. П. Адаптивные методы краткосрочного прогнозирования временных рядов / Ю. П. Лукашин. – М. : Финансы и статистика, 2003. – 416 с.
3. Нейросети для работы с последовательностями [Электронный ресурс] // Яндекс Практикум: справочник по машинному обучению. – Режим доступа: https://education.yandex.ru/handbook/ml/article/nejroseti-dlya-raboty-s-posledovatelnostyami. – Дата доступа: 22.03.2025.
4. Нильсен, Э. Практический анализ временных рядов: прогнозирование со статистикой и машинным обучением / Э. Нильсен ; пер. с англ. – СПб. : Диалектика, 2021. – 544 с.
5. Созыкин, А. Программирование нейронных сетей на Python [Электронный ресурс] / А. Созыкин. – Режим доступа: https://www.asozykin.ru/courses/nnpython/. – Дата доступа: 26.11.2024.
6. Хайкин, С. Нейронные сети: полный курс / С. Хайкин. – 2-е изд. – М. : Вильямс, 2014. – 1104 с.
7. Шолтанюк, С. В. Устойчивость моделей прогнозирования: алгоритмы и примеры. Классификация, анализ и верификация прогностических моделей / С. В. Шолтанюк. – LAP LAMBERT Academic Publishing, 2019. – 60 с.
8. Barker, D. The weather research and forecasting model's community variational / ensemble data assimilation system: WRFDA / D. Barker [et al.] // Bulletin of the American Meteorological Society. – 2012. – Vol. 93, № 6. – С. 831–843.
9. Box, G. E. P. Time series analysis forecasting and control / G. E. P. Box, G. M. Jenkins. – Rev. ed. – Oakland, California : Holden-Day, 1976. – С. 238–242.
10. Brownlee, J. Introduction to Time Series Forecasting with Python / J. Brownlee. – Jason Brownlee, 2020. – 352 с.
11. Hyndman, R. J. Forecasting: Principles and Practice / R. J. Hyndman, G. Athanasopoulos. – OTexts.com, 2014. – 291 с.
12. Khandelwal, I. Time series forecasting using hybrid ARIMA and ANN models based on DWT decomposition / I. Khandelwal, R. Adhikari, G. Verma // Procedia Computer Science. – 2015. – Vol. 48. – С. 173–179.