САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Перла Лев Аркадьевич

Выпускная квалификационная работа

Применение моделей рекуррентных нейронных сетей для прогнозирования финансовых временных рядов

Уровень образования: Магистратура
Направление 38.04.05 «Бизнес-информатика»
Основная образовательная программа ВМ.5604.2020
«Информационная бизнес-аналитика»

Научный руководитель:
доцент кафедры информационных
систем в экономике,
кандидат экономических наук
Забоев М.В.

Рецензент:
ведущий экономист,
ООО «ПроИнСофт»
Петров Н. А.

Санкт-Петербург
2022

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

В рамках любого бизнеса, особенно крупного, важную роль играет процесс планирования. От того, насколько точно будут оценены доходы и издержки будущих периодов напрямую зависит эффективность деятельности компании. Одной из наиболее частых задач, возникающих в процессе планирования, является прогнозирование временных рядов.

В этой области использование нейронных сетей, как и других моделей машинного обучения, многие специалисты еще недавно считали неконкурентоспособным методом по сравнению с более простыми по структуре статистическими моделями, такими как экспоненциальное сглаживание и ARIMA, которые широко используются в экономических исследованиях.

Это было связано с тем, что модели нейронных сетей имеют достаточно сложную структуру и большое количество гиперпараметров, от подбора комбинации которых напрямую зависит вероятность переобучения модели, а следственно и точность итогового прогноза.

Также нейронные сети критиковали за то, что они имеют природу «черного ящика», в котором могут быть миллионы параметров, значения которых проходят через нелинейные преобразования в рамках архитектуры сети. Из-за этого становится крайне сложно интерпретировать влияние входных данных на прогноз. Поэтому исследователи выбирали более простые статистические модели, зарекомендовавшие себя в качестве простых, надежных и легко интерпретируемых.

Помимо этого, традиционные модели для прогнозирования одномерных временных рядов показывали лучшее качество прогноза по сравнению с другими вычислительными методами на многих соревнованиях по прогнозированию, включая NN3, NN5 и M3.

Однако статистические методы ограничены малым количеством обучаемых параметров, отчего возникает риск недообучения модели в случае, когда количество наблюдений в обучающей выборке кратно возрастет. Также, если горизонт прогнозирования становится достаточно большим, качество традиционных методов начинает падать. Из-за этого исследователей все больше стали интересовать модели рекуррентных нейронных сетей (RNN). Их преимуществом является то, что они позволяют гибко настраивать свою структуру, а это представляет большой потенциал для увеличения качества прогноза.

С другой стороны, вариативность моделей RNN влечет за собой проблемы настройки гиперпараметров, выбора архитектуры, методов и стратегий для прогнозирования.

В литературе нет конечного понимания, какие комбинации методов предобработки временного ряда лучше всего влияют на качество прогноза. Особенно остро это проявляется

в вопросе могут ли модели рекуррентных нейронных сетей моделировать сезонную составляющую. Также в публикациях нет оценки влияния стратегии прогнозирования конкретно для моделей рекуррентных нейронных сетей. Поиск ответов на данные вопросы отражает актуальность нашего исследования.

В настоящее время данных, собираемых из различных бизнес-процессов деятельности компании, становится все больше. Поэтому зачастую становится невозможным в ручном режиме прогнозировать все необходимые показатели, и аналитики прибегают к упрощению задачи путем уменьшения детализации временных рядов, теряя при этом необходимую точность. Это привело к тому, что в области прогнозирования временных рядов появляется стремление к разработке автоматических прогнозных систем. Одним из решений подобной задачи стала библиотека forecast для языка программирования R[1], выпущенная Робом Хиндаманом в 2008 году, в которой были представлены автоматизированные статистические методы, такие как auto.arima и auto.ets.

Основные фреймворки для работы с нейронными сетями на настоящий момент написаны для языка Python. Наиболее популярными из них являются Tensorflow от компании Google и PyTorch, который разрабатывается преимущественно группой искусственного интеллекта Facebook. Однако эти фреймворки являются достаточно сложными для использования, так как для написания на их основе прогнозных систем необходимо обладать глубокими знаниями в предметной области и потратить достаточно большое количество времени на разработку. Поэтому чаще всего они становятся вычислительным ядром для верхнеуровневых библиотек, которые хотя и жертвуют вариативностью методов, но позволяют аналитикам выполнять многие действия в автоматическом формате.

Применительно к задаче прогнозирования временных рядов мы можем отметить два открытых фреймворка, которые упрощают процедуру получения прогнозов с помощью методов глубокого обучения:

GluonTS[2] – открытая библиотека для языка Python, которая предоставляет возможность гибко моделировать прогнозные модели нейронных сетей. В ней содержатся инструменты для первичной обработки временных рядов, создания и тренировки моделей глубокого обучения. Помимо этого, есть возможность приведения прогнозов в вероятностную форму.

---

[1] Hyndman, R., Khandakar, Y., 2008. Automatic time series forecasting: The forecast package for R. Journal of Statistical Software, Articles 27 (3), 1–22.

[2] Документация фреймворка GluonTS [Электронный ресурс] - Режим доступа: www.github.com/awslabs/gluon-ts/, (дата обращения: 11.04.2022)

Darts[3] — это библиотека Python для упрощенного прогнозирования временных рядов. Она содержит множество моделей от классических до глубоких нейронных сетей. Все модели имеют единую структуру, которая воспроизводит методологию scikit-learn, в которой все процессы агрегируются в функциях fit() и predict(). Библиотека также позволяет легко проводить кросс-валидационное тестирование моделей, комбинировать прогнозы нескольких моделей и учитывать внешние данные в качестве прогнозных факторов.

Однако в представленных библиотеках содержится ряд недостатков:

1. Нет возможности выбирать стратегию прогнозирования.
2. Нет автоматической возможности использовать глобальные модели.
3. Нет встроенного оптимизатора гиперпараметров моделей.

Это потребовало от нас предварительно разработать собственный фрейворк для того, чтобы реализовать цель нашей работы, и заложить в нем все перечисленные выше возможности, что дополнительно подкрепляет актуальность нашего исследования.

Цель работы - разработать автоматическую прогнозную систему, основанную на моделях рекуррентных нейронных сетей, на языке Python для того, чтобы увеличить точность прогнозирования финансовых временных рядов в рамках процесса бизнес-планирования.

При создании прогнозной системы мы опирались на результаты, полученные в рамках проведенных нами экспериментов в области финансовых временных рядов.

Объектом исследования являются финансовые временные ряды.

Предметом исследования являются прогнозные модели рекуррентных нейронных сетей, применяемые для финансовых временных рядов.

Осуществление поставленной цели демонстрируется на конкретном примере – кейсе применения разработанной прогнозной системы на проекте компании ООО «Газпромнефть-Региональные продажи».

Для реализации поставленной цели нам необходимо выполнить следующие задачи:

1. В теоретическом блоке:
    a. Проанализировать область исследования, выделяя основные методы предобработки временных рядов, стратегий к построению прогноза и подходов, сочетающих информацию из разных временных рядов;
    b. Определить какие существуют особенности в области прогнозирования финансовых временных рядов.
2. В экспериментальном блоке:

[3] Документация фреймворка Darts [Электронный ресурс] - Режим доступа: www.github.com/unit8co/darts, (дата обращения: 11.04.2022)

   a. Провести эксперимент с оценкой влияния комбинаций методов предобработки временного ряда на качество прогноза RNN;
   b. Провести эксперимент с оценкой влияния стратегии прогнозирования на качество прогноза RNN;
   c. Провести эксперимент с оценкой использования моделей RNN, сочетающих информацию из нескольких временных рядов.

3. В практическом блоке:
   a. На основе созданного нами фреймворка разработать прогнозную систему и применить ее в условиях проекта компании Газпром нефть.

В основе всей практической части работы лежит созданный нами фреймворк, но сам по себе процесс написания фреймворка здесь не приводится, так это выходит за рамки наших задач. Программный код фреймфорка является открытым, с ним можно ознакомиться в источнике[4].

В первой главе нашей работы мы решали задачи из теоретического блока: представили количественный и качественный анализ области прогнозирования временных рядов с помощью RNN моделей, а также описали методы, которые были использованы в рамках работы.

Вторая глава посвящена решению задач экспериментального блока: описанию методологии проводимых нами экспериментов, а также результатов, которые мы получили в результате их реализации.

В третьей главе, касающийся практического блока задач, описывается реальный проект компании «Газпром нефть», в рамках которого мы применили разработанный фреймворк по прогнозированию с помощью RNN. Приводится описание поставленной задачи, а также результаты от внедрения моделей.

[4] Программный код библиотеки TS_RNN [Электронный ресурс] - Режим доступа: www.github.com/LevPerla/Time_Series_Prediction_RNN, (дата обращения: 11.04.2022)

ГЛАВА 1. ТЕОРЕТИЧЕСКАЯ ЧАСТЬ

1.1. Литературный обзор

Область нашего исследования в настоящее время активно развивается. Мы рассмотрели актуальные публикации в отраслевой научной и профессиональной литературе России и других стран и составили на их основе обзор современного состояния области исследования, а также основных этапов ее развития.

Целью данного раздела является поиск ответов на следующие вопросы:

1. Какие методы чаще всего используют исследователи, решая задачу прогнозирования временных рядов? Какое развитие получили эти методы?
2. На какой стадии развития находится область прогнозирования временных рядов с помощью моделей рекуррентных нейронных сетей?
3. Какие особенности существуют при прогнозировании финансовых временных рядов?

В первом подразделе мы представляем количественный анализ области исследования, в котором отражается распределение публикаций по годам, наиболее популярные области знания, в которых отражен предмет исследования, а также анализ ключевых слов в текстах аннотаций отобранных статей.

Второй подраздел отражает качественный анализ источников. В нем мы расширяем область поиска и разделяем статьи по группам, основываясь на их содержании.

В третьем подразделе мы рассматриваем эволюцию развития рекуррентных нейронных сетей применительно к задаче прогнозирования временных рядов.

В четвертом подразделе отражается обзор использования моделей RNN применительно к финансовым временным рядам.

1.1.1. Анализ области исследования

Количественный анализ состояния области исследования осуществлен на основе базы Web of Science[5], из которой были взяты статьи с 1990 по 2022 год, содержащие основные ключевые слова. Данная площадка имеет удобный интерфейс для выгрузки выборки статей, включая их метаданные по сформированному запросу. Полученная статистическая информация о характеристиках статей послужила нам основой для анализа представленности в литературе области нашего исследования.

Полный текст сформированного нами запроса представлен ниже:

((AK=(time series)) OR (TI=(time series)))

[5] Международная научная база данных Web of Science [Электронный ресурс] Режим доступа: https://www.webofscience.com, (дата обращения: 11.04.2022)

AND (AK=(recurrent neural network*)

OR AK=(RNN*)

OR AK=(LSTM*)

OR AK=(GRU*))

AND (AK=(forecasting*) OR AK=(prediction*))

AND PY=(1990-2022)

В результате его выполнения мы получили 664 публикации, первая из которых относится к 1996 году.

На рисунке 1.1 представлен график распределения количества статей по году их выпуска. Область нашего исследования находится на пике своей популярности: наибольшее количество публикаций по теме было сделано за последние три года, что подтверждается в статье[6], в которой исследователи приводят аналогичный рисунок, обозревая применение моделей нейронных сетей для прогнозирования финансовых временных рядов.

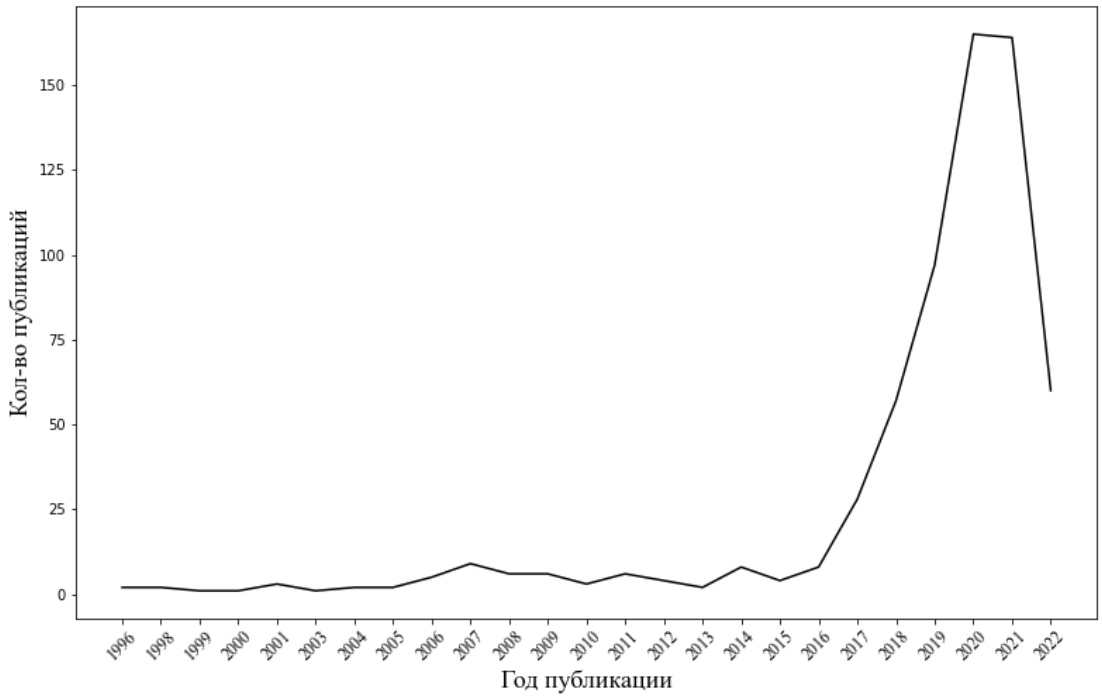


Рисунок 1.1 – **Распределение публикаций по годам**

Необходимо отметить, что наблюдаемый спад в 2022 году связан с тем, что наше исследование проводится в начале этого года, поэтому не все статьи еще были опубликованы и могли попасть в текущую выборку.

Можно предположить, что популярность прогнозирования с помощью рекуррентных нейронных сетей связана с такими тенденциями, как увеличение производительности вычислительных устройств и цифровизация многих отраслей общественной жизни.

[6] Omer Berat Sezer, Mehmet Ugur Gudelek, Ahmet Murat Ozbayoglu, Financial time series forecasting with deep learning: A systematic literature review: 2005–2019, Applied Soft Computing, Volume 90, 2020, 106181

Также существенный вклад в развитие области прогнозирования временных рядов внесло соревнование M4, которое проходило в 2018 году и предоставило широкое поле возможностей для исследователей. В большом количестве статей, выпущенных в 2019–2021 годах, авторы опирались либо на данные этого соревнования, либо на дальнейший анализ методов, которые показали наиболее точные результаты в рамках соревнования.

На рисунке Рисунок 1.2 представлены основные 5 областей знаний, представленных категориями Web of Science, к которым относятся отобранные статьи:

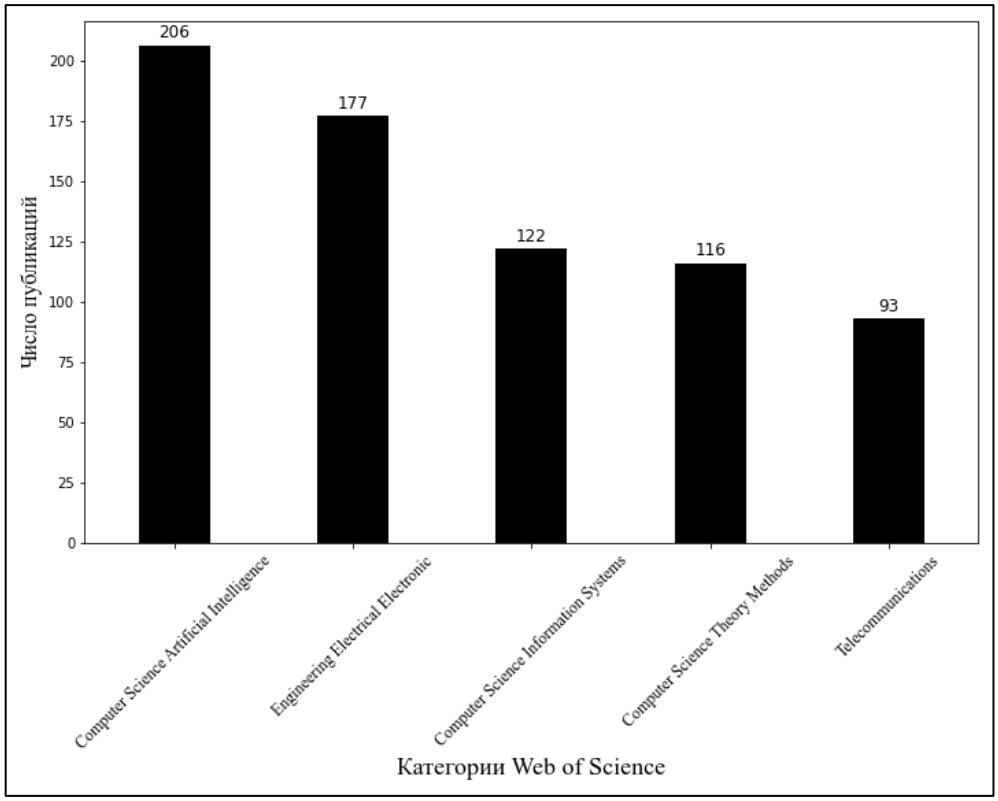


Рисунок 1.2 - **Распределение публикаций по категориям**

Большую часть занимают компьютерные науки (Computer Science) в подобластях «Искусственный интеллект» (Artificial Intelligence), «Информационные системы» (Information Systems) и «Теоретические методы» (Theory Methods).

Можно заметить, что значительная часть публикаций относится к областям телекоммуникаций и электроники, в которых решаются такие задачи, как прогнозирование нагрузки на электросети, нахождение аномальный значений во временных рядах для диагностики неполадок и многие другие. Для подобных задач свойственно использование больших объемов данных и требование к высокому качеству прогнозов, возможно именно поэтому сложные модели рекуррентных нейронных сетей получили свое распространение именно в этих областях. Статьи, которые были направлены на прогнозирование финансовых временных рядов, не выделялись в отдельную категорию, а попадали в одну из пяти, рассмотренных нами.

.

С помощью методов обработки текста мы выделили рейтинг биграм и триграм, встречающихся в аннотациях к публикациям из нашей выборки статей. Первые 20 наиболее часто используемых в публикациях комбинаций слов представлены на рисунке 1.3:

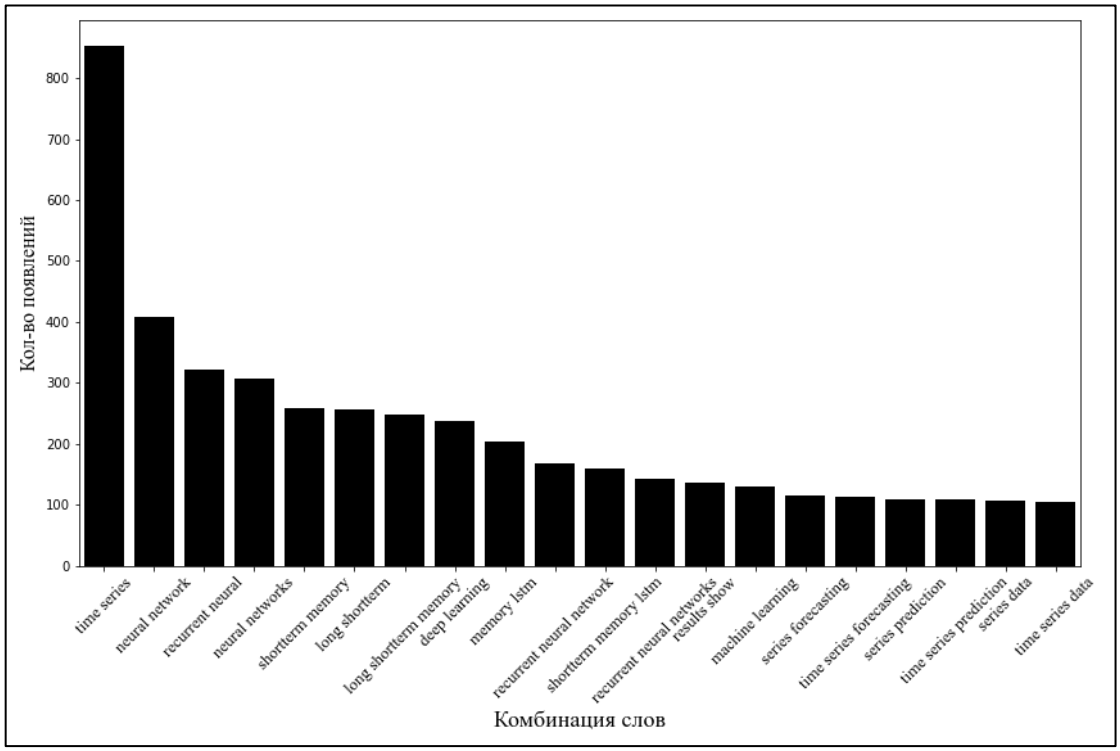


Рисунок 1.3 – **Диаграмма упоминаний словосочетаний в публикациях**

Ключевые слова действительно верно отражают содержание выбранных публикаций. Помимо этого, можно отметить, что большое число упоминаний в литературе имеют термины deep learning и machine learning, к которым в настоящее время нередко относят модели нейронных сетей.

1.1.2. Определение области поиска источников

Для поиска публикаций мы выбрали следующие электронные библиотеки и информационные базы:

1. eLibrary
2. Web of Science
3. Science Direct
4. JSTOR

Ключевые слова и словосочетания, характеризующие тематику НИР и ожидаемые результаты: временные ряды (time series), прогнозирование (forecasting), рекуррентная нейронная сеть (RNN), LSTM, финансы (finance).

Для того, чтобы ограничить область поиска, были определены атрибуты поиска:

1. Язык: Английский и Русский

2. Год публикации: 2016–2022
3. Атрибуты: заголовок и ключевые слова
4. Тип публикации:
    a. Статья в журнале (Journal article)
    b. Материалы конференций (Conference)

В результате анализа полученных статей мы выяснили, что публикации в основном разделяются на несколько групп:

1. Сравнение моделей RNN с другими эконометрическими моделями. Примером может служить статья[7], в которой сравнивали прогнозы нагрузки на электросети по моделям ARMA, SARIMA, ARMAX по метрике MAPE. Сравнение прогнозов показало, что сети LSTM при соответствующих данных превосходят традиционные методы не только для краткосрочного, но и для долгосрочного прогноза.
2. Результат применения моделей RNN для решения конкретной задачи прогнозирования. Примером публикации из данной группы служит статья[8], в которой авторы описывают результаты от применения модели RNN для задачи прогнозирования временных рядов в области загрязнения воздуха.
3. Описание новых подходов к построению моделей RNN (техники прогноза, архитектуры, оптимизации параметров, предобработки временного ряда и т.д.).
4. Обзорные статьи, описывающие и систематизирующие результаты работы разных исследователей.

Для нашего исследования мы отобрали несколько публикаций, в которых раскрывается поставленные нами вопросы. Их обзору посвящен следующий подраздел данной работы.

1.1.3. Обзор современного состояния области исследования

Как уже было сказано, с появлением большого количества данных нейронные сети становятся все более популярными. В статье[9] авторы приводят обзор большого количества работ, посвященных использованию нейронных сетей в задачах прогнозирования. Они отмечали, что нейронные сети имеют отличительную особенность: они легко адаптируются под широкий класс задач и позволяют аппроксимировать нелинейные зависимости данных, что отражает потенциал их использования для задач прогнозирования временных рядов.

---

[7] Muzaffar, Shahzad and A. Afshari. “Short-Term Load Forecasts Using LSTM Networks.” Energy Procedia 158 (2019): 2922-2927

[8] Алжеев, А. В. Сравнительный анализ прогнозных моделей ARIMA и LSTM на примере акций российских компаний / А. В. Алжеев, Р. А. Кочкаров // Финансы: теория и практика. – 2020. – Т. 24. – № 1. – С. 14-23. – DOI 10.26794/2587-5671-2020-24-1-14-23.

[9] Zhang, G., Eddy Patuwo, B., Hu, M. Y., 1998. Forecasting with artificial neural networks: The state of the art. Int. J. Forecast. 14, 35–62

RNN — это наиболее часто используемая архитектура нейронных сетей для задач прогнозирования последовательностей. Рекуррентные нейронные сети особенно приобрели популярность в области обработки текстовой информации. Подобно нейронным сетям прямого распространения, RNN также являются универсальными аппроксиматорами[10]. Однако, из-за обратных связей рекуррентные слои позволяют дополнительно учитывать временной порядок входных данных.

Наиболее популярными блоками RNN, используемыми для задач моделирования последовательностей, являются простой блок RNN Элмана, блок долговременной краткосрочной памяти LSTM и блок GRU. Блоки LSTM и GRU были разработаны специально для того, чтобы решить проблемы взрывного и исчезающего градиентов, которые были свойственны простому рекуррентному блоку RNN Элмана.

Выделяют несколько практических подходов к прогнозированию с помощью рекуррентных нейронных сетей:

- прогнозирование одномерных временных рядов;
- прогнозирование многомерных временных рядов.

В первом случае, в качестве прогнозных факторов берутся только прошлые значения временного ряда. Во втором, в них могут включаться лаги дополнительных рядов, выступающих в качестве прогнозных факторов, от которых по гипотезе исследователя зависит целевой показатель.

Согласно работе[11] существует пять стратегий построения прогноза: в первых трех из них выходом нейронной сети является число, в последних двух – вектор. Как отмечают авторы, в литературе стратегии были описаны раздельно с использованием различной терминологии. В статье[12] отмечается, что наиболее популярной стратегией в исследованиях является MiMo.

В публикациях можно найти ряд различных архитектур RNN предназначенных для решения задачи прогнозирования временных рядов. Мы можем выделить несколько их подвидов:

1. Stacked (Составная) RNN – это сеть, которая состоит из нескольких скрытых рекуррентных слоев и выходного линейного слоя для выполнения прогноза. В случае, когда модель содержит лишь один рекуррентный слой, такую архитектуру иногда называют Vanilla RNN.

---

[10] Schafer, A. M., Zimmermann, H. G., 10–14 Sep 2006. Recurrent neural networks are universal approximators. In: Proceedings of the 16th International Conference on Artificial Neural Networks - Volume Part I. ICANN'06. Springer-Verlag, Berlin, Heidelberg, pp. 632–640.

[11] S. Ben Taieb, G. Bontempi, A.F. Atiya, A. Sorjamaa, A review and comparison of strategies for multi-step ahead time series forecasting based on the NN5 forecasting competition, Expert Syst. Appl. 39 (8) (2012) 7067–7083.

[12] H. Hewamalage, C. Bergmeir, K. Bandara, Recurrent neural networks for time series forecasting: current status and future directions, Int. J. Forecast. (ISSN: 0169-2070) (2020).

2. Sequence to Sequence – разновидность архитектуры Encoder-Decoder, которую преобразовали под задачу прогнозирования. Подробно эта архитектура сравнивалась с предыдущей в статье[13].
3. CNN – свёрточные нейронные сети, широко используемые в задачах обработки изображений. Многие исследователи считают данный вид архитектуры потенциальной возможностью увеличить качество прогнозов, которые можно получить с при помощи нейронных сетей.

Специалисты могут также использовать гибридные архитектуры, сочетая преимущества того или иного типа нейронов. Однако подтверждение эффективности подобных подходов требует дополнительных исследований.

Так, в 2016 году Smyl и Kuber[14] усовершенствовали многоуровневую архитектуру рекуррентной нейронной сети, добавив туда межблочные соединения (skip connections). Данная модификация позволяет слоям, расположенным далеко внизу в стеке слоев сети, напрямую передавать информацию слоям, расположенным значительно выше, и таким образом минимизировать эффект исчезающего градиента. Впервые межблочные соединения были представлены в архитектуре ResNet, которая была первоначально разработана для распознавания изображений.

В 2018 году на международном соревновании по прогнозированию временных рядов M4 победила модель, которая комбинировала метод экспоненциального сглаживания и рекуррентную нейронную сеть[15], что подтвердило актуальность исследования данной области.

При этом, в статье[16], в которой описываются основные выводы по результатам соревнования M4, сказано, что отдельные модели машинного обучения показали себя хуже, чем ожидалось. Из-за этого авторы исследования указывают на необходимость дополнительной валидации статей, которые были направлены на доказательство эффективности методов машинного обучения в задачах прогнозирования временных рядов. Также авторы призывают к тому, чтобы подобные исследования проводились открыто и давали бы возможность доступа к данным и алгоритмам, которые использовались в работе.

---

[13] H. Hewamalage, C. Bergmeir, K. Bandara, Recurrent neural networks for time series forecasting: current status and future directions, Int. J. Forecast. (ISSN: 0169-2070) (2020).
[14] Smyl, S., Kuber, K., 19–22 Jun 2016. Data preprocessing and augmentation for multiple short time series forecasting with recurrent neural networks. In: 36th International Symposium on Forecasting.
[15] S. Smyl, A hybrid method of exponential smoothing and recurrent neural networks for time series forecasting, Int. J. Forecast. 36 (1) (2020) 75–85.
[16] Makridakis, S., Spiliotis, E., Assimakopoulos, V., 2020. The M4 Competition: 100,000 time series and 61 forecasting methods. Int. J. Forecast. 36 (2020) 54–74

Результаты соревнования M4 подтвердили выводы исследования[17], что самая точная ML модель будет в среднем хуже по точности, чем статистическая. Однако в настоящее время есть понимание, что эта проблема относится только к локальным моделям машинного обучения. В исследовании[18] авторы утверждают, что использование глобальных моделей рекуррентных нейронный сетей, которые обучаются на больших наборах временных рядов, позволяет значительно увеличить качество прогноза.

В противовес этому тезису в статье[19] отмечается, глобальные модели рекуррентных нейронных сетей часто страдают от выбросов в отдельных временных рядах. Вероятно, это связано с тем, что средние веса нейронной сети, найденные путем подгонки глобальных моделей, могут не подходить для индивидуальных требований определенных временных рядов. Следовательно, возникает необходимость в разработке новых моделей, которые включают как глобальные параметры, так и локальные параметры для отдельных временных рядов, в виде иерархических моделей, потенциально комбинируемых с ансамблем, где отдельные модели обучаются по-разному с существующим набором данных (например, на разных подмножествах). Работы Bandara[20], Smyl[21] являются шагами в этом направлении, но все же эта тема остается в значительной степени открытым исследовательским вопросом.

Bandara отмечает, что плохая эффективность моделей машинного обучения в прогнозировании одномерных временных рядов на соревнованиях скорее всего связана с тем, что временные ряды были слишком короткие, чтобы обучать на них сложные модели нейронных сетей. В такой ситуации более простые статистические модели, не чувствительные к шуму, как правило, будут работать хорошо.

Bandara предлагает преодолеть проблему накопления ошибок при переходе на глобальные модели, построив отдельную нейронную сеть для каждого кластера схожих временных рядов. В качестве основного метода авторы использовали рекуррентную нейронную сеть со слоем LSTM и стратегией построения прогноза MIMO. Авторы выбрали метод кластеризации временных рядов на основе 18 признаков, полученных через функцию tsmeasures из пакета anomalous-acm для языка R. В них входили: среднее значение, дисперсия, значение первого порядка автокорреляционной функции и другие. Далее временные ряды

[17] Makridakis, S., Spiliotis, E., & Assimakopoulos, V. (2018a). Statistical and machine learning forecasting methods: concerns and ways forward. PLoS One, 13(3), 1–26
[18] P. Montero-Manso, R.J. Hyndman, Principles and algorithms for forecasting groups of time series: locality and globality, Int. J. Forecast. (ISSN: 0169-2070) (2021).
[19] H. Hewamalage, C. Bergmeir, K. Bandara, Recurrent neural networks for time series forecasting: current status and future directions, Int. J. Forecast. (ISSN: 0169-2070) (2020).
[20] K. Bandara, C. Bergmeir, S. Smyl, Forecasting across time series databases using recurrent neural networks on groups of similar series: a clustering approach, Expert Syst. Appl. (ISSN: 0957-4174) 140 (2020) 112896.
[21] S. Smyl, A hybrid method of exponential smoothing and recurrent neural networks for time series forecasting, Int. J. Forecast. 36 (1) (2020) 75–85.

разделялись на кластеры с помощью таких методов, как K-Means, Partition Around Medoids (PAM), DBSCAN и Snob.

Также Bandara уделяет большое внимание предобработке временных рядов, давая рекомендации по поводу удаления сезонности из временного ряда, его логарифмирования, применения процедуры Бокса-Кокса и нормализации временного ряда. Их мы более подробно рассмотрим в разделе 1.3 нашей работы. В заключении статьи[22] Bandara делает вывод, что глобальные модели действительно превосходят по качеству локальные, при этом подход с построением отдельной модели для каждого кластера временных рядов дает наилучшие результаты.

В целом, глубокое обучение — это быстро развивающаяся область исследований, в которой быстро внедряется и обсуждается множество новых архитектур и методов. Однако для разработчиков прогнозных систем часто остается неясным, в каких ситуациях эти методы наиболее полезны и насколько трудно адаптировать их к конкретному случаю применения.

1.1.4. Прогнозирование финансовых временных рядов

Прогнозирование финансовых временных рядов является одной из популярных областей практического приложения науки анализа временных рядов. Исследователи опубликовали множество работ, показывающих варианты применения моделей рекуррентных нейронных сетей в этой сфере. Однако можно найти не так много обзорных статей, посвященных моделям нейронных сетей применимо к финансам.

Одной из таких работ является публикация[23], в которой рассматриваются модели глубокого обучения, такие как Deep Multilayer Perceptron (DMLP), RNN, LSTM, CNN, Restricted Boltzmann Machines (RBMs), DBN, Autoencoder (AE), DRL и их применение для прогнозирования финансовых временных рядов. В своей работе авторы рассматривают несколько подобластей: прогнозирование цен на акции, биржевые индексы, облигации, сырьевые товары и криптовалюты.

Также авторы отмечают, что публикации кластерезируются на две группы, основанные на сущности выхода модели: предсказание цены или предсказание тренда временного ряда. Вторую группу иногда выделяют как сопутствующую задачу для первой, ведь, чтобы предсказать направление тренда, нужно предсказать сами значения временного ряда. При этом некоторые исследователи рассматривают эту задачу в качестве самостоятельной и решают ее как задачу классификации, обходя тем самым задачу прогнозирования.

---

[22] K. Bandara, C. Bergmeir, S. Smyl, Forecasting across time series databases using recurrent neural networks on groups of similar series: a clustering approach, Expert Syst. Appl. (ISSN: 0957-4174) 140 (2020) 112896.

[23] Omer Berat Sezer, Mehmet Ugur Gudelek, Ahmet Murat Ozbayoglu, Financial time series forecasting with deep learning: A systematic literature review: 2005–2019, Applied Soft Computing, Volume 90, 2020, 106181

Автор отмечает, что LSTM и его вариации наряду с некоторыми гибридными моделями доминируют в области прогнозирования финансовых временных рядов, так как эти блоки по своей природе хорошо выделяют временные характеристики исследуемых рядов. Также некоторые исследователи рекомендуют нормализовывать временные ряды, удалять из них сезонную составляющую, либо преобразовывать временные ряды таким образом, чтобы результирующие финансовые данные становились стационарными с временной точки зрения.

Согласно статистике публикаций, приведенной в статье, более половины опубликованных работ по прогнозированию временных рядов относятся к категории моделей RNN. Анализ публикаций иллюстрирует доминирование рекуррентных нейронных сетей (которые в основном состоят из блоков LSTM). Независимо от типа проблемы: прогнозирования цены или направления тренда, последовательный характер данных заставлял исследователей рассматривать RNN, GRU и LSTM в качестве основных моделей. Также рекуррентные нейронные сети часто выбирались для сравнительного анализа производительности с другими моделями.

В публикации делается вывод, что большинство реализаций RNN для прогнозирования финансовых временных рядов выполнялось на языке программирования Python с помощью библиотек Keras и Tensorflow. Данный факт может свидетельствовать о потребности в разработке прогнозной системы с помощью RNN именно на этом языке. Также это свидетельствует о некотором изменении инструментария в сфере прогнозирования временных рядов относительно зарекомендовавших себя реализаций статистических методов на языке R, которые мы упоминали раньше.

В нашей работе мы рассмотрим только область прогнозирования цен на акции. Авторы публикации называют эту область одной из самых популярных среди финансовых временных рядов. Также в статье описываются публикации, где исследователи использовали в LSTM моделях для прогнозирования котировок различные макроэкономические показатели, интеллектуальный анализ новостей, показатели технического анализа и другие дополнительные источники данных. Данный факт отражает востребованность изучения моделей прогнозирования многомерных временных рядов, где статистические методы показывают себя менее эффективными.

Также авторы отмечают, что в большинстве исследований модели нейронных сетей становились точнее моделей машинного обучения. Однако было также много случаев, когда их показатели были сопоставимы. Авторы подчеркивают, что возможно это связано с большой популярностью нейронных сетей в настоящий момент, когда средняя модель глубокого обучения пользуется большим вниманием, чем лучшая статистическая модель или модель машинного обучения. Они связывают эту популярность с увеличением вычислительной

мощности устройств, доступностью больших объемов открытых данных и возможностью аппроксимации нелинейных функций нейронными сетями.

Далее рассмотрим практическое применение моделей LSTM в публикациях российских исследователей.

В статье[24] приводится сравнительный анализ моделей LSTM и ARIMA в прогнозировании курса биткоина к доллару на основе дневных данных с января 2015 года по декабрь 2018. Авторы выполняли прогноз на небольшой прогнозный горизонт (1 период вперед), без дополнительной предобработки временного ряда. При этом они получили достаточное качество прогноза, чтобы рекомендовать использовать рассматриваемые модели в задачах прогнозирования котировок криптовалюты на короткие горизонты.

В статье[25] исследователи рассмотрели 36 конфигураций архитектур LSTM-сетей для построения прогнозов длительностью до 70 шагов по данным, размер которых составляет 300–500 элементов. В качестве данных использовались математическое ожидание, дисперсия, коэффициенты асимметрии и эксцесса смесей плазмы в различные моменты времени.

Временные ряды приводились к диапазону от 0 до 1, а также разделялись методом скользящего окна. В работе авторы рассмотрели архитектуры с одним, двумя и тремя скрытыми слоями LSTM и пришли к выводам, что наибольший прирост точности получается в результате перехода от архитектуры с одним скрытым слоем к архитектуре с двумя скрытым слоями. В среднем архитектура с двумя слоями дает на 18% меньшую ошибку RMSE и на 20% меньшую ошибку MAE. При этом важную роль играет соотношение между прогнозным горизонтом и входным окном. Авторы эмпирически установили, что если данное отношение меньше 0.1, то эффект от использования архитектуры с двумя скрытыми слоями менее заметен. Также авторы отмечают, что, хотя среднесрочные прогнозы, построенные LSTM моделями, не всегда способны предсказать точную величину изменения временного ряда, модели данного типа достаточно точно предсказывают поведение тренда на протяжении десятков наблюдений (его рост или падение).

В статье[26] авторы решают задачу прогнозирования количества заявок в крупную федеральную информационную систему с учетом их характеристик. В данной работе исследователи делают вывод, что наименьшая ошибка наблюдается у нейронных сетей,

---

[24] Прогностические модели развития рынка криптовалюты (ARMA, LSTM): сравнительный анализ / А. А. Абдукаева, Л. А. Ельшин, А. М. Гильманов, В. В. Бандеров // Казанский экономический вестник. – 2019. – № 6(44). – С. 88-96.

[25] Горшенин, А. К. Анализ конфигураций LSTM-сетей для построения среднесрочных векторных прогнозов / А. К. Горшенин, В. Ю. Кузьмин // Информатика и ее применения. – 2020. – Т. 14. – № 1. – С. 10-16. – DOI 10.14357/19922264200102.

[26] Обрубов, М. О. Применение LSTM-сети в решении задачи прогнозирования многомерных временных рядов / М. О. Обрубов, С. Ю. Кириллова // Национальная Ассоциация Ученых. – 2021. – № 68-2. – С. 43-48.

которые тренировались с оптимизаторами adam и nadam и количеством LSTM блоков равным 20, при этом рассматривалась архитектура только с одним скрытым слоем.

В статье[27] авторы сравнивали качество прогнозов для моделей ARIMA и LSTM на основе значения котировок акций российских компаний: Алроса, Газпром, КамАЗ, НЛМК, Киви, Роснефть, ВТБ и Яндекс за период с 02.06.2014 по 11.11.2019 г. с разбивкой по неделям. Результаты исследования подтверждают превосходство модели LSTM, при которой среднеквадратическая ошибка RMSE на 65% меньше, чем при использовании модели ARIMA. Стоит отметить, что авторы данного исследования использовали локальные модели.

Можно сделать вывод, что применение методов глубокого обучения в финансовой сфере становится все более и более актуальным. Для любого крупного бизнеса, функционирующего в цифровую эпоху, крайне важно повышать качество планирования через использование передовых методов, в число которых входят модели рекуррентных нейронных сетей.

В следующем разделе мы рассмотрим основные методы и подходы, которые используются для прогнозирования с помощью моделей RNN.

1.2. Теоретические основы RNN

Как мы выяснили в предыдущем разделе, помимо классических статистических моделей исследователи все чаще и чаще обращают внимание на использование рекуррентных нейронных сетей для решения задачи прогнозирования временных рядов.

Данный вид нейронных сетей позволяет прогнозировать или классифицировать объекты на основе объясняющих факторов. Также рекуррентные нейронные сети наиболее эффективны в задачах обработки естественного языка (Natural Language Processing), где смысл текущего слова может зависеть от смысла предыдущих. Исследователи заметили, что это свойство также применимо к прогнозированию временных рядов, в которых подобным образом присутствуют автокорреляционные эффекты.

1.2.1. Проблемы классической рекуррентной нейронной сети

При использовании рекуррентных нейронных сетей алгоритм обратного распространения ошибки может работать неверно из-за рекуррентных связей между слоями. Этой проблемы возможно избежать, если развернуть сеть, создавая копии нейронов, имеющих рекуррентные связи. Данная модификация называется обратное распространение ошибки сквозь время (Backpropagation through time) и показана на рисунке 1.4.

[27] Алжеев, А. В. Сравнительный анализ прогнозных моделей ARIMA и LSTM на примере акций российских компаний / А. В. Алжеев, Р. А. Кочкаров // Финансы: теория и практика. – 2020. – Т. 24. – № 1. – С. 14-23. – DOI 10.26794/2587-5671-2020-24-1-14-23.

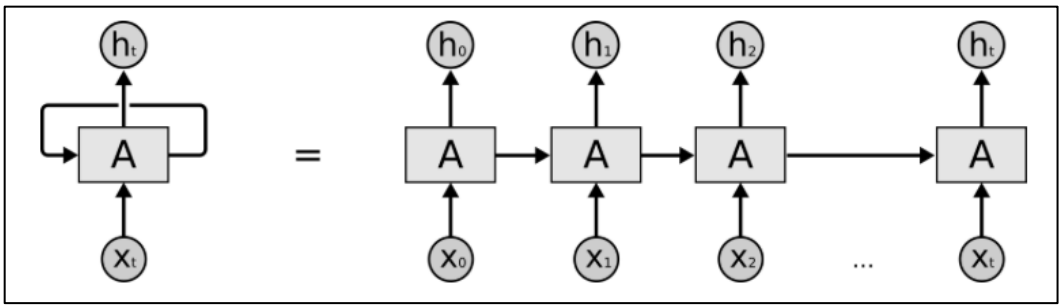


Рисунок 1.4 – **Backpropagation through time**

Развернутая данным образом нейронная сеть может быть очень глубокой, что может создать проблемы с вычислением градиента. Веса могут стать очень маленькими, что называется проблемой исчезающего градиента (Vanishing gradient problem), или наоборот очень большими, что называется проблемой взрывного градиента (Exploding gradient problem). На практике эти проблемы решаются с помощью использования нейронов вида Long short-term memory (LSTM) и Gated Recurrent Units (GRU), которые могут сохранять текущую информацию для дальнейшего использования.

В статье[28] авторы сравнивали влияние рекуррентных блоков LSTM, GRU и ERNN на качество прогноза. Исследователи заключили, что блок LSTM показывает наилучшую производительность, поэтому именно его мы будем рассматривать в рамках нашей работы. Также, мы уже упоминали, что применимо к экономическим временным рядам блок LSTM является наиболее популярным, что дополнительно отражает актуальность его использования.

1.2.2. Long short-term memory (LSTM)

Блоки LSTM были придуманы в 1997 году Зеппом Хохрайтером и Юргеном Шмидхубером[29]. В настоящее время LSTM чаще всего используют в задачах обработки текстовой информации (машинный перевод, генерация текста).

LSTM – это рекуррентный нейрон, который создан для того, чтобы обучаться понимать долгосрочные зависимости в данных. Уникальным является то, что в архитектуре LSTM есть блок памяти (memory cell), который способен обеспечивать свою активацию бесконечно. Блок памяти состоит из линейных нейронов, которые сохраняют его текущее состояние, и трех вентилей, которые могут открываться и закрываться в зависимости от состояния блока[30].

---

[28] H. Hewamalage, C. Bergmeir, K. Bandara, Recurrent neural networks for time series forecasting: current status and future directions, Int. J. Forecast. (ISSN: 0169-2070) (2020).
[29] Hochreiter S., Long short-term memory / S. Hochreiter, J. Schmidhuber // Neural computation - 1997. - № 9 (8) - pp. 1735–1780
[30] Schmidhuber J., Evolino: Hybrid Neuroevolution / J. Schmidhuber, D. Wierstra, F. J. Gomez // Optimal Linear Search for Sequence Learning. Proceedings of the 19th International Joint Conference on Artificial Intelligence (IJCAI), Edinburgh - 2005. - pp. 853—858.

Эти вентили реализованы в виде логистической функции для вычисления значения в диапазоне [0; 1]. Умножение на это значение используется для частичного допуска или запрещения потока информации внутрь и наружу памяти.

Например, «входной вентиль» контролирует меру вхождения нового значения в память, а «вентиль забывания» контролирует меру сохранения значения в памяти. «Выходной вентиль» контролирует меру того, в какой степени значение, находящееся в памяти, используется при расчёте выходной функции активации для блока. Идея заключается в том, что старое значение следует забывать тогда, когда появится новое, достойное запоминания).

Из-за этого в сетях типа LSTM могут быть проблемы с памятью из-за подавления новыми значениями старых. Однако в задаче прогнозирования временных рядов этот факт даже играет на руку исследователям, так как последние временные промежутки имеют большее влияние на прогноз модели.

Визуализация нейрона LSTM представлена на рисунке 1.5.

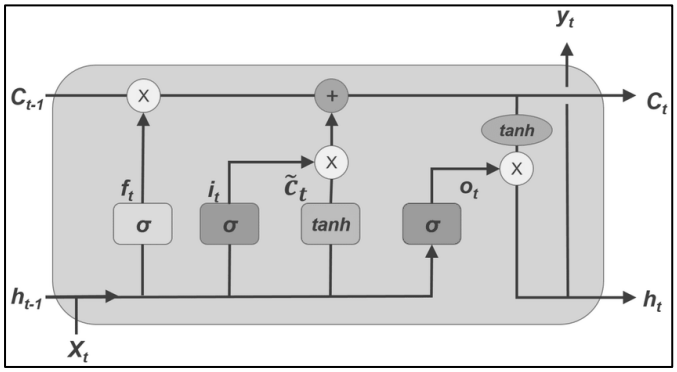


Рисунок 1.5– **Визуализация нейрона LSTM**

Каждый LSTM нейрон задается следующими формулами (1)-(7):

$$f_t = \sigma_g\left(W_f x_t + U_f * h_{t-1} + b_f\right) \quad (1)$$

$$i_t = \sigma_g(W_i x_t + U_i * h_{t-1} + b_i) \quad (2)$$

$$o_t = \sigma_g(W_o x_t + U_o * h_{t-1} + b_o) \quad (3)$$

$$c_t = f_t \circ c_{t-1} + i_t \circ \sigma_c(W_c x_t + U_c * h_{t-1} + b_c) \quad (4)$$

$$h_t = o_t \circ \sigma_h(c_t) \quad (5)$$

$$c_0 = 0 \quad (6)$$

$$h_0 = 0 \quad (7)$$

где

- x_t -входной вектор;
- h_t -выходной вектор;
- c_t- вектор состояний;
- W, U -матрицы весов;
- b - вектор свободного члена;
- f_t - вектор вентиля забывания, вес запоминания старой информации;

- i_t - вектор входного вентиля, вес получения новой информации;
- o_t- вектор выходного вентиля.

Функции активации:

- σ_g- на основе сигмоиды (формула (8));
- σ_c - на основе гиперболического тангенса (формула (9));
- σ_h- на основе гиперболического тангенса (формула (9)).

$$S(x) = \frac{1}{1 + e^{-x}} = \frac{e^x}{e^x + 1} \tag{8}$$

$$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{9}$$

Оператор $\circ$ обозначает произведение Адамара – бинарная операция над двумя матрицами одинаковой размерности, результатом которой является матрица той же размерности, в которой каждый элемент с индексами i, j — это произведение элементов с индексами i, j исходных матриц.

1.3. Предобработка временного ряда для моделей RNN

В данном разделе мы раскрываем теоретическую основу методов предобработки временного ряда, которые использовались в практической части нашей работы.

1.3.1. Метод скользящего окна

При использовании архитектуры Stacked RNN основным шагом предобработки временного ряда является метод скользящего окна, который представляет из себя следующую логику (см. Рисунок 1.6).

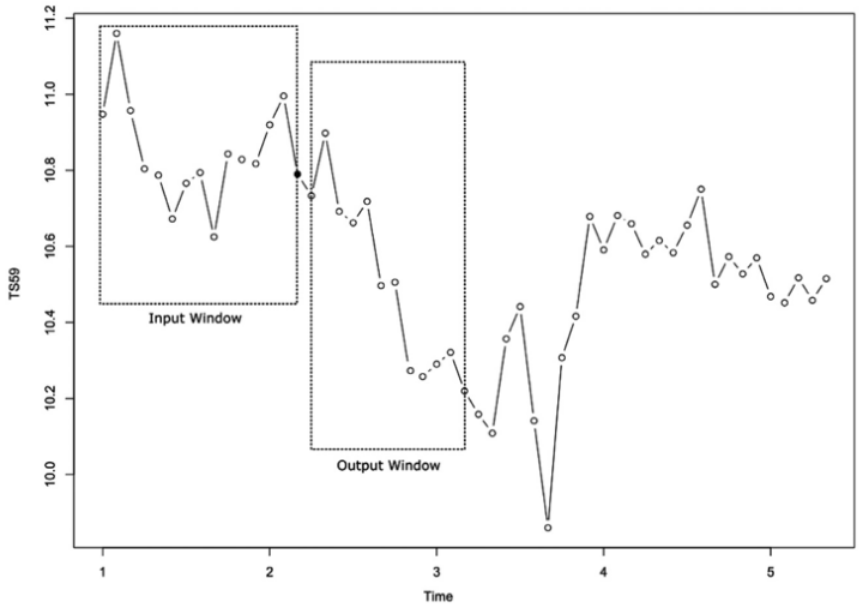


Рисунок 1.6 – **Пример применения методы скользящего окна**[31]

[31] K. Bandara, C. Bergmeir, S. Smyl, Forecasting across time series databases using recurrent neural networks on groups of similar series: a clustering approach, Expert Syst. Appl. (ISSN: 0957-4174) 140 (2020) 112896.

Сначала временной ряд длиной L преобразуется в участки длины (inputsize + outputsize). В общей сложности существует (L – outputsize –inputsize) таких участков. Здесь outputsize относится к длине выходного окна, в то время как inputsize представляет длину входного окна, используемого в каждом фрагменте обучающей выборки.

Обучающий набор данных генерируется путем повторения описанного выше процесса до тех пор, пока последняя точка окна ввода не будет расположена в L – outputsize.

Что касается конкретного выбора размера входных и выходных окон, то размер выходного окна в значительной степени определяется требуемым горизонтом прогноза и стратегии прогнозирования.

В статье[32] выбирали размер входного окна в зависимости от размера выходного окна и порядка сезонности по формуле (10):

$$InputSize = 1.25 * \max(outputsize, seasonal\_period) \quad (10)$$

То есть исследователи выбирали входное окно большего размера (с коэффициентом 1,25), чем размер выходного окна или продолжительность сезонного периода, в зависимости от того, что больше. Авторы подчеркивают, что этот выбор скорее эмпирический, и он кажется адекватным для ситуаций, когда полный период сезонности может быть охвачен относительно небольшим количеством точек данных. В ситуациях, когда необходимо прогнозировать ежедневный ряд с годовой сезонностью, лучше будет использовать больший размер входного окна. Кроме того, в ситуациях, когда временные ряды очень короткие, может потребоваться выбрать меньший размер входного окна, чтобы увеличить обучающую выборку.

Согласно результатам исследования[33], увеличение входного окна позволяет Stacked архитектуре захватить больше внутренних закономерностей временного ряда вне зависимости от того, была ли удалена сезонность или нет.

В экспериментах в рамках данной работы мы использовали 30 точек данных в качестве размера входного окна. Мы сделали предположение, что для прогнозирования будет достаточно информации о последнем месяце наблюдений. Используемые нами временные ряды нельзя назвать короткими, так как они содержат более 3000 ежедневных значений. Поэтому у нас не было жесткой необходимости уменьшать размер входного окна в пользу увеличения обучающей выборки.

[32] K. Bandara, C. Bergmeir, S. Smyl, Forecasting across time series databases using recurrent neural networks on groups of similar series: a clustering approach, Expert Syst. Appl. (ISSN: 0957-4174) 140 (2020) 112896.
[33] H. Hewamalage, C. Bergmeir, K. Bandara, Recurrent neural networks for time series forecasting: current status and future directions, Int. J. Forecast. (ISSN: 0169-2070) (2020).

1.3.2. Заполнение пропущенных значений

Из-за того, что рекуррентные нейронные сети не могут обрабатывать пропущенные значения, а в нашем источнике они имелись, перед нами встала задача их заполнения. Так как пропущенных значений было немного (<5%), для их заполнения мы воспользовались методом линейной интерполяции.

1.3.3. Удаление выбросов

В качестве метода, позволяющего удалить выбросы из временного ряда, мы использовали Hampel Filter из библиотеки sktime. Он представляет из себя следующий алгоритм:

Используется скользящее окно длины k, в рамках которого рассчитывается медиана и среднее абсолютное отклонение. Каждое наблюдение в рамках окна мы сравниваем с медианой и если их разность больше, чем m средних абсолютных отклонений, то мы считаем наблюдение выбросом и заменяем на медиану. Далее окно смещается в рамках ряда на одно наблюдение, и процедура повторяется.

В наших экспериментах мы использовали фиксированную длину окна k=10 и количество средних абсолютных отклонений m=3.

1.3.4. Удаление сезонности

На данный момент нет единого однозначного мнения насчет того, насколько точно рекуррентные нейронные сети способны моделировать сезонные колебания. Авторы исследования[34] считают, что при достаточной длине входного окна модели RNN способны моделировать сезонность, однако Bandara в статье[35] отмечает, что это не является гарантией того, что на практике они будут делать это достаточно точно. Он также приводит ряд исследований[36], в которых доказывается, что нейронные сети, обученные на рядах, очищенных от сезонности, показывают лучшую точность прогноза.

В наших экспериментах мы предполагали, что ряды имеют годовую мультипликативную сезонность. Мы сделали такое предположение, исходя из экономической сути используемых временных рядов. Сезонные колебания в котировках акций зачастую связаны с датами опубликования отчетности, датами выплаты дивидендов и другими периодичными мероприятиями. Мы предполагаем, что с увеличением капитализации компании подобные сезонные эффекты будут выражаться сильнее.

---

[34] H. Hewamalage, C. Bergmeir, K. Bandara, Recurrent neural networks for time series forecasting: current status and future directions, Int. J. Forecast. (ISSN: 0169-2070) (2020).
[35] K. Bandara, C. Bergmeir, S. Smyl, Forecasting across time series databases using recurrent neural networks on groups of similar series: a clustering approach, Expert Syst. Appl. (ISSN: 0957-4174) 140 (2020) 112896.
[36] Zhang, G. P., & Qi, M. (2005). Neural network forecasting for seasonal and trend time series. European Journal of Operational Research, 160(2), 501–514. ISSN: 0377- 2217.

В нашей работе в качестве функции, удаляющей сезонность, мы использовали Deseasonalizer из библиотеки sktime. Эта функция реализует метод сезонной декомпозиции временного ряда с помощью скользящего среднего. С подробным описанием этого метода можно ознакомиться в книге Роба Хиндмана[37].

1.3.5. Нормализация

Функции активации, используемые в ячейках RNN, такие как сигмоида и гиперболический тангенс, имеют область насыщения, после которой выходные данные остаются постоянными. Следовательно, при использовании ячеек RNN следует быть уверенным, что подаваемые входные данные нормализованы надлежащим образом, чтобы выходные данные не находились в диапазоне насыщения[38].

Hewamalage и др. в своей статье[39] используют локальную нормализацию в рамках входного скользящего окна. В статьях российских исследователей в основном применяется метод нормализации путем приведения всех значений временного ряда к диапазону от 0 до 1. Это преобразование происходит по формуле (11):

$$x_i^* = \frac{x_i - \min(X)}{\max(X) - \min(X)} \tag{11}$$

В практической части нашей работы мы использовали именно этот метод, реализованный в функции MinMaxScaler из библиотеки sklearn.

1.3.6. Стабилизация дисперсии ряда

Самыми популярными методами стабилизации дисперсии во временном ряду являются логарифмирование и преобразование Бока-Кокса.

Bandara в статье отмечает, что логарифмическое преобразование является сильно нелинейным и обычно используется с осторожностью, поскольку небольшие различия в логарифмическом пространстве могут привести к большим различиям в исходном пространстве, из-за чего обучение модели может привести к неоптимальным результатам. Поэтому в прогнозировании временных рядов популярнее более консервативный метод Бокса-Кокса, который преобразует временной ряд по следующей формуле (12):

$$w_t = \begin{cases} \log(y_t), & \lambda = 0 \\ \frac{y_t^{\lambda} - 1}{\lambda}, & \lambda \neq 0 \end{cases} \tag{12}$$

---

[37] Hyndman, R.J. Forecasting: principles and practice [Электронный ресурс] // R.J.Hyndman, G. Athanasopoulos ; OTexts: Melbourne, Australia, - 2nd edition - 2018. - Режим доступа: www.otexts.com/fpp2, свободный (дата обращения: 03.05.2022)

[38] Smyl, S., Kuber, K., 19–22 Jun 2016. Data preprocessing and augmentation for multiple short time series forecasting with recurrent neural networks. In: 36th International Symposium on Forecasting.

[39] H. Hewamalage, C. Bergmeir, K. Bandara, Recurrent neural networks for time series forecasting: current status and future directions, Int. J. Forecast. (ISSN: 0169-2070) (2020).

Преобразование Бокса-Кокса представляет собой комбинацию логарифмического преобразования (когда $\lambda = 0$) и степенного преобразования (когда $\lambda \neq 0$), обозначаемого y_t^{λ}. Параметр λ должен быть тщательно подобран. В наших экспериментах мы воспользовались программной реализацией этого метода BoxCoxTransformer из библиотеки sktime, в которой в качестве оптимизатора параметра λ использовался метод максимального правдоподобия.

1.3.7. Переход к разностям

Взятие первой разности зачастую позволяет привести временной ряд к стационарной форме. В этом случае модели требуется научиться прогнозировать прирост целевого фактора, что позволяет снизить влияние возрастающего тренда, присущего большинству финансовых временных рядов.

Многие эконометрические методы чувствительны к нестационарным временным рядам. Одна из задач нашего первого эксперимента состоит в том, чтобы оценить чувствительность модели рекуррентной нейронной сети к данному преобразованию.

1.4. Стратегии прогнозирования с помощью RNN

Согласно исследованию[40] существует пять стратегий построения прогноза, в первых трех из них выходом нейронной сети является число, в последних двух – вектор. Как отмечает автор, в литературе данные стратегии были описаны раздельно, зачастую используя различную терминологию.

Recursive (рекурсивная стратегия). Прогнозирование происходит циклическим добавлением полученного прогноза во вход модели на следующем шаге. Главный недостаток этой стратегии состоит в том, что происходит накопление ошибки с каждым новым значением прогнозного горизонта. Поэтому она подходит только для прогнозов на короткий период.

Direct (прямая стратегия) – для построения прогноза строится H моделей, каждая из которых совершает прогноз на i-ое значение из прогнозного периода, где $i \in \{1, \ldots, H\}$. Предположим, для прогноза температуры на улице на два дня вперед обучались бы две модели RNN: первая - прогнозировать на завтра, а вторая на послезавтра.

В данной стратегии не происходит накопление ошибки, так как при прогнозировании не используются полученные ранее модельные значения. Но у нее имеются два существенных недостатка: из-за раздельного вычисления прогноза, данный метод может ошибаться в формировании тренда, выдавая ломанную кривую прогноза, а также данная стратегия требует

[40] S. Ben Taieb, G. Bontempi, A.F. Atiya, A. Sorjamaa, A review and comparison of strategies for multi-step ahead time series forecasting based on the NN5 forecasting competition, Expert Syst. Appl. 39 (8) (2012) 7067–7083.

значительно больших вычислительных ресурсов, так как вместо одной модели необходимо обучать несколько.

DirRec – третья стратегия является комбинацией двух предыдущих. В данной технике производится прогноз для каждого значения из прогнозного горизонта с помощью отдельной модели (как в Direct технике), при этом на каждом шаге увеличивается набор входных данных путем добавления прогноза предыдущего шага (как в рекурсивной технике).

Multi-Input Multi-Output (MIMO) – выход нейронной сети является вектором длиной прогнозного периода. Обоснование стратегии MIMO заключается в сохранении между прогнозируемыми значениями стохастической зависимости, характеризующей временной ряд. Эта стратегия позволяет избежать предположения об условной независимости прогнозных значений, сделанного Direct стратегией, а также накопления ошибок, от которых страдает рекурсивная стратегия. Однако при использовании одной модели имеется недостаток ограничения максимального горизонта прогнозирования. При возникновении необходимости прогнозировать на большее количество периодов вперед модель придется переобучать заново. Несмотря на это стратегия MIMO является наиболее популярной среди исследователей и аналитиков данных.

DIRMO – данная стратегия является улучшением моделей DIRect и miMO, где прогноз на горизонт H разбивается на блоки, и каждый блок прогнозируется с помощью MIMO стратегии. Прогнозирование на H периодов вперед разбивается на n multi-out задач, где n=H/s для размера выхода s $\in \{1, ..., H\}$. Параметр s можно варьировать для получения оптимального значения.

В статье[41] приводятся формулы для прогнозирования на 4 значения вперед каждой стратегией (рисунок 1.7)

	$\hat{y}_{N+1}$	$\hat{y}_{N+2}$	$\hat{y}_{N+3}$	$\hat{y}_{N+4}$
Recursive	$\hat{f}(y_N,...,y_{N-d+1})$	$\hat{f}(\hat{y}_{N+1},y_N,...,y_{N-d+2})$	$\hat{f}(\hat{y}_{N+2},\hat{y}_{N+1},...,y_{N-d+3})$	$\hat{f}(\hat{y}_{N+3},\hat{y}_{N+2},...,y_{N-d+4})$
Direct	$\hat{f}_1(y_N,...,y_{N-d+1})$	$\hat{f}_2(y_N,...,y_{N-d+1})$	$\hat{f}_3(y_N,...,y_{N-d+1})$	$\hat{f}_4(y_N,...,y_{N-d+1})$
DirRec	$\hat{f}_1(y_N,...,y_{N-d+1})$	$\hat{f}_2(\hat{y}_{N+1},y_N,...,y_{N-d+1})$	$\hat{f}_3(\hat{y}_{N+2},\hat{y}_{N+1},...,y_{N-d+1})$	$\hat{f}_4(\hat{y}_{N+3},\hat{y}_{N+2},...,y_{N-d+1})$
MIMO	$\hat{F}(y_N,...,y_{N-d+1})$			
DIRMO $(s=2)$	$\hat{F}_1(y_N,...,y_{N-d+1})$		$\hat{F}_2(y_N,...,y_{N-d+1})$	

Рисунок 1.7 – **Формулы формирования прогноза различными стратегиями**

Также в статье приводится оценка времени для обучения модели с помощью каждой стратегии, которая представлена на рисунке 1.8.

[41] S. Ben Taieb, G. Bontempi, A.F. Atiya, A. Sorjamaa, A review and comparison of strategies for multi-step ahead time series forecasting based on the NN5 forecasting competition, Expert Syst. Appl. 39 (8) (2012) 7067–7083.

	Number of Models	Types of models	Size of output	Computational time
Recursive	1	SO	1	$1 \times T_{SO}$
Direct	H	SO	1	$H \times T_{SO}$
DirRec	H	SO	1	$H \times (T_{SO} + \mu)$
MIMO	1	MO	H	$1 \times T_{MO}$
DIRMO	$\frac{H}{s}$	MO	s	$\frac{H}{s} \times T_{MO}$

Рисунок 1.8 – **Сравнение времени обучения модели с каждой стратегией**

H – длина прогнозного периода

SO – single output (выход модели – число)

MO – multi output (выход модели – вектор)

T_{SO} – время обучения модели типа SO

T_{MO} – время обучения модели типа MO

Исходя из этого, авторы статьи делают вывод, что по временным затратам стратегии распределяются следующим образом (рисунок 1.9)

$$\underbrace{1 \times T_{SO}}_{Recursive} < \underbrace{1 \times T_{MO}}_{MIMO} < \underbrace{\frac{H}{s} \times T_{MO}}_{DIRMO} < \underbrace{H \times T_{SO}}_{Direct} < \underbrace{H \times (T_{SO} + \mu)}_{DirRec}$$

Рисунок 1.9 – **Ранжирование стратегий по времени обучения**

Авторы статьи провели сравнительный анализ стратегий на данных соревнования N5 и сформулировали следующие выводы:

1. Стратегии с множественным выходом (MIMO и DIRMO) являются лучшими. Они превосходят по точности стратегии с единственным выходом, такие как Direct, Recursive и DirRec. При этом стратегии MIMO и DIRMO обеспечивают сопоставимую производительность.
2. Для DIRMO выбор параметра s имеет решающее значение, поскольку он оказывает большое влияние на производительность. Если бы существовал улучшенный подход к отбору, эта стратегия имела бы большой потенциал.
3. Среди стратегий с единственным выходом стратегия Recursive почти всегда имеет меньший размер и более высокую точность, чем стратегия Direct.
4. DirRec в целом является худшей стратегией и дает особенно низкую точность, когда не выполняется предварительное удаление сезонности из временного ряда. Более того она является наиболее затратной по необходимым ресурсам и времени обучения.

Мы реализовали каждую из рассмотренных стратегий в рамках библиотеки TS_RNN для языка Python[42]. В разделе 2.3 практической части нашей работы мы оцениваем влияние каждой из стратегий на качество прогноза применительно к финансовым временным рядам.

1.5. Подходы к обучению одной RNN на нескольких временных рядах

До недавнего времени основные методы прогнозирования временных рядов были направлены на одномерные локальные модели, в которых модель обучалась только на лагах исходного временного ряда. За последние пару лет глубокое обучение проникло в область прогнозирования временных рядов и принесло много интересных инноваций. Во-первых, это позволило создавать более точные модели, которые потенциально могут фиксировать больше закономерностей, а также работать с многомерными временными рядами. Во-вторых, эти модели также потенциально могут быть обучены на нескольких связанных временных рядах.

Существует два основных метода учесть информацию из группы временных рядов в одной модели рекуррентной нейронной сети. Первым является построение многомерной прогнозной модели.

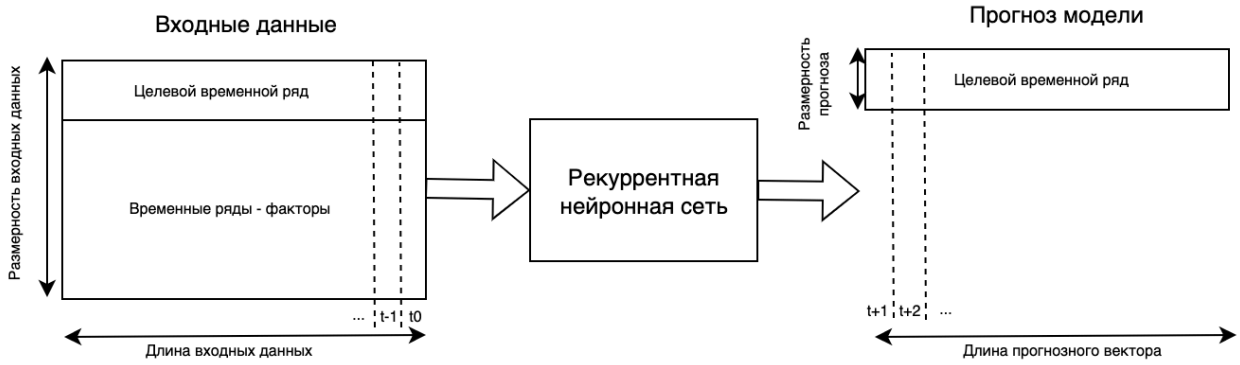


Рисунок 1.10 – **Схема выполнения прогноза многомерной RNN**

Процесс построения прогноза для многомерных моделей представлен на рисунке 1.10. В данном подходе временные ряды, подаваемые модели на вход, разделяют на два вида: целевой ряд — это ряд, который нам необходимо прогнозировать, учитывая его историю, и ряды факторы — другие временные ряды, прогноз которых нам не интересен, но которые потенциально могут помочь спрогнозировать целевой показатель.

В данном случае, после преобразования методом скользящего окна на вход в модель рекуррентной нейронной сети поступают трехмерные массивы, где первая размерность – количество полученных в результате преобразования блоков, вторая – количество лагов временных рядов, которые мы используем для прогнозирования и третья – само количество временных рядов, включая целевой.

[42] Программный код библиотеки TS_RNN [Электронный ресурс] - Режим доступа: www.github.com/LevPerla/Time_Series_Prediction_RNN, (дата обращения: 11.04.2022)

Нужно отметить, что также существует подвид многомерных моделей, которые помимо прошлых значений временных рядов включают и текущие значения прогнозных факторов. Таким образом модель становится частично регрессионной. В нашей работе подобный вид моделей мы рассматривать не будем.

В статье[43], представляющей собой обзорное исследование области прогнозирования финансовых временных рядов, говорится, что хотя в статьях лаги целевого временного ряда использовались для прогнозирования почти всегда, большое количество исследователей включало в свои модели дополнительную информацию из других источников. Согласно выводам авторов исследования, факторы фундаментального анализа и технического анализа были одними из наиболее благоприятных вариантов для исследований прогнозирования акций и индексов.

Между тем, в последние годы большую популярность получил интеллектуальный анализ финансовых текстов, в основном для извлечения информации о возможном поведении инвесторов. Непрерывный поток финансовых новостей, твитов, заявлений и блогов позволяет исследователям создавать более совершенные модели прогнозирования, сочетая числовые и текстовые данные. Общая методология включает в себя извлечение показателей финансовых настроений с помощью методов обработки естественного языка (Natural Language Processing) и объединение этой информации с данными фундаментального и технического анализа для достижения лучшей общей производительности прогнозной системы.

Вторым подходом является переход к глобальным моделям нейронных сетей. В отличие от подхода многомерных моделей, применение глобальных моделей к набору временных рядов не указывает на какую-либо взаимозависимость между ними в отношении прогноза. Глобальная модель, обученная по набору временных рядов, обычно работает независимо от отдельных рядов одномерным образом при составлении прогнозов, однако этот факт не ограничивает использование многомерных глобальных моделей.

В современных задачах прогнозирования часто требуется составлять прогнозы для групп временных рядов, которые могут иметь схожие закономерности, в отличие от прогнозирования только одного временного ряда. Одним из распространенных примеров в области розничной торговли является составление прогнозов для аналогичных продуктов в разных торговых точках. В таких сценариях глобальные модели могут продемонстрировать свой истинный потенциал, захватывая временные закономерности из нескольких похожих временных рядов, чтобы включить больше информации по сравнению с локальными методами.

---

[43] Omer Berat Sezer, Mehmet Ugur Gudelek, Ahmet Murat Ozbayoglu, Financial time series forecasting with deep learning: A systematic literature review: 2005–2019, Applied Soft Computing, Volume 90, 2020, 106181

Сложность прогнозных систем, состоящих из локальных моделей, увеличивается с увеличением количества необходимых для прогноза временных рядов, поскольку для каждого из них подбирается отдельная модель, в то время как сложность глобальных моделей остается неизменной независимо от количества рядов.

Однако при обучении на основе временных рядов с различной структурой точность глобальной модели может ухудшиться, и поэтому необходимо каким-то образом учитывать сходство между временными рядами. Bandara в статье[44] предлагает прогнозную систему, которая использует информацию о перекрестных рядах путем построения отдельных моделей для подгрупп временных рядов, определенных методом кластерного анализа. В заключении автор делает вывод, что это действительно позволяет увеличить точность глобальных моделей. Тем не менее, также наблюдалось, что количество групп временных рядов, полученных в результате кластерного анализа, влияет на конечную производительность модели. Это связывается с потерей информации, которая может произойти в результате группировки временных рядов.

В рамках нашего фреймворка мы также реализовали оба подхода к обучению одной нейронной сети на основе нескольких временных рядов[45]. В разделе 2.4 практической части нашей работы мы оцениваем влияние каждого из них на качество прогноза применительно к экономическим временным рядам.

Выводы

В первой главе мы занимались решением задач из теоретического блока. Был проведен количественный и качественный анализ области исследования, в результате которого было выяснено, что область прогнозирования временных рядов с помощью рекуррентных нейронных сетей в настоящий момент находится на своем пике. Рост популярности моделей RNN связан с результатами соревнования M4, на котором с помощью глобальной рекуррентной нейронной сети в сочетании с методом экспоненциального сглаживания удалось добиться наибольшей точности.

Также были раскрыты основные тенденции применения RNN моделей для прогнозирования экономических временных рядов. Исследователи рекомендуют использовать блоки LSTM в качестве основы для архитектуры моделей RNN, производить нормализацию временных рядов перед обученным моделей, а также использовать дополнительную экономическую информацию для улучшения точности прогноза.

[44] K. Bandara, C. Bergmeir, S. Smyl, Forecasting across time series databases using recurrent neural networks on groups of similar series: a clustering approach, Expert Syst. Appl. (ISSN: 0957-4174) 140 (2020) 112896.
[45] Программный код библиотеки TS_RNN [Электронный ресурс] - Режим доступа: www.github.com/LevPerla/Time_Series_Prediction_RNN, (дата обращения: 11.04.2022)

Отмечается, что основным языком программирования, используемым для решения задачи прогнозирования финансовых временных рядов является язык Python.

Во втором разделе первой главы были раскрыты основные проблемы рекуррентных нейронных сетей и возможные способы их избежать. Также в нем была представлена математическая модель нейрона LSTM, который мы используем в нашей работе.

Мы рассмотрели теоретическую основу методов предобработки временных рядов, применяемых в экспериментальной и практической части нашей работы.

Было представлено теоретическое описание стратегий прогнозирования временных рядов вместе с оценкой времени их выполнения и рекомендаций исследователей по их использованию.

Также мы рассмотрели, используемые в рамках третьего эксперимента, два подхода к обучению одной модели рекуррентной нейронной сети на нескольких временных рядах, представили основные предпосылки и ограничения данных подходов.

ГЛАВА 2. ПРАКТИЧЕСКАЯ ЧАСТЬ

2.1. Методология экспериментов

Для реализации задач нашего исследования мы провели три эксперимента:

1. Сравнительный анализ влияния техник предобработки временного ряда для прогнозирования с помощью RNN.
2. Сравнительный анализ влияния стратегий прогнозирования для моделей RNN.
3. Сравнительный анализ использования в модели RNN информации из нескольких временных рядов.

В каждом эксперименте мы использовали один набор из 8 временных рядов, которые представляли цены закрытия акций российских и иностранных компаний с 2012 по 2021 год по дням. Список компаний и их тикеров представлен в таблице 2.1.

Таблица 2.1

Компании, отобранные для экспериментов

Тикер	Название	Отрасль	Фондовая биржа
YNDX.ME	Яндекс	Информационные технологии	Московская биржа
SBER.ME	Сбербанк	Финансовый сектор	Московская биржа
POLY.ME	Полиметалл	Металлургия	Московская биржа
SIBN.ME	Газпром нефть	Нефтегазовая отрасль	Московская биржа
AMZN	Amazon	Розничная торговля	NASDAQ
AAPL	Apple	Электроника	NASDAQ
GOOGL	Google	Информационные технологии	NASDAQ
NFLX	Netflix	Кинопроизводство, телепроизводство	NASDAQ

Данные о котировках были получены с помощью открытого API сервиса yahoo finance[46].

2.1.1. Инициализация модели

В первом и третьем экспериментах строилась модель рекуррентной нейронной сети по стратегии MiMo, во втором стратегии итеративно перебирались.

В каждом эксперименте на вход модели подавался вектор длиной 30, который захватывал один месяц наблюдений. Прогнозным горизонтом являлась одна неделя (7 наблюдений).

В нашей работе мы использовали архитектуру Stacked RNN с одним рекуррентным слоем:

1. Слой LSTM (количество нейронов и функция активации подбирались как гиперпараметры)
2. Слой Dropout (коэффициент подбирался как гиперпараметр)

[46] Информационная база финансовой информации yahoo finance [Электронный ресурс] - Режим доступа: www.finance.yahoo.com, (дата обращения: 11.04.2022)

3. Линейный слой (количество нейронов зависит от стратегии прогнозирования)

В статье[47] авторы сравнивали эффективность архитектур Stacked и Sequence to Sequence, о которых мы упоминали ранее. В результате исследователи сделали вывод, что хотя Sequence to Sequence RNN в некоторых случаях показывала сопоставимое качество, в среднем Stacked RNN дает более точные прогнозы. Именно поэтому мы выбрали ее в качестве основы для наших экспериментов.

Также сравнивались рекуррентные блоки LSTM, GRU, ERNN. Авторы заключили, что блок LSTM показывает наилучшую производительность, поэтому именно его мы включили в нашу архитектуру.

Слой Dropout мы добавили для регуляризации параметров модели, а также для борьбы с переобучением.

2.1.2. Обучение модели

В процессе обучения моделей использовалась минимизация средней абсолютной ошибки прогноза MAE, представленной на формуле (13), с помощью оптимизационного алгоритма ADAM.

$$MAE = \frac{\sum_{i=1}^{n} |y_i - y_i^*|}{n} \tag{13}$$

Обучение каждой модели занимало 40 эпох, при этом был настроен алгоритм ранней остановки в случае, если ошибка переставала уменьшаться на протяжении 10 эпох.

2.1.3. Подбор гиперпараметров

Количество нейронов в скрытом слое нейронной сети, функция активации и Dropout rate подбирались с помощью метода Байесовой оптимизации, который реализован в рамках библиотеки keras-tuner[48]. С границами подбираемых параметров можно ознакомиться в приложении А. В программной реализации нашей библиотеки подбор параметров автоматически осуществляется внутри метода fit модели рекуррентной нейронной сети, поэтому в дальнейшем представлении алгоритмов экспериментов мы не будем выделять его в качестве отдельного этапа.

В экспериментах проводилось 5 итераций подбора для каждой из обучаемых моделей.

2.1.4. Кросс-валидация

Для оценки качества алгоритмов машинного обучения часто использует метод кросс-валидации, в котором известные данные делятся на три части: тренировочную,

[47] H. Hewamalage, C. Bergmeir, K. Bandara, Recurrent neural networks for time series forecasting: current status and future directions, Int. J. Forecast. (ISSN: 0169-2070) (2020).
[48] Документация фреймворка keras-tuner [Электронный ресурс] - Режим доступа: www.github.com/keras-team/keras-tuner, (дата обращения: 11.04.2022)

валидационную и тестовую. Тренировочная выборка является основой для обучения модели. На валидационной выборке оценивают качество модели в процессе обучения, а также во время подбора гиперпараметров. На тестовой выборке измеряют итоговую ошибку прогноза. Данный подход используется для того, чтобы оценить качество модели на новых данных, которые не использовались в процессе обучения и тем самым снизить вероятность переобучения модели.

Для временных рядов из-за автокорреляционных эффектов нельзя использовать методы кросс-валидации, основанные на перемешивании данных. Поэтому исследователи применяют специальный метод TimeSeriesSplit, алгоритм которого рассмотрим далее.

Пусть n_splits – количество итераций кросс-валидации, val_len – длина валидационной выборки, а test_len – длина тестовой выборки.

На первой итерации из временного ряда удаляются последние (n_splits - 1) * (val_len + test_len) наблюдений. Мы будем называть полученные данные кросс-валидационной выборкой (cv_sample).

Последние test_len наблюдений из cv_sample назначаются тестовой выборкой.

val_len наблюдений из cv_sample перед тестовой выборкой назначаются валидационной выборкой.

Оставшиеся наблюдения из cv_sample назначаются тренировочной выборкой.

На следующей итерации в конец cv_sample добавляются val_len + test_len наблюдений из исходного ряда и происходит переформирование выборок.

Визуализацию этого процесса можно увидеть на рисунке 2.1.

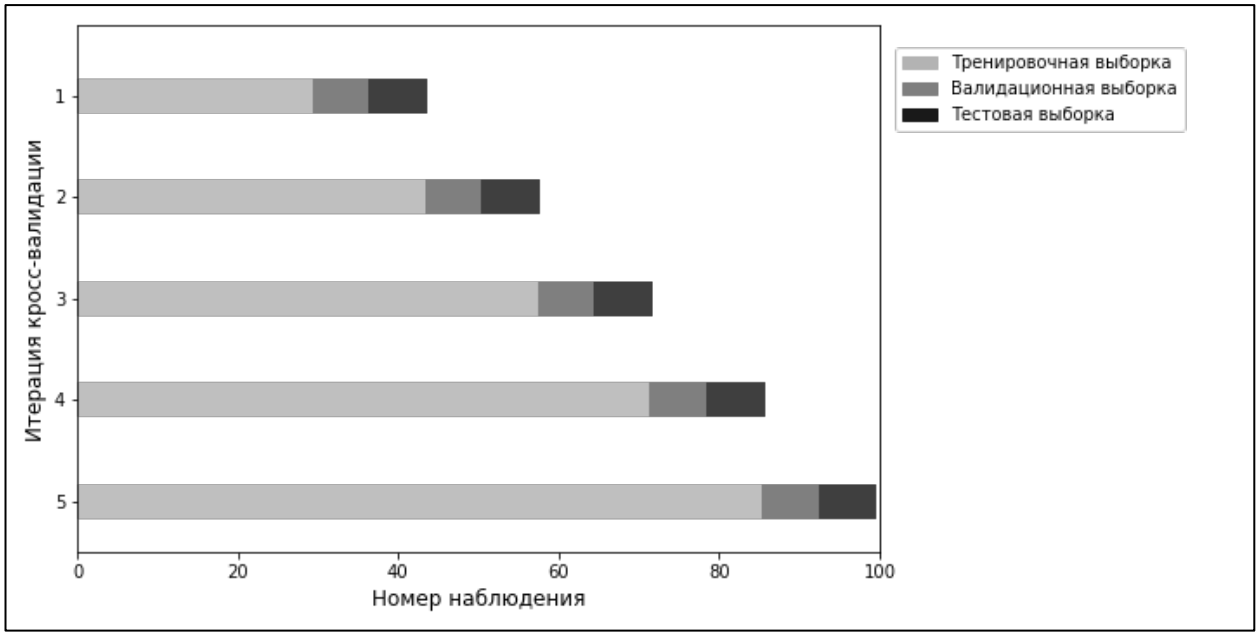


Рисунок 2.1 – **Визуализация метода TimeSeriesSplit**

В нашей работе мы использовали длины валидационной и тестовой выборок равные семи наблюдениям, что совпадало с длиной выбранного прогнозного периода. В каждом из наших экспериментов мы проводили 5 итераций кросс-валидации для оценки качества исследуемых моделей. Мы считаем, что этого достаточно, чтобы захватить разные промежутки монотонности временного ряда и делать на их основании выводы об эффективности моделей.

2.1.5. Метрика качества

Так как временные ряды имели разный масштаб, в качестве целевой метрики мы выбрали среднюю абсолютную процентную ошибку MAPE, которая представлена на формуле (14).

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{y_i - y_i^*}{y_i}\right| \tag{14}$$

Также мы фиксировали симметричную среднюю абсолютную процентную ошибку SMAPE (формула (15)), которая также является основной во многих исследованиях.

$$SMAPE = \frac{1}{n}\sum_{i=1}^{n}\frac{|y_i - y_i^*|}{(|y_i| + |y_i^*|)/2} \tag{15}$$

Значения обеих метрик фиксировались на каждой итерации кросс-валидации каждого исследуемого временного ряда отдельно на валидационной и тестовой выборках.

2.1.6. Базовая модель

В качестве базовой прогнозной модели мы использовали классическую эконометрическую модель SARIMA из библиотеки для языка Python pmdarima. Данная реализация позволяет в автоматическом режиме подбирать необходимые гиперпараметры модели по информационному критерию Акаике, на подобие тому, как это реализовано в библиотеке forecast для языка R.

Прогноз этой модели будет рассматриваться нами в качестве базового для сравнения с моделями рекуррентных нейронных сетей.

2.2. Сравнительный анализ влияния методов предобработки временных рядов на прогноз моделями RNN.

В данном разделе мы описываем эксперимент с оценкой влияния методов предобработки временных рядов на качество прогноза рекуррентных нейронных сетей.

2.2.1. Методология эксперимента

В первом эксперименте использовались только локальные рекуррентные нейронные сети, для каждого временного ряда из таблицы 2.1 строилась отдельная модель.

Основной целью нашего первого эксперимента являлась оценка влияния методов предобработки временных рядов, которые мы рассмотрели в разделе 1.3, а также их комбинаций на точность прогнозирования финансовых временных рядов с помощью моделей RNN. С конфигурацией первого эксперимента можно ознакомиться в приложении Б.

Путь движения данных в рамках эксперимента представлен на рисунке 2.2.

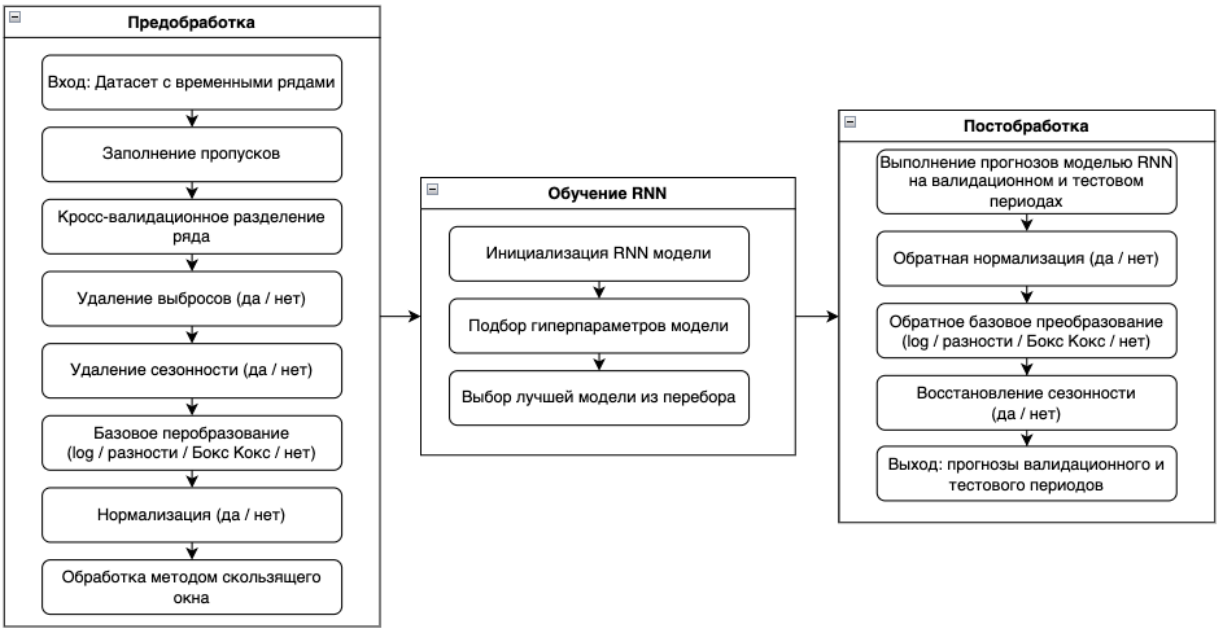


Рисунок 2.2 – **Этапы работы с данными в первом эксперименте**

Перед тем, как тот или иной временной ряд подавался в модель для обучения, он проходил через определенные этапы предобработки. Некоторые из них, такие как заполнение пропусков, кросс-валидация и разделение методом скользящего окна, применялись к каждому временному ряду.

Для каждого временного ряда из исследуемой выборки изначально производилась процедура удаления выбросов. Далее происходило деление ряда в рамках алгоритма кросс-валидации. После шли 4 шага предобработки, на которые был направлен наш первый эксперимент:

1. удаление выбросов;
2. удаление сезонности;
3. базовое преобразование;
4. нормализация.

Под базовым преобразованием мы условно понимали методы логарифмирования, Бокса-Кокса или взятие первой разности, которые перебирались по ходу эксперимента. Также подразумевалось, что на описанных шагах преобразование может и не производиться.

Этапы удаления выбросов, сезонности, базовое преобразование и нормализация являлись опциональными и перебирались в циклах в рамках алгоритма, представленного ни рисунке 2.3.

```
for OutlayerTR in [HampelFilter, None] do
    for SeasonalTR in [Deseasonalizer, None] do
        for BaseTR in [Differencer, Log, BoxCox, None] do
            for NormTR in [MinMaxScaler, None] do
                for TS in Data do
                    TS ← FillMissValues(TS)
                    for train, val, test in CV.split(TS) do
                        trainTr ← OutlayerTR.FitTransform(train)
                        for TR in [SeasonalTR, BaseTR, NormTR] do
                            trainTr ← TR.FitTransform(trainTr)
                            valTr ← TR.Transform(val)
                        end for

                        model ← RNN(InitParams)
                        model.fit(FitParams, trainTr, valTr)

                        valPred ← model.predict(trainTr)
                        testPred ← model.predict(trainTr, valTr)
                        for TR in [NormTR, BaseTR, SeasonalTR] do
                            valPred, testPred ← TR.InverseTransform(valPred, testPred)
                        end for

                        CalculateMetrics(val, valPred)
                        CalculateMetrics(test, testPred)
                    end for
                end for
            end for
        end for
    end for
end for
```

Рисунок 2.3 – **Алгоритм первого эксперимента**

Пример влияния на исходный ряд шагов предобработки данных для комбинации методов HampelFilter, Deseasonalizer, BoxCoxTransformer и MinMaxScaler представлен в приложении Д.

Итого в рамках эксперимента мы получили оценку точности для 32 комбинаций методов предобработки временных рядов. В следующем подразделе мы рассмотрим ее подробнее.

2.2.2. Результаты эксперимента

В результате проведения эксперимента мы получили метрики точности прогноза, усредненные по блокам кросс-валидации и исследуемым временным рядам (см. Приложение Е).

Первые 10 лучших конфигураций предобработки на валидационном множестве представлены в таблице 2.2.

Таблица 2.2

Лучшие 10 конфигураций на валидационной выборке

Удаление выбросов	Удаление сезонности	Базовое преобразование	Нормализация	SMAPE	MAPE
Нет	Нет	Differencer	MinMaxScaler	1.74	1.73
Нет	Deseasonalizer	Differencer	MinMaxScaler	1.76	1.75
HampelFilter	Нет	Differencer	MinMaxScaler	1.83	1.82
Нет	Нет	BoxCoxTransformer	MinMaxScaler	1.90	1.90
Нет	Нет	Нет	MinMaxScaler	1.96	1.95
HampelFilter	Deseasonalizer	Differencer	MinMaxScaler	2.09	2.08
HampelFilter	Нет	Нет	MinMaxScaler	2.11	2.10
HampelFilter	Нет	LogTransformer	MinMaxScaler	2.14	2.12
Нет	Нет	Differencer	Нет	2.12	2.13
Нет	Deseasonalizer	Нет	MinMaxScaler	2.16	2.15
Auto ARIMA	Auto ARIMA	Auto ARIMA	Auto ARIMA	2.64	2.65

Лучше всего себя показала комбинация методов взятия первой разности и нормализации. При этом, конфигурации, в которых данные преобразования использовались отдельно друг от друга, также попали в топ 10.

Основные выводы по эксперименту мы формулировали по значениям метрик на тестовом множестве, которые представлены в таблице 2.3.

Таблица 2.3

Лучшие 10 конфигураций на тестовой выборке

Удаление выбросов	Удаление сезонности	Базовое преобразование	Нормализация	SMAPE	MAPE
Нет	Нет	Differencer	Нет	1.86	1.87
HampelFilter	Нет	Differencer	Нет	1.9	1.91
HampelFilter	Нет	Differencer	MinMaxScaler	2.17	2.14
Нет	Нет	Differencer	MinMaxScaler	2.29	2.25
HampelFilter	Deseasonalizer	Differencer	Нет	2.48	2.49
Нет	Deseasonalizer	Differencer	Нет	2.50	2.50
HampelFilter	Deseasonalizer	Differencer	MinMaxScaler	2.69	2.67
Нет	Deseasonalizer	Differencer	MinMaxScaler	2.74	2.70
Нет	Deseasonalizer	Нет	MinMaxScaler	2.77	2.75
Нет	Нет	Нет	MinMaxScaler	2.79	2.77
Auto ARIMA	Auto ARIMA	Auto ARIMA	Auto ARIMA	4.73	4.77

Здесь также прослеживается доминирование методов взятия первой разности и нормализации. Однако первые два места занимают конфигурации без нормализации. Это может свидетельствовать о том, что если приросты временного ряда невелики, и их распределение незначительно попадает в зоны насыщения сигмоидальной функции и гиперболического тангенса, то можно не использовать нормализацию, как дополнительный шаг предобработки временного ряда.

При детальном рассмотрении графиков прогнозов, сделанных по проработанным с помощью только взятия первой разности или только нормализации (см. Рисунок 2.4), можно отметить, что прогноз по первой методике более гладкий, чем по второй. Это можно дополнительно учитывать, исходя из целей задачи прогнозирования.

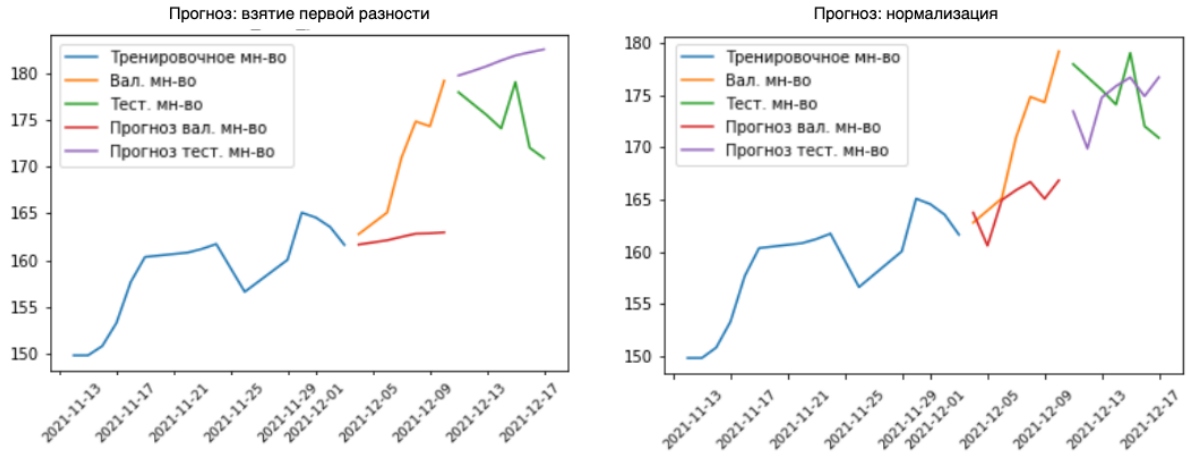


Рисунок 2.4 – **Прогнозы методом первой разности и нормализации**

Далее рассмотрим средние метрики на тестовой выборке, сгруппированные по признаку применяемого преобразования.

Таблица 2.4

Средние метрики прогнозов относительно удаления выбросов

Удаление выбросов	SMAPE	MAPE
HampelFilter	4.24	4.11
Нет	4.20	4.06

В по таблице Таблица 2.4 можно сделать вывод, что удаление выбросов не оказывает серьезного влияния на качество моделей. Скорее всего этот эффект слишком мелкий, чтобы уловить его конфигурации нашего эксперимента.

Таблица 2.5

Средние метрики прогнозов относительно удаления сезонности

Удаление сезонности	SMAPE	MAPE
Deseasonalizer	4.32	4.20
Нет	4.12	3.98

Удаление годовой мультипликативной сезонности не только не позволило улучшить качество прогноза, а даже ухудшило его (см. Таблица 2.5). Мы связываем это с тем, что котировки акций не обладают выраженной сезонной составляющей, поэтому ошибки метода ее удаления оказывают большее влияние, чем ошибки RNN, которая обучалась на временных рядах без удаления сезонности. В связи с этим можно сделать вывод, что для исследуемых

временных рядов модель рекуррентной нейронной сети справляется с выделением сезонности лучше, чем специальный метод, который мы использовали.

Таблица 2.6

Средние метрики прогнозов относительно базового преобразования

Базовое преобразование	SMAPE	MAPE
BoxCoxTransformer	4.55	4.46
Differencer	2.33	2.32
LogTransformer	3.80	3.74
Нет	6.20	5.84

По метрикам, сгруппированным относительно базового преобразования, явно прослеживается, что каждый из трех методов увеличивает качество прогноза (см. Таблица 2.6). Лучше всего с этим справляется взятие первой разности. Мы связываем это с тем, что без этого преобразования на модель RNN слишком сильно влияет возрастающий тренд, и из-за этого она придает меньше значения изменениям котировок.

На втором месте в качестве базового преобразования показал себя метод логарифмирования. Скорее всего он оказался лучше преобразования Бокса-Кокса из-за ошибки в подборе параметра λ, о которой высказывал свои опасения Bandara в своей работе[49].

Таблица 2.7

Средние метрики прогнозов относительно нормализации

Нормализация	SMAPE	MAPE
MinMaxScaler	2.95	2.92
Нет	5.49	5.25

По таблице 2.7 можно сделать вывод, что нормализация ожидаемо помогает увеличить качество прогноза моделей рекуррентных нейронных сетей. Как уже было сказано в разделе 1.3.5, это происходит из-за того, что функции активации в блоках LSTM имеют области насыщения, в которые ненормализованные данные могут попадать.

2.3. Сравнительный анализ стратегий прогнозирования RNN

В данном разделе мы описываем эксперимент с оценкой влияния стратегии прогнозирования на качество прогноза экономических временных рядов с помощью модели рекуррентной нейронной сети.

2.3.1. Методология эксперимента

Архитектура второго эксперимента схожа с архитектурой первого, но в нем итеративно перебирались не методы предобработки, а стратегии прогнозирования (см. Рисунок 2.5).

[49] K. Bandara, C. Bergmeir, S. Smyl, Forecasting across time series databases using recurrent neural networks on groups of similar series: a clustering approach, Expert Syst. Appl. (ISSN: 0957-4174) 140 (2020) 112896.

```
for Strategy in [Direct, Recursive, MiMo, DirMo, DirRec] do
    for TS in Data do
        TS ← FillMissValues(TS)
        for train, val, test in CV.split(TS) do
            trainTr ← HampelFilter.FitTransform(train)
            for TR in [Differencer, MinMaxScaler] do
                trainTr ← TR.FitTransform(trainTr)
                valTr ← TR.Transform(val)
            end for

            model ← RNN(InitParams, Strategy)
            model.fit(FitParams, trainTr, valTr)

            valPred ← model.predict(trainTr)
            testPred ← model.predict(trainTr, valTr)
            for TR in [MinMaxScaler, Differencer] do
                valPred, testPred ← TR.InverseTransform(valPred, testPred)
            end for

            CalculateMetrics(val, valPred)
            CalculateMetrics(test, testPred)
        end for
    end for
end for
```

Рисунок 2.5 - **Алгоритм второго эксперимента**

Тип стратегии прогнозирования подается в качестве параметра в момент инициализации модели RNN. Для модели DirRec в качестве параметра s использовалось значение 3. С конфигурацией второго эксперимента можно ознакомиться в приложении В.

В качестве методов предобработки мы выбрали последовательно: удаление выбросов, взятие первой разности и нормализацию. Хотя метод удаления выбросов в первом эксперименте не показал значительного увеличения качества прогноза, мы все равно решили его использовать, чтобы облегчить процесс обучения RNN. Мы не использовали метод удаления сезонной составляющей из временных рядов из-за того, что в первом эксперименте он оказался неэффективным для выбранных временных рядов.

2.3.2. Результаты эксперимента

После проведения эксперимента мы получили значения метрик на валидационном множестве, которые представлены в таблице 2.8.

Таблица 2.8

Метрики стратегий на валидационной выборке

Стратегия	SMAPE	MAPE	SMAPE Ранг	MAPE Ранг
DirMo	1.82	1.81	1.0	1.0
Direct	1.84	1.83	2.0	2.0
DirRec	1.93	1.91	3.0	3.0
MiMo	1.96	1.95	4.0	4.0
Recursive	1.96	1.95	4.0	4.0
Auto ARIMA	2.64	2.65	5.0	5.0

Хотя каждая из стратегий прогнозирования RNN превзошла по качеству прогноз модели ARIMA, между собой значительного отличия по метрике MAPE не наблюдается.

Метрики, полученные на основе прогнозов на тестовом множестве, представлены в таблице 2.9.

Таблица 2.9

Метрики стратегий на тестовой выборке

Стратегия	SMAPE	MAPE	SMAPE Ранг	MAPE Ранг
MiMo	2.11	2.09	1.0	1.0
DirMo	2.24	2.21	2.0	2.0
Direct	2.34	2.31	3.0	3.0
Recursive	2.37	2.33	4.0	4.0
DirRec	2.41	2.37	5.0	5.0
Auto ARIMA	4.73	4.77	6.0	6.0

На тестовом множестве стратегии, основанные подходе множественного выхода, показали наилучшую точность. Это согласуется с результатами исследования[50], которое мы подробно рассмотрели в разделе 1.4 текущей работы. Мы можем объяснить это тем, что при прогнозировании подобным подходом не накапливаются ошибки, что свойственно рекурсивным стратегиям. Можно сделать вывод, что подобная тенденция сохраняется применимо к экономическим временным рядам.

2.4. Сравнительный анализ использования моделей RNN, сочетающих информацию из нескольких временных рядов.

В данном разделе мы опишем эксперимент, в котором мы оценивали различные возможности учитывать в одной модели рекуррентной нейронной сети информацию из нескольких временных рядов.

2.4.1. Методология эксперимента

Первым подходом учесть дополнительную информацию является прогнозирование многомерных временных рядов. В этом случае при прогнозировании используются не только прошлые значения целевого временного ряда, но и прошлые значения рядов, которые могли бы на него повлиять. В нашем эксперименте в качестве объясняющих факторов мы выбрали показатели, показанные в таблице 2.10, часть из которых, согласно статье[51], исследователи часто используют в задачах прогнозирования экономических временных рядов.

Второй подход представляет из себя использование глобальных моделей, которые последовательно обучаются на наборе временных рядов, как в одномерном случае.

---

[50] S. Ben Taieb, G. Bontempi, A.F. Atiya, A. Sorjamaa, A review and comparison of strategies for multi-step ahead time series forecasting based on the NN5 forecasting competition, Expert Syst. Appl. 39 (8) (2012) 7067–7083.
[51] Omer Berat Sezer, Mehmet Ugur Gudelek, Ahmet Murat Ozbayoglu, Financial time series forecasting with deep learning: A systematic literature review: 2005–2019, Applied Soft Computing, Volume 90, 2020, 106181

Таблица 2.10

Список используемых факторов

Тикер	Название фактора
USDRUB=X	Курс доллара к рублю
EURRUB=X	Курс евро к рублю
BZ=F	Цена на нефть Brent
GC=F	Цена на золото
^GSPC	Цена индекса S&P 500
^IXIC	Цена индекса NASDAQ
^DJI	Цена индекса Dow Jones

С конфигурацией третьего эксперимента можно ознакомиться в приложении Г.

Общая схема предобработки данных и процесса обучения моделей в третьем эксперименте представлена на рисунке 2.6.

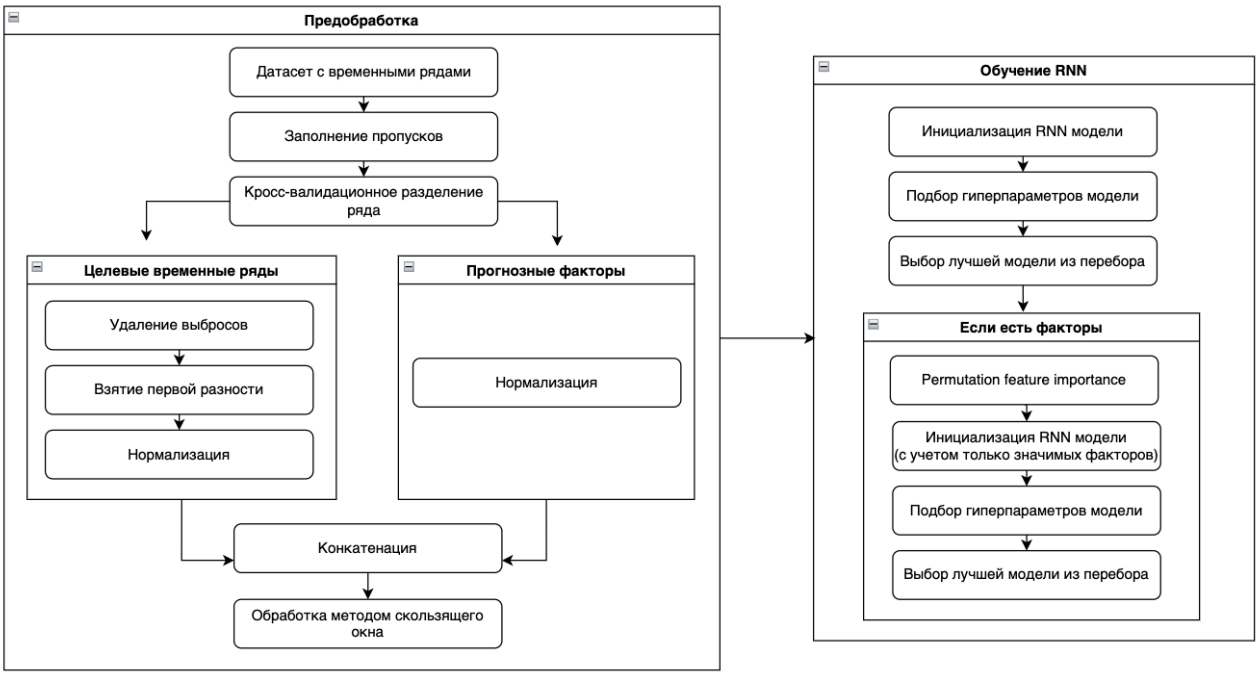


Рисунок 2.6 – **Схема предобработки данных и обучения моделей в третьем эксперименте**

Для целевых временных рядов мы использовали методы предобработки, аналогичные тем, которые использовались во втором эксперименте: удаление выбросов, взятие первой разности и нормализацию.

Если при предобработке целевого ряда нам было необходимо очистить его от всех постоянных зависимостей, например сезонности, чтобы упростить задачу для рекуррентной нейронной сети на этапе обучения, то для факторов наоборот необходимо максимально оставить всю содержащуюся в них информацию, которая может помочь модели объяснить изменение целевого ряда. Поэтому в качестве предобработки временных рядов, которые выступали в качестве факторов, мы использовали только нормализацию.

Для снижения влияния незначимых факторов мы воспользовались методом Permutation feature importance, суть которого заключается в том, чтобы после обучения модели со всеми доступными факторами итеративно перемешивать каждый из них и замерять качество прогноза с учетом изменённого входа. Эта процедура разрывает связь между фактором и целевым показателем, тем самым снижение качества прогноза модели указывает на то, насколько сильно модель зависит от перемешиваемого фактора. Далее модель инициализируется и обучается заново с учетом только значимых факторов.

Преимущество этого метода заключается в том, что он не зависит от модели и может быть рассчитан много раз с различными перестановками.

В случае нейронных сетей, которые содержат огромное количество параметров, данный метод зарекомендовал себя в качестве одного из популярных в практической среде.

2.4.2. Результаты эксперимента

В результате проведения эксперимента мы получили распределение метрик качества прогноза рассматриваемых моделей на валидационном (см Таблица 2.11) и тестовом множествах (см Таблица 2.12).

Таблица 2.11

Метрики исследуемых моделей на валидационном множестве

Вид модели	SMAPE	MAPE	SMAPE Ранг	MAPE Ранг
Локальная с факторами	1.92	1.92	1.0	1.0
Локальная без факторов	1.93	1.92	2.0	1.0
Auto ARIMA	2.64	2.65	3.0	2.0
Глобальная без факторов	7.35	6.91	4.0	3.0
Глобальная с факторами	7.50	7.04	5.0	4.0

Для валидационного множества модели локальных рекуррентных нейронных сетей сошлись примерно до одного значения метрик. Глобальные модели показали значительную степень переобучения в случае как одномерного, так и многомерного прогнозирования.

Таблица 2.12

Метрики исследуемых моделей на тестовом множестве

Вид модели	SMAPE	MAPE	SMAPE Ранг	MAPE Ранг
Локальная с факторами	2.02	2.00	1.0	1.0
Локальная без факторов	2.10	2.07	2.0	2.0
Auto ARIMA	4.73	4.77	3.0	3.0
Глобальная без факторов	7.66	7.21	4.0	4.0
Глобальная с факторами	7.94	7.46	5.0	5.0

В случае с локальными моделями использование подхода многомерного прогнозирования действительно позволяет увеличить точность прогнозов. Однако стоит отметить, что качество прогноза во многом будет зависеть от выбора объясняющих временных рядов.

Использование глобальных моделей в нашем эксперименте не показало приемлемых результатов. Их прогнозы оказались почти в два раза менее точными по сравнению с базовой моделью ARIMA. Мы связываем это с недостаточным количеством временных рядов в обучающей выборке. Также необходимо отметить, что исследуемые нами временные ряды сложно назвать схожими по своей структуре: они номинированы в разных валютах, имеют различную форму тренда и сезонные компоненты. Мы делаем вывод, что применение глобальных моделей требует более детального внимания к составу и методам предобработки временных рядов. Подобному виду моделей довольно легко переобучиться на шуме, присущему каждому отдельному временному ряду из обучающей выборки.

Выводы

Во второй главе мы решили задачи экспериментального блока, поставленные во введении данной работы.

Был проведен первый эксперимент, в котором оценивалось влияние комбинаций методов предобработки финансовых временных рядов на точность прогнозирования моделей рекуррентных нейронных сетей. На основе результатов эксперимента мы рекомендуем использовать методы взятия первой разности и нормализации временных рядов. При этом для финансовых временных рядов не стоит использовать методы удаления сезонности, а нужно давать возможность моделям рекуррентным нейронным сетям самостоятельно выделять сезонную составляющую.

По итогам второго эксперимента, который был направлен на анализ влияния стратегий прогнозирования применительно к моделям RNN для финансовых временных рядов, следует вывод, что стоит использовать стратегию MiMo, которая позволила добиться высокой точности прогноза на тестовой выборке при сравнительно небольших временных затратах.

Третий эксперимент был посвящен анализу подходов к обучению одной модели рекуррентной нейронной сети на основе информации из нескольких временных рядов. Мы рекомендуем использовать многомерные модели, включающие в себя объясняющие факторы в комбинации с лагами целевого временного ряда. При верном выборе объясняющих временных рядов использование подобных моделей действительно позволяет увеличить качество прогноза.

ГЛАВА 3. ПРИМЕНЕНИЕ RNN МОДЕЛЕЙ В РАМКАХ ПРОЕКТА КОМПАНИИ «ГАЗПРОМ НЕФТЬ»

Рассмотренные в рамках данной работы методы мы применили на реальной задаче в рамках проекта компании «Газпром нефть». В текущем разделе мы представим его ограничения и цели, а также приводим результаты внедрения моделей рекуррентных нейронных сетей.

3.1. Описание поставленной задачи

ПАО «Газпром нефть» — вертикально-интегрированная нефтяная компания, работающая по принципу акционерного общества холдингового типа, организационно объединяя предприятия по всей технологической цепочке производства нефтепродуктов.

Ее основные виды деятельности охватывают весь процесс от геологоразведки, разработки нефтяных месторождений и добычи нефти до её переработки в продукты конечного пользования и их оптовой и розничной реализации потребителю. Наше исследование проходило в рамках дочерней компании ООО «Газпромнефть-Региональные продажи», деятельность которой сфокусирована на реализации готовых нефтепродуктов.

Формат продаж нефтепродуктов по описанию Российского топливного союза[52] подразделяется по способу и объемам их поставки на:

- крупный опт - от нефтеперерабатывающего завода по железной дороге до покупателя;
- мелкий опт - от нефтебазы автомобильным транспортом до покупателя.
- розничную торговлю – продажа нефтепродуктов на АЗС.

Основными покупателями в крупном опте являются вертикально-интегрированные нефтяные компании, которые покупают нефтепродукты напрямую с нефтеперерабатывающих заводов большими объемами и далее перевозят их на свои нефтебазы.

Мелкий опт характеризуется небольшими объемами поставок. Продавцами в данном сегменте являются нефтяные компании, а основными покупателями - независимые АЗС, промышленные предприятия, сельхозпредприятия, транспортные предприятия-перевозчики, строительные компании и т.д.

В каждом из рассмотренных форматов продаж образуются свои локальные рынки, на которых устанавливаются цены на тот или иной нефтепродукт. Зачастую эти цены неоднородны и зависят от условий множества факторов, поэтому для их оценки используются специальные индексы.

---

[52] Топливо - мелкий опт крупным планом [Электронный ресурс]: Российский топливный союз, 2017 - Режим доступа: www.rfu.ru/1553-toplivo-melkij-opt-krupnym-planom.html, свободный (дата обращения: 11.04.2022)

Для планирования деятельности в среднесрочном периоде в компании «Газпром нефть» аналитики решают задачу прогнозирования финансового результата. В рамках рассматриваемого проекта мы ограничивались сегментом мелкого опта. Важнейшей составляющей, которая может повлиять на прибыльность компании, является себестоимость нефтепродуктов, представляющая из себя цены на рынке крупного опта, взятые на несколько периодов назад от расчетной даты. По каждому нефтепродукту и по каждому нефтеперерабатывающему заводу их агрегируют в индекс биржевых котировок (ИБК), который рассчитывается по методике, принятой в компании.

Для аналитиков необходим прогноз данного индекса на 6 декад (периоды по 10 дней) вперед, относительно текущей даты. В текущей методике ИБК экстраполировался с помощью прогнозного алгоритма, который был написан на языке R и состоял из ансамбля следующих моделей:

- Prophet – усовершенствование аддитивной регрессионной модели, разработанное в 2017 году компанией Facebook[53].
- Sarima – разновидность модели ARIMA, в которой учитывается сезонность.

Нашей задачей стало улучшение данного подхода путем внедрения моделей рекуррентных нейронных сетей с целью увеличения точности прогноза.

Для работы мы получали из базы данных индекс биржевых котировок в крупном опте по дням, заполняли пропуски линейной интерполяцией и агрегировали его до декад. Исходные временные ряды имели данные начиная с января 2013 года. После агрегации в каждом целевом временном ряду содержалось по 326 наблюдений.

Всего было необходимо сделать прогноз для 9 временных рядов, которые разбивались по:

- 3 нефтеперерабатывающим заводам;
- 3 нефтепродуктам;

В дальнейшем планируется расширение моделей на большее количество временных рядов.

Данные, которые использовались в исследовании, являются коммерческой тайной компании «Газпром нефть», поэтому для опубликования данной работы их пришлось обезличить и изменить.

В дальнейшем мы будем использовать условные обозначения:

- Для нефтеперерабатывающих заводов: НПЗ 1, НПЗ 2 и т.д.
- Для нефтепродуктов: Нефтепродукт 1, Нефтепродукт 2 и т.д.

---

[53] Prophet: forecasting at scale [Электронный ресурс]: Sean J. Taylor, Ben Letham, Facebook research – 2017. - Режим доступа: www.doi.org/10.7287/peerj.preprints.3190v2, свободный (дата обращения: 11.04.2022)

Для сопоставления результатов моделирования в качестве валидационного периода мы выбрали 6 декад, относящихся к сентябрю и октябрю 2021 года. В качестве тестового периода были выбраны декады, относящиеся к ноябрю и декабрю 2021 года.

3.2. Внедрение модели RNN в прогнозный алгоритм

В качестве тестируемых моделей выступали многомерные и одномерные локальные рекуррентные нейронные сети со стратегией прогнозирования MiMo. Архитектура моделей была идентична той, которая использовалась в экспериментах в рамках данной работы (см. Приложение А). Конфигурация прогнозной системы представлена в приложении Ж.

Для многомерных моделей мы отобрали факторы, которые потенциально влияют на курс продажи нефтепродуктов (см. Таблица 3.1).

Таблица 3.1

Список используемых факторов

Тикер	Название фактора
USDRUB=X	Курс доллара к рублю
EURRUB=X	Курс евро к рублю
BZ=F	Цена на нефть Brent

Целевые временные ряды ИБК выгружались из внутренней базы данных компании. Прогнозные факторы мы получали через открытое API сервиса yahoo finance.

Временные ряды проходили несколько этапов предобработки, схема которых представлена на рисунке 3.1.

На первом этапе во временных рядах заполнялись пропуски методом линейной интерполяции. Данный шаг был необходим, потому что нейронные сети не умеют обрабатывать пропущенные значения. На втором этапе наблюдения в каждом временном ряду мы агрегировали до декад, используя среднее значение по дням внутри каждой декады. На следующем шаге временные ряды разделялись на тренировочную (train), валидационную (val) и тестовую (test) выборки. Следующие шаги зависели от типа временного ряда.

Для целевых временных рядов последовательно использовались: метод удаления выбросов Hampel Filter, взятие первой разности и нормализация. Для временных рядов, которые выступали в качестве прогнозных факторов, мы применяли только метод нормализации, как и в третьем эксперименте.

Последним шагом временные ряды нарезались методом скользящего окна. На вход каждой модели RNN подавался вектор длиной 36, который захватывал полный год декадных наблюдений. Мы выбрали такую длину из-за способности рекуррентных нейронных сетей

лучше выделять сезонную составляющую в случае, когда входной вектор охватывает полный цикл сезонности. Так как после агрегации до декад структура сезонности нам была неясна, мы решили взять полный год наблюдений в качестве входного вектора для прогноза. Такой подход позволит RNN захватить как годовую, так и месячную сезонность.

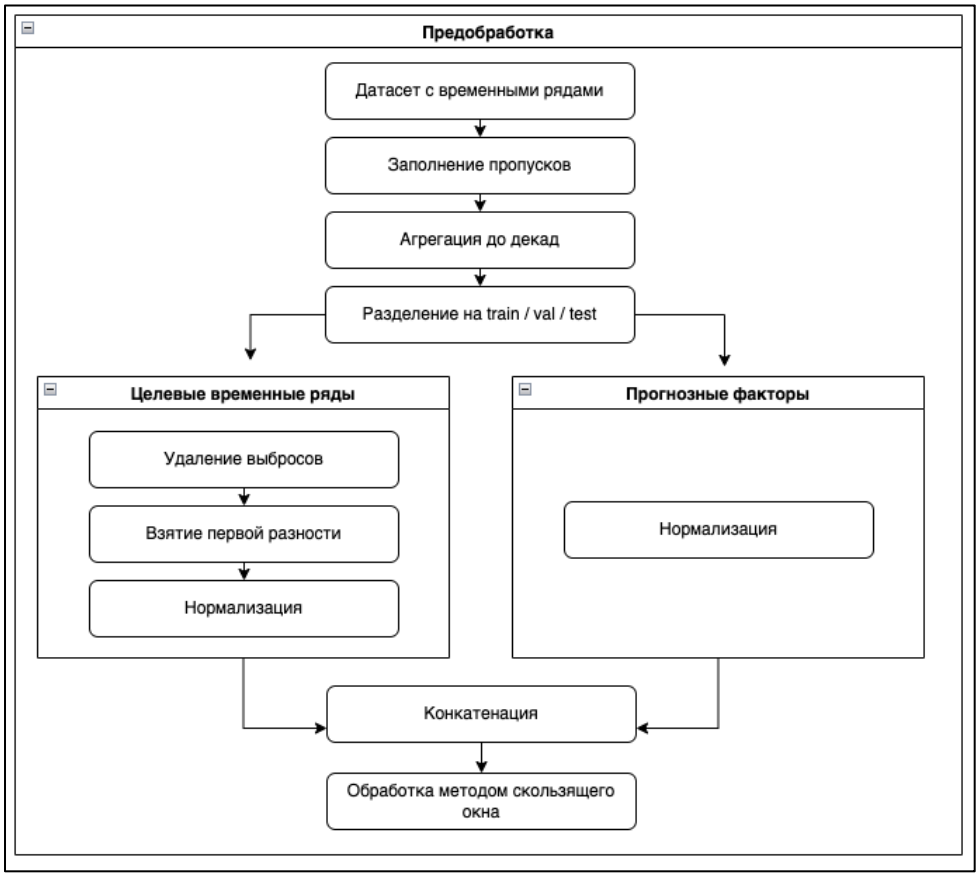


Рисунок 3.1 – **Этапы предобработки временных рядов**

Также по результатам первого эксперимента мы выяснили, что модели RNN в случае прогнозирования экономических временных рядов без явной сезонности выделяют сезонную составляющую лучше, чем специальные методы ее удаления на этапе предобработки временного ряда. В связи с этим, мы решили не использовать методы удаления сезонной составляющей.

Полный алгоритм реализованного решения представлен в приложении 3.

3.3. Результаты внедрения модели RNN

В данном разделе мы рассматриваем результаты моделирования, а также сравниваем их с действующей в проекте прогнозной системой.

В качестве метрик качества прогноза в проекте были выбраны:

- MAE - средняя абсолютная ошибка прогноза (формула (13)).
- MAPE – средняя абсолютная процентная ошибка прогноза (формула (14)).

Выбранные метрики являются хорошо интерпретируемыми для бизнеса, и поэтому их часто используют в реальных экономических задачах. Первая показывает на сколько рублей в

среднем ошибается модель, выполняя прогноз, а вторая какой процент занимает эта ошибка от реального значения целевого показателя.

Таблица 3.2

Метрики моделей на тестовом периоде

Метрика	Текущая методика	RNN без факторов	RNN с факторами
Mean Absolute Error	2450,14	1779,75	2199,52
Mean absolute percentage error	4,84	3,52	4,35

В таблице 3.2 представлены средние метрики по всем исследуемым временным рядам в разрезе трех прогнозных методик. Все метрики были рассчитаны на тестовом периоде, который не участвовал в обучении моделей и охватывал 2 месяца наблюдений (6 декад).

Можно сделать вывод, что модели RNN в обеих конфигурациях оказались точнее текущей методики компании.

При этом нейронные сети, обучаемые с использованием дополнительной информации из прогнозных факторов, показали качество прогноза хуже, чем RNN, обучение которой происходило только на основе лагов целевого показателя. Это может быть связано и с ошибкой, которая появляется при агрегировании факторов до декад, и с недостаточным количеством наблюдений для обучения.

Таблица 3.3

Распределение лучших моделей на тестовом периоде

НПЗ	Нефтепродукт	Лучшая модель
НПЗ 1	Нефтепродукт 1	RNN без факторов
НПЗ 1	Нефтепродукт 2	RNN без факторов
НПЗ 1	Нефтепродукт 3	RNN без факторов
НПЗ 2	Нефтепродукт 1	RNN без факторов
НПЗ 2	Нефтепродукт 2	RNN без факторов
НПЗ 2	Нефтепродукт 3	RNN без факторов
НПЗ 3	Нефтепродукт 1	Текущая методика
НПЗ 3	Нефтепродукт 2	RNN без факторов
НПЗ 3	Нефтепродукт 3	RNN без факторов

В таблице 3.3 представлено распределение лучших моделей по метрике MAE для каждого исследуемого временного ряда. Можно сделать вывод, что в большинстве случаев внедрение модели рекуррентной нейронной сети без использования дополнительных факторов приводит к увеличению точности прогноза.

Однако стоит отметить, что в одном временном ряду модели текущей методики показали более высокую точность. Поэтому мы рекомендовали рассмотреть не только замену текущей методики на RNN, но и возможность интеграции моделей рекуррентных нейронных сетей в текущий прогнозный алгоритм. Данный подход позволит дополнительно увеличить точность всей прогнозной системы.

Также было замечено, что ИБК в рамках одного и того же вида нефтепродукта изменяется схожим образом. По мере расширения базы временных рядов возможно обучение глобальных RNN для каждого вида нефтепродукта.

Выводы

На основании проведенного анализа, а также разработанного фреймворка мы реализовали прогнозную систему и применили ее в рамках реального проекта компании «Газпром нефть».

Для планирования стратегии деятельности компании в среднесрочном периоде нужно иметь оценку финансового результата в каждом из сегментов. В связи с этим было необходимо спрогнозировать 9 временных рядов, которые представляли из себя агрегированные в индекс биржевых котировок цены нефтепродуктов в сегменте крупного опта, которые в свою очередь представляют из себя себестоимость в сегменте мелкого опта.

Требовалось построить прогноз на 2 месяца подекадно (периоды по 10 дней), при этом превзойдя текущую методику, которая представляла из себя прогнозную систему из ансамбля моделей Prophet и SARIMA.

Созданная нами прогнозная система позволила увеличить точность текущей методики на 1.3%, и снизить среднюю абсолютную процентную ошибку с 4.84 до 3.52%, что является значимым результатом и удовлетворило заказчика.

ЗАКЛЮЧЕНИЕ

Взяв в качестве объекта исследования - финансовые временные ряды и в качестве предмета исследования - прогнозные модели рекуррентных нейронных сетей, применяемые для финансовых временных рядов, мы реализовали поставленную цель - разработали прогнозную систему на языке Python и применили ее в рамках реальной задачи бизнес-планирования компании Газпром нефть. Созданная нами прогнозная система позволила увеличить точность на 1.3%, и снизить среднюю абсолютную процентную ошибку с 4.84 до 3.52%, что является значимым результатом.

В процессе работы мы выполнили поставленные во введении задачи. В первой главе, исходя из задач теоретического блока, были рассмотрены основные этапы развития области прогнозирования временных рядов с помощью рекуррентных нейронных сетей и были описаны текущие исследовательские вопросы. Из проведенного анализа можно сделать вывод, что на данный момент область нашего исследования находится на своем пике. Благодаря соревнованию M4, проводимому в 2018 году, стал явным потенциал использования нейронных сетей в задачах прогнозирования временных рядов.

Также в первой главе мы рассмотрели основные тенденции применения RNN моделей для прогнозирования экономических временных рядов. Исследователи рекомендуют использовать блоки LSTM в качестве основы для архитектуры моделей RNN, производить нормализацию временных рядов перед обучением моделей, а также использовать дополнительную экономическую информацию для улучшения точности прогноза. Отмечается, что основным языком программирования, используемым для решения задачи прогнозирования финансовых временных рядов является язык Python.

После анализа существующих решений нами был разработан фреймворк для языка Python, который позволяет исследователям автоматически выполнять прогнозы с помощью моделей RNN, используя различные стратегии прогнозирования. Также наша реализация поддерживает обучение одной нейронной сети на основании информации из нескольких временных рядов: как в случае глобальных моделей, так и в случае использования объясняющих рядов-факторов. Важным достоинством нашего фреймворка является встроенный в процесс обучения автоматический подбор гиперпараметров нейронной сети по заранее заданной сетке допустимых значений. Код проекта является открытым, каждый желающий может воспроизвести результаты наших экспериментов[54].

В рамках второй главы мы решали задачи из экспериментального блока. Было проведено три эксперимента, направленные на оценку влияния комбинаций методов предобработки

[54] Программный код библиотеки TS_RNN [Электронный ресурс] - Режим доступа: www.github.com/LevPerla/Time_Series_Prediction_RNN, (дата обращения: 11.04.2022)

входных данных, стратегий прогнозирования и методов учета информации из нескольких временных рядов на точность прогнозирования рекуррентной нейронной сети.

Мы выяснили, что для финансовых временных рядов, которым часто свойственен возрастающий тренд и отсутствие явной сезонной составляющей, лучше всего использовать такие методы предобработки, как взятие первой разности и нормализацию. Методы удаления сезонности в нашем случае показали себя не так хорошо, поэтому мы рекомендуем давать возможность рекуррентной нейронной сети самостоятельно выделять сезонную составляющую на этапе обучения модели.

В качестве наилучшей стратегии прогнозирования мы рекомендовали использовать multiout подход, который является оптимальным с точки зрения сложности модели и получаемой точности прогнозов.

Использование объясняющих факторов в комбинации с лагами целевого временного ряда действительно позволяет увеличить качество прогноза, однако требует более детального внимания к конфигурации модели рекуррентной нейронной сети.

На основании проведенного анализа, а также разработанного фреймворка мы реализовали прогнозную систему, которую мы применили в рамках реального проекта компании «Газпром нефть». Нужно отметить, что применение моделей рекуррентных нейронных сетей действительно имеет экономическую ценность, так как увеличение качества прогноза напрямую влияет на эффективность процесса планирования. Например, в рамках рассмотренного проекта возможность точнее прогнозировать себестоимость нефтепродуктов повлияет на своевременность принятия управленческих решений. Также при использовании прогнозов в процессе оптимизации поставок возможно снизить издержки, связанные с доставкой и хранением нефтепродуктов, продаваемых по каналу мелкого опта.

Дальнейшим направлением работы будет проведение экспериментов по применению глобальных моделей в рамках каждого из видов нефтепродуктов. Основная гипотеза заключается в том, что благодаря данному подходу возможно решить проблему малого количества обучающих данных для моделей. Также было выяснено, что изменения отпускных цен крупного опта в разрезе каждого нефтепродукта имеет схожий вид, что дополнительно создает стимул к использованию данного похода.

Сформулированные в работе рекомендации по прогнозированию временных рядов с помощью моделей рекуррентных нейронных сетей и разработанная прогнозная система могут быть использованы во многих сферах, в которых необходим прогноз финансовых показателей.

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

Книги

1. Hyndman, R.J. Forecasting: principles and practice [Электронный ресурс] // R.J.Hyndman, G. Athanasopoulos ; OTexts: Melbourne, Australia, - 2nd edition - 2018. - Режим доступа: www.otexts.com/fpp2, свободный (дата обращения: 03.05.2022)

Статьи в журналах

2. Алжеев, А. В. Сравнительный анализ прогнозных моделей ARIMA и LSTM на примере акций российских компаний / А. В. Алжеев, Р. А. Кочкаров // Финансы: теория и практика. – 2020. – Т. 24. – № 1. – С. 14-23. – DOI 10.26794/2587-5671-2020-24-1-14-23.

3. Горшенин, А. К. Анализ конфигураций LSTM-сетей для построения среднесрочных векторных прогнозов / А. К. Горшенин, В. Ю. Кузьмин // Информатика и ее применения. – 2020. – Т. 14. – № 1. – С. 10-16. – DOI 10.14357/19922264200102.

4. Обрубов, М. О. Применение LSTM-сети в решении задачи прогнозирования многомерных временных рядов / М. О. Обрубов, С. Ю. Кириллова // Национальная Ассоциация Ученых. – 2021. – № 68-2. – С. 43-48.

5. Прогностические модели развития рынка криптовалюты (ARMA, LSTM): сравнительный анализ / А. А. Абдукаева, Л. А. Ельшин, А. М. Гильманов, В. В. Бандеров // Казанский экономический вестник. – 2019. – № 6(44). – С. 88-96.

6. Freeman, Brian S, Graham W. Taylor, Bahram Gharabaghi and J. The. “Forecasting air quality time series using deep learning.” Journal of the Air & Waste Management Association 68 (2018): 866 - 886.

7. H. Hewamalage, C. Bergmeir, K. Bandara, Recurrent neural networks for time series forecasting: current status and future directions, Int. J. Forecast. (ISSN: 0169-2070) (2020).

8. Hochreiter S., Long short-term memory / S. Hochreiter, J. Schmidhuber // Neural computation - 1997. - № 9 (8) - pp. 1735–1780

9. Hyndman, R., Khandakar, Y., 2008. Automatic time series forecasting: The forecast package for R. Journal of Statistical Software, Articles 27 (3), 1–22.

10. K. Bandara, C. Bergmeir, S. Smyl, Forecasting across time series databases using recurrent neural networks on groups of similar series: a clustering approach, Expert Syst. Appl. (ISSN: 0957-4174) 140 (2020) 112896.

11. Makridakis, S., Spiliotis, E., & Assimakopoulos, V. (2018a). Statistical and machine learning forecasting methods: concerns and ways forward. PLoS One, 13(3), 1–26

12. Makridakis, S., Spiliotis, E., Assimakopoulos, V., 2020. The M4 Competition: 100,000 time series and 61 forecasting methods. Int. J. Forecast. 36 (2020) 54–74

13. Muzaffar, Shahzad and A. Afshari. “Short-Term Load Forecasts Using LSTM Networks.” Energy Procedia 158 (2019): 2922-2927

14. Omer Berat Sezer, Mehmet Ugur Gudelek, Ahmet Murat Ozbayoglu, Financial time series forecasting with deep learning: A systematic literature review: 2005–2019, Applied Soft Computing, Volume 90, 2020, 106181

15. P. Montero-Manso, R.J. Hyndman, Principles and algorithms for forecasting groups of time series: locality and globality, Int. J. Forecast. (ISSN: 0169-2070) (2021).

16. R. Godahewa, K. Bandara, G.I. Webb, S. Smyl, C. Bergmeir, Ensembles of localised models for time series forecasting, Knowledge-Based Systems 233 (2021) 107518

17. Prophet: forecasting at scale [Электронный ресурс]: Sean J. Taylor, Ben Letham, Facebook research – 2017. - Режим доступа: www.doi.org/10.7287/peerj.preprints.3190v2, свободный (дата обращения: 11.04.2022)

18. S. Ben Taieb, G. Bontempi, A.F. Atiya, A. Sorjamaa, A review and comparison of strategies for multi-step ahead time series forecasting based on the NN5 forecasting competition, Expert Syst. Appl. 39 (8) (2012) 7067–7083.

19. S. Smyl, A hybrid method of exponential smoothing and recurrent neural networks for time series forecasting, Int. J. Forecast. 36 (1) (2020) 75–85.

20. Schafer, A. M., Zimmermann, H. G., 10–14 Sep 2006. Recurrent neural networks are universal approximators. In: Proceedings of the 16th International Conference on Artificial Neural Networks - Volume Part I. ICANN’06. Springer-Verlag, Berlin, Heidelberg, pp. 632–640.

21. Schmidhuber J., Evolino: Hybrid Neuroevolution / J. Schmidhuber, D. Wierstra, F. J. Gomez // Optimal Linear Search for Sequence Learning. Proceedings of the 19th International Joint Conference on Artificial Intelligence (IJCAI), Edinburgh - 2005. - pp. 853—858.

22. Smyl, S., Kuber, K., 19–22 Jun 2016. Data preprocessing and augmentation for multiple short time series forecasting with recurrent neural networks. In: 36th International Symposium on Forecasting.

23. Zhang, G. P., & Qi, M. (2005). Neural network forecasting for seasonal and trend time series. European Journal of Operational Research, 160(2), 501–514. ISSN: 0377- 2217.

24. Zhang, G., Eddy Patuwo, B., Hu, M. Y., 1998. Forecasting with artificial neural networks: The state of the art. Int. J. Forecast. 14, 35–62

Интернет-ресурсы и электронные базы данных

25. Международная научная база данных Web of Science [Электронный ресурс] Режим доступа: https://www.webofscience.com, (дата обращения: 11.04.2022)

26. Программный код библиотеки TS_RNN [Электронный ресурс] - Режим доступа: www.github.com/LevPerla/Time_Series_Prediction_RNN, (дата обращения: 11.04.2022)

27. Документация фреймворка GluonTS [Электронный ресурс] - Режим доступа: www.github.com/awslabs/gluon-ts/, (дата обращения: 11.04.2022)

28. Документация фреймворка Darts [Электронный ресурс] - Режим доступа: www.github.com/unit8co/darts, (дата обращения: 11.04.2022)

29. Документация фреймворка sktime [Электронный ресурс] - Режим доступа: www.github.com/alan-turing-institute/sktime, (дата обращения: 11.04.2022)

30. Документация фреймворка keras-tuner [Электронный ресурс] - Режим доступа: www.github.com/keras-team/keras-tuner, (дата обращения: 11.04.2022)

31. Информационная база финансовой информации yahoo finance [Электронный ресурс] - Режим доступа: www.finance.yahoo.com, (дата обращения: 11.04.2022)

32. Топливо - мелкий опт крупным планом [Электронный ресурс]: Российский топливный союз, 2017 - Режим доступа: www.rfu.ru/1553-toplivo-melkij-opt-krupnym-planom.html, свободный (дата обращения: 11.04.2022)

ПРИЛОЖЕНИЯ

Приложение А. Конфигурация архитектуры

```
hp = HyperParameters()
rnn_arch = {"layers": [
                ["LSTM", {"units": hp.Int(name='units',
                                  min_value=2,
                                  max_value=30,
                                  step=10,
                                  default=12),
                          "return_sequences": False,
                          "kernel_initializer": "glorot_uniform",
                          "activation": hp.Choice(name='LSTM_1_activation',
                                        values=['relu', 'tanh', 'sigmoid', "linear"],
                                        default='relu')}],
                ["Dropout", {"rate": hp.Float(name='dropout',
                                      min_value=0.0,
                                      max_value=0.5,
                                      default=0.2,
                                      step=0.05)}],
                ["Dense", {"activation": "linear"}]
            ]}
```

Приложение Б. Конфигурация первого эксперимента

```
my_callbacks = [callbacks.EarlyStopping(patience=10, monitor='val_loss')]

CONFIG = {
   "TARGET": {'TICKERS': [
                     'YNDX.ME',
                     'SBER.ME',
                     'POLY.ME',
                     'SIBN.ME',
                     'AMZN',
                     'AAPL',
                     'GOOGL',
                     'NFLX'],
            'MIN_DATE': '2012-01-01',
            'MAX_DATE': '2022-01-01'},
   "VAL_LEN": 7,
   "TEST_LEN": 7,
   "CV_FOLDS": 5,
   "MODEL": {'INIT':{
                'rnn_arch': rnn_arch,
                'tuner_hp': hp,
                "strategy": "MiMo",
                "n_lags": 30,
                "horizon": 7,
                "tuner": "BayesianOptimization",
                "max_trials": 5,
                "loss": 'mae',
                "optimizer": 'adam'},
             'FIT':{"epochs": 40, "batch_size": 14, 'callbacks': my_callbacks}},
   "OUTLAYER_TRANSFORMERS": ['HampelFilter', None],
   "BASE_TRANSFORMERS": [
                  'Differencer',
                  'LogTransformer',
                  'BoxCoxTransformer',
                  None
                 ],
   "SEASON_TRANSFORMERS": ['Deseasonalizer', None],
   "NORM_TRANSPORMERS": ["MinMaxScaler", None],
   "TRANSFORMERS_ARGS": { 'HampelFilter':{'window_length': 10},
                          'MinMaxScaler': {"feature_range": (0, 1)},
                          'Differencer': {'lags': [1], "drop_na": False},
                          'LogTransformer':{},
                          'BoxCoxTransformer': {},
                           'Deseasonalizer':{'sp': 364, 'model': 'multiplicative'},
                       }
}
```

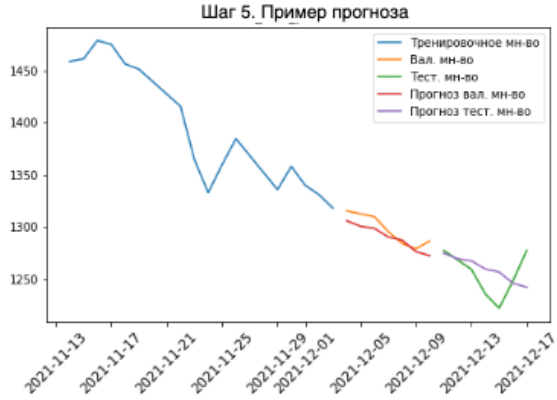
Приложение В. Конфигурация второго эксперимента

```
my_callbacks = [callbacks.EarlyStopping(patience=10, monitor='val_loss')]

CONFIG = {
   "TARGET": {'TICKERS': [
                     'YNDX.ME',
                     'SBER.ME',
                     'POLY.ME',
                     'SIBN.ME',
                     'AMZN',
                     'AAPL',
                     'GOOGL',
                     'NFLX'
                     ],
            'MIN_DATE': '2012-01-01',
            'MAX_DATE': '2022-01-01'},
   "VAL_LEN": 7,
   "TEST_LEN": 7,
   "CV_FOLDS": 5,
   "STRATEGIES": ["Recursive", "MiMo", "Direct", 'DirRec', "DirMo"],
   "MODEL": {'INIT':{
                'rnn_arch': rnn_arch,
                'tuner_hp': hp,
                "n_lags": 30,
                "horizon": 7,
                "tuner": "BayesianOptimization",
                "max_trials": 5,
                "loss": 'mae',
                "optimizer": 'adam'},
             'FIT':{"epochs": 40, "batch_size": 14, 'callbacks': my_callbacks}},
   "OUTLAYER_TRANSFORMERS": 'HampelFilter',
   "BASE_TRANSFORMERS": 'Differencer',
   "SEASON_TRANSFORMERS":None,
   "NORM_TRANSPORMERS": "MinMaxScaler",
   "TRANSFORMERS_ARGS": { 'HampelFilter':{'window_length': 10},
                          'MinMaxScaler': {"feature_range": (0, 1)},
                          'Differencer': {'lags': [1], "drop_na": False},
                          'LogTransformer':{},
                          'BoxCoxTransformer': {},
                           'Deseasonalizer':{'sp': 364, 'model': 'multiplicative'},
                       }
}
```

Приложение Г. Конфигурация третьего эксперимента

```
my_callbacks = [callbacks.EarlyStopping(patience=10, monitor='val_loss')]

CONFIG = {
   "TARGET": {'TICKERS': [
                     'YNDX.ME',
                     'SBER.ME',
                     'POLY.ME',
                     'SIBN.ME',
                     'AMZN',
                     'AAPL',
                     'GOOGL',
                     'NFLX'
                     ],
            'MIN_DATE': '2012-01-01',
            'MAX_DATE': '2022-01-01'},
   'FACTORS': {'TICKERS': [
                     'USDRUB=X',
                     'EURRUB=X',
                     'BZ=F', # Brent
                     'GC=F', # Gold
                     '^GSPC', # S&P 500
                     '^IXIC', # NASDAQ
                     '^DJI', # Dow Jones
                     ],
            'MIN_DATE': '2012-01-01',
            'MAX_DATE': '2022-01-01'
   },
   "VAL_LEN": 7,
   "TEST_LEN": 7,
   "CV_FOLDS": 5,
   "STRATEGIES": ["Recursive", "MiMo", "Direct", 'DirRec', "DirMo"],
   "MODEL": {'INIT':{
               'rnn_arch': rnn_arch,
               'tuner_hp': hp,
               "n_lags": 30,
               "horizon": 7,
               "tuner": "BayesianOptimization",
               "max_trials": 5,
               "loss": 'mae',
               "optimizer": 'adam'},
            'FIT':{"epochs": 40, "batch_size": 14, 'callbacks': my_callbacks}},
   "OUTLAYER_TRANSFORMERS": 'HampelFilter',
   "BASE_TRANSFORMERS": 'Differencer',
   "SEASON_TRANSFORMERS": None,
   "NORM_TRANSPORMERS": "MinMaxScaler",
   "TRANSFORMERS_ARGS": { 'HampelFilter':{'window_length': 10},
                          'MinMaxScaler': {"feature_range": (0, 1)},
                          'Differencer': {'lags': [1], "drop_na": False},
                          'LogTransformer':{},
                          'BoxCoxTransformer': {},
                        }
```

Приложение Д. Пример влияния методов предобработки временного ряда



Приложение Е. Метрики первого эксперимента

Таблица А.1 – Метрики на валидационной выборке.

Удаление выбросов	Удаление сезонности	Базовое преобразование	Нормализация	SMAPE	MAPE	SMAPE Ранг	MAPE Ранг
Нет	Нет	Differencer	MinMaxScaler	1.74	1.73	1.0	1.0
Нет	Deseasonalizer	Differencer	MinMaxScaler	1.76	1.75	2.0	2.0
HampelFilter	Нет	Differencer	MinMaxScaler	1.83	1.82	3.0	3.0
Нет	Нет	BoxCoxTransformer	MinMaxScaler	1.9	1.9	4.0	4.0
Нет	Нет	Нет	MinMaxScaler	1.96	1.95	5.0	5.0
HampelFilter	Deseasonalizer	Differencer	MinMaxScaler	2.09	2.08	6.0	6.0
HampelFilter	Нет	Нет	MinMaxScaler	2.11	2.1	7.0	7.0
HampelFilter	Нет	LogTransformer	MinMaxScaler	2.14	2.12	9.0	8.0
Нет	Нет	Differencer	Нет	2.12	2.13	8.0	9.0
Нет	Deseasonalizer	Нет	MinMaxScaler	2.16	2.15	10.0	10.0
Нет	Нет	LogTransformer	MinMaxScaler	2.26	2.23	12.0	11.0
HampelFilter	Нет	Differencer	Нет	2.22	2.23	11.0	11.0
HampelFilter	Deseasonalizer	BoxCoxTransformer	MinMaxScaler	2.33	2.32	13.0	12.0
Нет	Deseasonalizer	Differencer	Нет	2.36	2.38	14.0	13.0
HampelFilter	Deseasonalizer	Нет	MinMaxScaler	2.41	2.39	15.0	14.0
Нет	Deseasonalizer	LogTransformer	MinMaxScaler	2.42	2.4	16.0	15.0
HampelFilter	Deseasonalizer	LogTransformer	MinMaxScaler	2.48	2.46	17.0	16.0
HampelFilter	Нет	BoxCoxTransformer	MinMaxScaler	2.57	2.54	18.0	17.0
Нет	Deseasonalizer	BoxCoxTransformer	MinMaxScaler	2.65	2.62	20.0	18.0
Auto ARIMA	Auto ARIMA	Auto ARIMA	Auto ARIMA	2.64	2.65	19.0	19.0
HampelFilter	Deseasonalizer	Differencer	Нет	2.68	2.7	21.0	20.0
Нет	Нет	LogTransformer	Нет	2.79	2.78	22.0	21.0
HampelFilter	Нет	LogTransformer	Нет	3.01	2.94	23.0	22.0
Нет	Deseasonalizer	LogTransformer	Нет	3.93	3.84	24.0	23.0
HampelFilter	Deseasonalizer	BoxCoxTransformer	Нет	4.15	4.03	25.0	24.0
Нет	Нет	BoxCoxTransformer	Нет	4.35	4.22	26.0	25.0
HampelFilter	Deseasonalizer	LogTransformer	Нет	4.39	4.24	27.0	26.0
Нет	Deseasonalizer	BoxCoxTransformer	Нет	4.53	4.38	28.0	27.0
HampelFilter	Нет	BoxCoxTransformer	Нет	5.31	5.05	29.0	28.0
Нет	Нет	Нет	Нет	6.66	6.35	30.0	29.0
HampelFilter	Нет	Нет	Нет	7.32	6.9	31.0	30.0
HampelFilter	Deseasonalizer	Нет	Нет	7.86	7.41	32.0	31.0
Нет	Deseasonalizer	Нет	Нет	10.39	9.55	33.0	32.0

Таблица А.2 – Метрики на тестовой выборке.

Удаление выбросов	Удаление сезонности	Базовое преобразование	Нормализация	SMAPE	MAPE	SMAPE Ранг	MAPE Ранг
Нет	Нет	Differencer	Нет	1.86	1.87	1.0	1.0
HampelFilter	Нет	Differencer	Нет	1.9	1.91	2.0	2.0
HampelFilter	Нет	Differencer	MinMaxScaler	2.17	2.14	3.0	3.0
Нет	Нет	Differencer	MinMaxScaler	2.29	2.25	4.0	4.0
HampelFilter	Deseasonalizer	Differencer	Нет	2.48	2.49	5.0	5.0
Нет	Deseasonalizer	Differencer	Нет	2.5	2.5	6.0	6.0
HampelFilter	Deseasonalizer	Differencer	MinMaxScaler	2.69	2.67	7.0	7.0
Нет	Deseasonalizer	Differencer	MinMaxScaler	2.74	2.7	8.0	8.0
Нет	Deseasonalizer	Нет	MinMaxScaler	2.77	2.75	9.0	9.0
Нет	Нет	Нет	MinMaxScaler	2.79	2.77	10.0	10.0
HampelFilter	Нет	Нет	MinMaxScaler	2.9	2.88	11.0	11.0
HampelFilter	Deseasonalizer	Нет	MinMaxScaler	2.94	2.92	12.0	12.0
Нет	Нет	LogTransformer	MinMaxScaler	3.08	3.04	14.0	13.0
HampelFilter	Deseasonalizer	BoxCoxTransformer	MinMaxScaler	3.07	3.05	13.0	14.0
Нет	Нет	BoxCoxTransformer	MinMaxScaler	3.12	3.1	15.0	15.0
HampelFilter	Deseasonalizer	LogTransformer	MinMaxScaler	3.12	3.1	15.0	15.0
Нет	Deseasonalizer	LogTransformer	MinMaxScaler	3.21	3.18	16.0	16.0
HampelFilter	Нет	LogTransformer	MinMaxScaler	3.24	3.21	17.0	17.0
HampelFilter	Нет	BoxCoxTransformer	MinMaxScaler	3.5	3.46	18.0	18.0
Нет	Deseasonalizer	BoxCoxTransformer	MinMaxScaler	3.61	3.58	19.0	19.0
Нет	Нет	LogTransformer	Нет	3.78	3.76	20.0	20.0
HampelFilter	Нет	LogTransformer	Нет	4.07	3.98	21.0	21.0
Нет	Deseasonalizer	LogTransformer	Нет	4.56	4.5	22.0	22.0
Auto ARIMA	Auto ARIMA	Auto ARIMA	Auto ARIMA	4.73	4.77	23.0	23.0
HampelFilter	Deseasonalizer	LogTransformer	Нет	5.3	5.13	24.0	24.0
HampelFilter	Deseasonalizer	BoxCoxTransformer	Нет	5.42	5.29	25.0	25.0
Нет	Deseasonalizer	BoxCoxTransformer	Нет	5.51	5.38	26.0	26.0
Нет	Нет	BoxCoxTransformer	Нет	5.79	5.64	27.0	27.0
HampelFilter	Нет	BoxCoxTransformer	Нет	6.4	6.15	28.0	28.0
HampelFilter	Deseasonalizer	Нет	Нет	8.71	8.24	29.0	29.0
Нет	Нет	Нет	Нет	9.08	8.35	30.0	30.0
HampelFilter	Нет	Нет	Нет	9.97	9.2	31.0	31.0
Нет	Deseasonalizer	Нет	Нет	10.47	9.65	32.0	32.0

Приложение Ж. Конфигурация моделей RNN в рамках проекта Газпром нефти

```
CONFIG = {
   'FACTORS': {'TICKERS': [
                     'USDRUB=X',
                     'EURRUB=X',
                     'BZ=F',
                     ],
            'MIN_DATE': '2013-01-01',
            'MAX_DATE': '2022-01-01'
   },
   "MODEL": {'INIT': {
                'rnn_arch': rnn_arch,
                'tuner_hp': hp,
                "strategy": "MiMo",
                "n_lags": 36,
                "horizon": 6,
                "tuner": "BayesianOptimization”,
                "max_trials": 10,
                "loss": 'mae',
                "optimizer": 'adam'
                },
            'FIT':{"epochs": 40,
                  "batch_size": 9,
                  'callbacks': my_callbacks}
           },
   "OUTLAYER_TRANSFORMERS": 'HampelFilter',
   "BASE_TRANSFORMERS": 'Differencer',
   "SEASON_TRANSFORMERS": None,
   "NORM_TRANSPORMERS": "MinMaxScaler",
   "TRANSFORMERS_ARGS": { 'HampelFilter':{'window_length': 18},
                                          'MinMaxScaler': {"feature_range": (0, 1)},
                                          'Differencer': {'lags': [1], "drop_na": False},
                     },
   'FEATURE_SELECTION': {'ratio':0.3, 'metric':"mae", 'max_iter':100}
}
```

Приложение 3. Алгоритм реализованного решения в рамках проекта Газпром нефти

```
for FCTR in FactorsData do
    FCTR ← FillMissValues(FCTR)
    FCTR ← AggToDecade(FCTR)
end for

for UseFactors in [True, False] do
    for TS in TargetData do
        TS ← FillMissValues(TS)
        TS ← AggToDecade(TS)
        trainId, valId, testId ← split(TS)
        train, val, test ← TS[trainId], TS[valId], TS[testId]
        trainTr ← HampelFilter.FitTransform(train)
        for TR in [Differencer, MinMaxScaler] do
            trainTr ← TR.FitTransform(trainTr)
            valTr ← TR.Transform(val)
        end for

        if UseFactors == True then
            trainF, valF ← FactorsData[trainId], FactorsData[valId]
            trainFTr ← MinMaxScaler.FitTransform(trainF)
            valFTr ← MinMaxScaler.Transform(valF)
        else
            trainFTr ← None
            valFTr ← None
        end if

        model ← RNN(InitParams)
        model.fit(FitParams, trainTr, valTr, trainFTr, valFTr)
        if UseFactors == True then
            ImpFactors ← PermutFeatureImportance(model, trainTr, valTr, trainFTr, valFTr)
            model ← RNN(InitParams)
            model.fit(FitParams, trainTr, valTr, trainFTr[ImpFactors], valFTr[ImpFactors])
        end if

        valPred ← model.predict(trainTr, trainFTr)
        testPred ← model.predict(trainTr, valTr, trainFTr, valFTr)
        for TR in [MinMaxScaler, Differencer] do
            valPred, testPred ← TR.InverseTransform(valPred, testPred)
        end for

        CalculateMetrics(val, valPred)
        CalculateMetrics(test, testPred)
    end for
end for=0
```

АННОТАЦИЯ

В настоящее время данных, собираемых из различных бизнес-процессов деятельности компании, становится все больше. Поэтому зачастую невозможно в ручном режиме прогнозировать все необходимые показатели. Это привело к тому, что появилось стремление к разработке автоматических прогнозных систем, которые не будут требовать активного участия аналитиков. В области прогнозирования финансовых временных рядов на момент проведения исследования наибольший потенциал показывали модели рекуррентных нейронных сетей. В нашем исследовании мы поставили цель - разработать автоматическую прогнозную систему, основанную на моделях рекуррентных нейронных сетей, на языке Python для того, чтобы увеличить точность прогнозирования финансовых временных рядов в рамках процесса бизнес-планирования.

Было проведено три эксперимента, которые были направлены на оценку влияния комбинаций методов предобработки входных данных, стратегий прогнозирования и методов учета информации из нескольких временных рядов на точность прогнозирования моделей рекуррентных нейронных сетей.

На основании проведенного анализа была реализована автоматическая прогнозная система, которую мы применили в рамках реального проекта компании «Газпром нефть». Также были сформулированы основные методические рекомендации по использованию моделей рекуррентных нейронных сетей для прогнозирования финансовых временных рядов. Созданная нами прогнозная система позволила увеличить точность на 1.3%, и снизить среднюю абсолютную процентную ошибку с 4.84 до 3.52%, что является значимым результатом.

Представленные результаты работы могут быть использованы для решения аналогичных бизнес-задач, в которых требуется прогнозирования финансовых показателей.

Ключевые слова и словосочетания: временные ряды (time series), прогнозирование (forecasting), рекуррентная нейронная сеть (recurrent neural network), LSTM, финансы (finance).