Міністерство освіти і науки України
Харківський національний університет імені В. Н. Каразіна
Навчально-науковий інститут комп'ютерних наук та штучного інтелекту
Кафедра комп'ютерних систем та робототехніки

До захисту допущено
Кафедрою комп'ютерних систем та робототехніки
протокол № __ від __ грудня 2025р.

завідувач кафедри _____ Максим ХРУСЛОВ
(підпис)

«___»_____ 2025 р.

Кваліфікаційна робота

здобувача другого (магістерського) рівня вищої освіти

«МОДЕЛЬ АВТОМАТИЗОВАНОЇ СИСТЕМИ ПРОГНОЗУВАННЯ ЕКОНОМІЧНИХ ПОКАЗНИКІВ НА ОСНОВІ ЧАТ-БОТУ»

Спеціальність ***174 – Автоматизація, комп'ютерно-інтегровані технології та робототехніка***
Освітня програма ***Комп'ютеризовані системи управління та автоматика***

Виконавець _____ **О. О. ДУБІНІН**
(підпис)

Науковий керівник _____ **В. Є. СТРІЛЕЦЬ**
(підпис)

Харків – 2025

АНОТАЦІЯ

Пояснювальна записка до кваліфікаційної роботи складається зі вступу, трьох розділів, висновків, списку джерел та двох додатків. Загальний обсяг роботи складає 68 сторінок, із яких 50 сторінки основної частини з 9 рисунками, 5 таблицями, 39 найменувань списку використаних джерел та двома додатками.

Метою кваліфікаційної роботи є забезпечення ефективної та оперативної підтримки прийняття економічних рішень через створення і впровадження моделі автоматизованої системи прогнозування економічних показників на основі чат-боту і рекурентних нейронних мережах.

Об'єкт дослідження – процес автоматизованого короткострокового прогнозування попиту на продукцію підприємства за даними нестаціонарних часових рядів.

Предмет дослідження – математичні моделі глибокого навчання, методи попередньої обробки даних (тригонометричне кодування, Z-нормалізація) та програмні засоби інтеграції інтелектуальних алгоритмів у діалогові системи.

Проблема, яка вирішується в кваліфікаційній роботі, полягає в тому, щоб підвищити точність прогнозування в умовах сезонності та ринкової невизначеності, мінімізувати час на обробку аналітичних даних та забезпечити оперативний доступ до прогнозів з оцінкою ризиків (довірчих інтервалів) для осіб, що приймають рішення.

Область застосування – управління товарними запасами, логістика, ритейл, системи підтримки прийняття рішень. Розроблений програмний продукт може використовуватися в сфері малого та середнього торговельного бізнесу для оптимізації закупівель.

Ключові слова: *прогнозування, часові ряди, нейронні мережі, RNN, LSTM, Python, keras, Telegram API, чат-бот, довірчі інтервали, SMAPE.*

ABSTRACT

The explanatory note consists of an introduction, three chapters, conclusions, references, and two appendices. The total volume of the work is 68 pages, including 50 pages of the main body with 9 figures, 5 tables, 39 references, and two appendices.

The aim of the qualification work is to ensure effective and prompt support for economic decision-making through the creation and implementation of a model of an automated system for forecasting economic indicators based on a chatbot and recurrent neural networks.

The object of research is the process of automated short-term forecasting of demand for enterprise products based on non-stationary time series data.

The subject of research includes mathematical models of deep learning, methods of data preprocessing (trigonometric encoding, Z-normalization), and software tools for integrating intelligent algorithms into dialogue systems.

The problem solved in the qualification work is to increase forecasting accuracy under conditions of seasonality and market uncertainty, minimize the time for processing analytical data, and provide operational access to forecasts with risk assessment (prediction intervals) for decision-makers.

The area of application includes inventory management, logistics, retail, and decision support systems. The developed software product can be used in the sphere of small and medium-sized trading businesses for purchasing optimization.

Keywords: *forecasting, time series, neural networks, RNN, LSTM, Python, keras, Telegram API, chatbot, prediction intervals, SMAPE.*

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАЧЕНЬ

БД	–	База даних
ЕОМ	–	Електронна обчислювальна машина
ПЗ	–	Програмне забезпечення
ACF	–	Autocorrelation Function
API	–	Application Programming Interface
ARIMA	–	AutoRegressive Integrated Moving Average
BPTT	–	Backpropagation Through Time
CLI	–	Command Line Interface
CSV	–	Comma-Separated Values
DL	–	Deep Learning
ERP	–	Enterprise Resource Planning (
GRU	–	Gated Recurrent Unit
JSON	–	JavaScript Object Notation
LSTM	–	Long Short-Term Memory
MAE	–	Mean Absolute Error
ML	–	Machine Learning
PI	–	Prediction Interval
RMSE	–	Root Mean Squared Error
RNN	–	Recurrent Neural Network
sMAPE	–	Symmetric Mean Absolute Percentage Error
SKU	–	Stock Keeping Unit

ВСТУП

У зв’язку зі стрімкою цифровізацією бізнес-процесів та лавиноподібним зростанням обсягів накопичуваних даних (Big Data), все більш актуальною постає проблема переходу від інтуїтивних методів управління до рішень, що базуються на даних (Data Driven Decision Making). В умовах високої волатильності ринків ключовим фактором конкурентоспроможності стає здатність підприємства оперативно та точно передбачати майбутні стани економічної системи.

Актуальність роботи. Серед прикладних областей, де зазначена проблема є вкрай гострою, слід віднести управління товарними запасами та ритейл. Специфіка цієї галузі полягає у необхідності балансування між ризиками дефіциту товару (втрата клієнтів) та затоварення складів (заморожування капіталу). Попит є складною нелінійною величиною, що залежить від сезонності, маркетингових активностей та зовнішніх факторів. Традиційні методи прогнозування, які використовуються на більшості підприємств (експертні оцінки в Excel або прості статистичні моделі типу ковзного середнього), демонструють низьку надійність в умовах нестаціонарності ринку. Вони не здатні виявити приховані нелінійні закономірності, що призводить до значних фінансових втрат. Водночас сучасні потужні інструменти аналітики часто є занадто складними для оперативного персоналу. Отже, задача розробки автоматизованої системи, яка поєднує точність методів глибокого навчання (Deep Learning) із доступністю інтерфейсу (через чат-бот), є вкрай актуальною науково-прикладною проблемою.

Метою дослідження є забезпечення ефективної та оперативної підтримки прийняття економічних рішень через створення і впровадження моделі автоматизованої системи прогнозування економічних показників на основі чат-боту і рекурентних нейронних мережах.

Об’єкт дослідження – це процеси зміни економічних показників підприємства (попиту) у часі та методи їх короткострокового прогнозування в умовах невизначеності.

Методи дослідження: методи системного аналізу, теорія ймовірностей та математична статистика (для аналізу часових рядів), методи глибокого навчання (архітектури RNN, LSTM, GRU), методи попередньої обробки даних (нормалізація, тригонометричне кодування), методи оцінки якості регресійних моделей (метрики MAE, sMAPE).

Предмет дослідження – моделі, алгоритми та програмні засоби побудови нейромережевих систем прогнозування та їх інтеграції з діалоговими інтерфейсами (чат-ботами).

Завдання дослідження:

1. Виконати аналіз існуючого науково-методичного апарата прогнозування часових рядів та обґрунтувати переваги використання рекурентних нейронних мереж над класичними статистичними методами.
2. Розробити методику попередньої обробки даних та конструювання ознак (Feature Engineering), що враховує циклічний характер економічних процесів (сезонність).
3. Спроектувати архітектуру автоматизованої системи та розробити алгоритм оцінки невизначеності прогнозу (побудова довірчих інтервалів) для мінімізації управлінських ризиків.
4. Виконати програмну реалізацію системи та провести експериментальне порівняння ефективності архітектур Simple RNN, LSTM та GRU на реальних статистичних даних, оцінивши їх точність та економічну доцільність впровадження.

РОЗДІЛ 1.
АНАЛІЗ ЗАДАЧІ ПРОГНОЗУВАННЯ ЕКОНОМІЧНИХ ПОКАЗНИКІВ

1.1. Значення точного прогнозування попиту для ефективності підприємства

В умовах сучасної ринкової економіки, що характеризується високим рівнем динамізму, глобалізацією конкуренції та стрімкою цифровізацією бізнес-процесів, ефективність функціонування будь-якого суб'єкта господарювання напряму залежить від якості управлінських рішень. Фундаментальною основою для прийняття таких рішень є *прогнозування* – процес наукового обґрунтування можливих станів об'єкта управління в майбутньому на основі аналізу тенденцій його розвитку в минулому [2, 5].

Серед усіх економічних показників підприємства (собівартість, прибуток, рентабельність тощо) ключову роль відіграє попит на продукцію або послуги. Саме прогноз попиту є відправною точкою для планування всього ланцюга створення вартості: від закупівлі сировини та найму персоналу до управління складськими запасами та розробки маркетингових стратегій [10]. Як зазначають О. Є. Кузьмін та О. Г. Мельник, відсутність достовірного прогнозу перетворює управління підприємством на хаотичне реагування на поточні виклики, що унеможливлює стратегічний розвиток [10].

Основною проблемою, яку вирішує система прогнозування, є *мінімізація невизначеності зовнішнього середовища*. Згідно з даними Державної служби статистики України, коливання споживчого попиту в роздрібній торгівлі можуть сягати 30–50% залежно від сезону та макроекономічних факторів [7]. У таких умовах підприємство змушене балансувати між двома видами ризиків, які детально розглядаються в роботах з економічної кібернетики [17] та моделювання бізнес-процесів [4].

Ризик затоварення (Overstocking) виникає, коли прогнозований попит перевищує фактичний.

Ризик дефіциту (Stockout) виникає, коли фактичний попит перевищує прогнозований.

Клебанова Т. С. у своїх працях з економічної безпеки підкреслює, що обидві ситуації несуть пряму загрозу фінансовій стійкості підприємства [9]. Розглянемо економічні наслідки помилок прогнозування детальніше (табл. 1.1).

Таблиця 1.1.

Порівняльний аналіз економічних наслідків похибки прогнозування попиту

Тип помилки прогнозування	Вплив на логістику та операційну діяльність	Фінансові наслідки для підприємства	Маркетингові та репутаційні ризики
Завищений прогноз (Over-forecasting)	– Переповнення складських площ; – Зростання витрат на зберігання та інвентаризацію; – Псування товарів з обмеженим терміном придатності.	– «Заморожування» оборотного капіталу в неліквідних запасах; – Зниження показника оборотності активів (Asset Turnover); – Касові розриви через відсутність вільної ліквідності.	– Необхідність проведення вимушених розпродажів (Markdown); – Зниження цінності бренду через постійні знижки; – Зменшення середньої маржинальності чека.
Занижений прогноз (Under-forecasting)	– Порушення ритмічності поставок; – Необхідність екстрених закупівель за завищеними цінами; – Наднармові витрати на термінову логістику.	– Втрачена вигода (Lost Profit) від продажів, що не відбулися; – Штрафні санкції за невиконання контрактних зобов'язань (у B2B секторі).	– Зниження лояльності клієнтів (Churn Rate); – Перехід споживачів до конкурентів; – Погіршення іміджу надійного постачальника.

Вплив точності прогнозу на управління запасами. У сучасній логістиці домінуючою є концепція мінімізації втрат, відома як *Just-in-Time* («Саме вчасно»). Її реалізація неможлива без високоточних математичних моделей. Ключовим параметром, що зв'язує прогноз із реальними запасами, є *страховий запас* (Safety Stock). Він необхідний для компенсації випадкових коливань попиту та помилок прогнозування.

Класична формула розрахунку страхового запасу має вигляд:

$$SS = Z \cdot \sigma_E \cdot \sqrt{L},$$

де Z – коефіцієнт, що відповідає заданому рівню обслуговування (Service Level) (наприклад, 1.65 для 95%), σ_E – стандартне відхилення помилки прогнозу (Forecast Error Standard Deviation), L – час виконання замовлення (Lead Time).

З наведеної формули випливає пряма залежність: чим менша похибка прогнозу (σ_E), тим менший страховий запас потрібен підприємству.

Згідно з дослідженнями Р. Гайндмана [29], підвищення точності прогнозування лише на 1% дозволяє скоротити рівень страхових запасів на 5–10% без зниження рівня сервісу. У масштабах великого ритейлера це означає вивільнення мільйонів грошових одиниць, які раніше були «заморожені» на складах. Це підтверджує тезу про те, що інвестиції в розробку сучасних моделей прогнозування (зокрема на базі нейронних мереж) мають високий рівень повернення інвестицій (ROI).

Проблематика прогнозування в умовах «Великих даних». Традиційні підходи до прогнозування, описані в класичних підручниках з економетрики [12, 20], базувалися на припущенні про лінійність та стаціонарність економічних процесів. Проте сучасні торговельні мережі оперують тисячами товарних позицій (SKU), динаміка продажів яких має складний, нелінійний характер.

На попит впливає сукупність різнорідних факторів:

1) *сезонність*: тижнева (піки у вихідні), місячна (день отримання зарплати), річна (свята);

2) *цінова еластичність*: реакція покупців на зміну ціни;

3) *маркетингова активність*: наявність промо-акцій, банерів, спеціальних пропозицій;

4) календарні ефекти: «плаваючі» свята (Великдень), кількість робочих днів у місяці.

Як зазначає С. Сміл [35], прості методи екстраполяції (наприклад, ковзне середнє або наївний прогноз) не здатні врахувати ці фактори. Вони дають значну похибку в періоди промо-акцій або різкої зміни тренду, що робить їх непридатними для оперативного управління в умовах жорсткої конкуренції.

Водночас, ручна обробка даних аналітиками за допомогою електронних таблиць (Excel) є трудомісткою і не дозволяє оперативно реагувати на зміни ринку. Це актуалізує потребу в автоматизації процесу прогнозування з використанням методів інтелектуального аналізу даних (Data Mining) та машинного навчання [19].

Таким чином, точне прогнозування попиту є не просто статистичною задачею, а критично важливим бізнес-процесом, що безпосередньо впливає на ліквідність та прибутковість підприємства. Існуючі виклики – зростання обсягів даних, нелінійність попиту та висока ціна управлінської помилки – вимагають переходу від інтуїтивних та простих статистичних методів до побудови автоматизованих інтелектуальних систем.

Розробка такої системи, яка б поєднувала потужність сучасних алгоритмів (зокрема рекурентних нейронних мереж) зі зручністю використання для кінцевого користувача, є актуальною науково-прикладною задачею. Саме аналізу методів, здатних реалізувати цю мету, присвячено наступний підрозділ.

1.2. Порівняльний аналіз сучасних методів прогнозування: від статистики до нейромереж

Еволюція методів аналізу та прогнозування часових рядів є відображенням загального розвитку обчислювальної техніки та науки про дані (Data Science). Якщо у XX столітті домінували параметричні статистичні підходи, що вимагали суворих припущень щодо природи даних, то сьогодні акцент змістився у бік адаптивних алгоритмів машинного та глибокого навчання, здатних виявляти приховані закономірності у великих масивах інформації (Big Data).

У сучасній науковій літературі [2, 5, 29] прийнято класифікувати методи прогнозування на три великі групи:

1. Класичні статистичні методи (часові ряди як стохастичні процеси).
2. Методи машинного навчання (Machine Learning, ML).
3. Методи глибокого навчання (Deep Learning, DL), зокрема рекурентні нейронні мережі.

Класичні статистичні методи

Фундаментом прогнозування є статистичні моделі, які базуються на екстраполяції минулих тенденцій у майбутнє. До найпоширеніших методів цієї групи належать авторегресійні моделі.

Класичною моделлю, яка довгий час вважалася еталоном («золотим стандартом») для одновимірних часових рядів, є **ARIMA** (AutoRegressive Integrated Moving Average), розроблена Боксом і Дженкінсом [20]. Вона описує майбутнє значення ряду як лінійну комбінацію його минулих значень та помилок прогнозу.

Математично модель ARIMA(p,d,q) ефективна для даних, які є стаціонарними (мають постійне середнє значення та дисперсію). Якщо дані нестаціонарні (наприклад, мають тренд зростання продажів), застосовується процедура диференціювання (інтегрування порядку d).

Проте, як зазначають Г. П. Жанг [40] та Р. Гайндман [29], статистичні методи мають суттєві обмеження в сучасних бізнес-умовах:

- лінійність. ARIMA та експоненційне згладжування (Holt-Winters) погано моделюють складні нелінійні залежності, характерні для споживчого попиту;
- проблема мультиколінеарності. Їм важко врахувати вплив багатьох зовнішніх факторів (екзогенних змінних), таких як ціна конкурентів, маркетингові бюджети або погода;
- чутливість до викидів. Одиночна аномалія (наприклад, ажіотажний попит під час «Чорної п'ятниці») може спотворити параметри моделі на тривалий період.

Метод **Seasonal Naive** (сезонний наївний), описаний у [35], часто використовується як базовий рівень (baseline) для оцінки ефективності складніших моделей. Його логіка проста: прогноз на завтра дорівнює значенню цього ж дня у попередньому сезоні (році або тижні). Попри свою примітивність, він є важливим індикатором: якщо розроблена нейромережа показує точність нижчу за Naive-метод, її використання є недоцільним.

Методи машинного навчання (Machine Learning)

З розвитком алгоритмічних підходів набули популярності методи, що не роблять жорстких припущень про розподіл даних. До них відносяться:

- **Support Vector Regression (SVR)** – метод опорних векторів для регресії;
- **Random Forest** та **Gradient Boosting** (XGBoost, LightGBM, CatBoost) – ансамблеві методи на основі дерев рішень [15, 33].

Згідно з дослідженнями М. В. Полякова [15], ці алгоритми значно краще справляються з нелінійністю та дозволяють легко додавати будь-яку кількість додаткових ознак (features). Наприклад, модель може врахувати, що «якщо ціна знижена на 10% І сьогодні п'ятниця, то продажі зростуть на 30%».

Головним недоліком класичного ML для часових рядів є відсутність розуміння послідовності подій. Для алгоритму Random Forest набір даних – це просто «мішок» прикладів, де порядок рядків не має значення. Щоб така модель «побачила» час, інженер даних повинен вручну створити лагові змінні

(lag features: $t-1, t-7, t-30$), що значно ускладнює процес підготовки даних (Feature Engineering) [19, 32].

Нейронні мережі та глибоке навчання

Справжній прорив у прогнозуванні послідовних даних відбувся з появою та вдосконаленням штучних нейронних мереж (Artificial Neural Networks – ANN). На відміну від традиційних алгоритмів, нейромережі є універсальними апроксиматорами, здатними моделювати функції будь-якої складності [6, 11].

Для роботи з часовими рядами було розроблено спеціальний клас архітектур – **рекурентні нейронні мережі** (Recurrent Neural Networks – RNN).

Як пояснюють у своїй монографії Є. В. Бодянський та О. Г. Руденко [3], ключовою відмінністю RNN від звичайних мереж прямого поширення є наявність зворотних зв'язків. Вихід нейрона на поточному кроці t залежить не лише від вхідних даних x_t, але й від прихованого стану (hidden state) h_{t-1}, який зберігає інформацію про попередні кроки. Це дозволяє мережі мати «короткострокову пам'ять» і розуміти контекст динаміки ряду.

Проте класичні RNN (Simple RNN) зіткнулися з проблемою «зникаючого градієнта» (vanishing gradient problem) [25]. При навчанні на довгих послідовностях (наприклад, аналіз продажів за рік) градієнти помилки, що передаються зворотно у часі, стають нескінченно малими, і мережа перестає навчатися на віддалених залежностях.

Для вирішення цієї проблеми були розроблені вдосконалені архітектури, які є предметом дослідження у роботі:

1. **LSTM (Long Short-Term Memory).** Запропонована Хохрайтером і Шмідхубером [27] у 1997 році. Архітектура LSTM вводить поняття «комірки пам'яті» (cell state) і трьох логічних вентилів (gates): вхідного, вихідного та вентиля забування (forget gate). Це дозволяє мережі вибірково «запам'ятовувати» важливі події на довгий час і «забувати» шум. А. Грейвс [26] довів ефективність LSTM для складних послідовностей.

2. **GRU (Gated Recurrent Unit).** Запропонована у 2014 році Чо та ін. [23] як спрощена версія LSTM. Вона об'єднує вентилі забування та входу в єдиний вентиль оновлення (update gate). GRU часто демонструє схожу з LSTM точність, але потребує менше обчислювальних ресурсів для навчання [18, 39].

Останніми роками також набирають популярності механізми уваги (Attention Mechanisms) та трансформери [38], проте для задач прогнозування числових рядів середньої довжини RNN, LSTM та GRU залишаються найбільш надійним та перевіреним інструментом [21, 31].

Узагальнений порівняльний аналіз розглянутих підходів наведено у таблиці 1.2.

Таблиця 1.2.

Порівняльна характеристика методів прогнозування економічних показників

Критерій порівняння	**Статистичні методи (ARIMA, ES)**	**Традиційне машинне навчання (RF, SVR)**	**Глибоке навчання (RNN, LSTM, GRU)**
Здатність до моделювання нелінійності	Низька (тільки лінійні залежності)	Висока	Дуже висока (універсальні апроксиматори)
Врахування сезонності та циклічності	Вимагає попередньої декомпозиції ряду	Вимагає ручного створення ознак (Feature Engineering)	Виявляється автоматично в процесі навчання
Стійкість до шумів та викидів	Низька (викиди спотворюють прогноз)	Середня (залежить від алгоритму)	Висока (особливо при використанні робастних функцій втрат)
Вимоги до обсягу даних	Мінімальні (працюють на малих вибірках)	Середні	Високі (потребують репрезентативної історії)
Обчислювальна складність	Низька (секунди)	Середня (хвилини)	Висока (потребує GPU/TPU)
Інтерпретація моделі	Прозора («Біла скринька»)	Часткова (Feature Importance)	Ускладнена («Чорна скринька»)

На основі проведеного аналізу можна зробити висновок, що для вирішення задачі автоматизованого прогнозування попиту в умовах цифрової економіки найбільш перспективним є використання методів глибокого навчання, а саме рекурентних нейронних мереж.

Незважаючи на вищу обчислювальну складність, архітектури RNN, LSTM та GRU мають вирішальні переваги для даної предметної області:

1. End-to-end навчання: Вони здатні самостійно виділяти інформативні ознаки з "сирих" історичних даних, зменшуючи потребу в ручній роботі експерта [13, 22].

2. Робота з контекстом: Здатність враховувати послідовність подій дозволяє краще прогнозувати попит у періоди свят або після проведення промо-акцій.

3. Масштабованість: Нейромережеві моделі легко адаптуються до збільшення кількості товарних позицій без необхідності переналаштування структури моделі [36].

Саме тому в практичній частині роботи буде проведено порівняльне дослідження ефективності трьох архітектур (Simple RNN, LSTM, GRU) для визначення оптимальної моделі, що ляже в основу розроблюваної автоматизованої системи.

1.3. Особливості застосування чат-ботів як інтерфейсу для автоматизованих бізнес-систем

В умовах цифрової трансформації економіки змінюються не лише методи обробки даних, але й парадигми взаємодії людини з комп'ютерними системами. Традиційні графічні інтерфейси (GUI), реалізовані у вигляді десктопних програм або веб-сайтів, поступово поступаються місцем діалоговим інтерфейсам (Conversational UI – CUI). Найпоширенішою формою реалізації CUI є ***чат-боти*** – програмні агенти, що імітують людську комунікацію та інтегровані у звичні для користувача месенджери [1, 34].

Для систем підтримки прийняття рішень (СППР), до яких належить розроблювана система прогнозування, швидкість та зручність доступу до інформації є критичними факторами. Як зазначають В. В. Литвин та В. А. Висоцька [11], ефективність СППР визначається не лише точністю математичних моделей, а й здатністю доставити правильну інформацію потрібній особі в потрібний час.

Еволюція інтерфейсів у бізнес-системах. Історично автоматизація прогнозування проходила декілька етапів:

1. Електронні таблиці (Excel) вимагають ручного введення даних, складні для масштабування, не дозволяють працювати з телефону.

2. BI-системи (Business Intelligence) – потужні дашборди (PowerBI, Tableau), які дають глибоку аналітику, але є дорогими у впровадженні, перевантаженими функціоналом та вимагають навчання персоналу.

3. Мобільні додатки забезпечують мобільність, але розробка окремого додатку під iOS та Android є фінансово затратною. Крім того, користувачі неохоче встановлюють нові додатки для вузьких задач («втома від додатків»).

4. Чат-боти вирішують проблему «останньої милі» в доставці даних. Користувач залишається у звичному середовищі (Telegram, Viber, Slack), отримуючи складну аналітику через простий діалог.

Класифікація та архітектура чат-ботів

Згідно з класифікацією, наведеною А. Шеват [34], чат-боти поділяються на дві основні категорії:

1. *Декларативні (Rule-based)* працюють за чітко заданим сценарієм (деревом рішень). Взаємодія відбувається через команди (наприклад, /predict) або кнопки меню. Вони є надійними, передбачуваними та дешевими у розробці.

2. *Інтелектуальні (AI-based)* використовують методи обробки природної мови (NLP) для розуміння вільного тексту.

Для задач економічного прогнозування, де важлива точність передачі даних (завантаження CSV-файлів, отримання конкретних цифр), оптимальним

є використання *гібридного підходу*: чітка командна логіка для запуску процесів прогнозування та елементи NLP для інтерпретації запитів користувача.

Архітектурно сучасний чат-бот для бізнесу складається з трьох рівнів [1]:

- Інтерфейсний рівень (Frontend) – платформа месенджера (наприклад, Telegram), яка надає API для відображення тексту, графіків та кнопок.
- Логічний рівень (Backend) – серверна частина (наприклад, на мові Python), яка обробляє запити, звертається до бази даних або запускає обчислювальні скрипти.
- Інтелектуальне ядро (AI Engine). У контексті даної роботи – це модуль на базі нейронних мереж RNN/LSTM, який виконує прогнозування.

Переваги використання чат-ботів у прогнозуванні

Впровадження чат-боту як інтерфейсу до системи прогнозування надає підприємству низку конкурентних переваг, порівняно з традиційним софтом. Порівняльний аналіз наведено у таблиці 1.3.

Таблиця 1.3.

Порівняльна характеристика інтерфейсів систем прогнозування

Критерій порівняння	Веб-портал / Дашборд	Нативний мобільний додаток	Чат-бот у месенджері
Швидкість доступу	Середня (потрібен браузер, логін)	Висока	Дуже висока (миттєвий доступ)
Вартість розробки (TCO)	Висока (Frontend + Backend)	Дуже висока (iOS + Android)	Низька (тільки Backend)
Вимоги до пристрою	ПК або планшет	Смартфон певної версії ОС	Будь-який пристрій з месенджером
Оновлення ПЗ	Непомітне для користувача	Вимагає завантаження оновлень	Миттєве на стороні сервера
Спосіб взаємодії	Кліки мишею, фільтри	Тапи, жести	Діалог, передача файлів, команди
Push-сповіщення	Складно реалізувати (email)	Можливо	Вбудована функція (миттєво)

Окремо слід виділити можливості *Telegram Bot API* [37], який став де-факто стандартом для корпоративних ботів в Україні. Він дозволяє:

- Безпечно передавати файли (документи .csv, звіти).
- Відправляти зображення (графіки прогнозів .png) без стиснення.
- Створювати інтерактивні клавіатури для вибору параметрів прогнозу (наприклад, горизонт прогнозування 7 або 14 днів).
- Забезпечувати авторизацію користувачів за ID, що гарантує конфіденційність комерційних даних.

Роль чат-боту в розроблюваній системі

У кваліфікаційному дослідженні чат-бот виконує роль *єдиного вікна* (Single Point of Entry) для взаємодії з математичною моделлю. Він автоматизує рутинні операції, які раніше виконувалися аналітиками вручну:

1) прийом даних: менеджер просто перетягує файл з продажами у вікно чату.

2) валідація: бот миттєво перевіряє структуру файлу і повідомляє про помилки.

3) запуск обчислень: бот ініціює роботу Python-скрипту на сервері, передаючи йому нові дані.

4) візуалізація: бот повертає результат не у вигляді сухої таблиці, а у вигляді графіку з довірчими інтервалами (Prediction Intervals), який зручно аналізувати на екрані смартфона.

Використання чат-боту дозволяє трансформувати складну математичну задачу прогнозування у простий та інтуїтивно зрозумілий діалог. Це знижує поріг входження для персоналу, прискорює процес прийняття рішень та робить аналітику доступною в режимі 24/7. Поєднання потужності рекурентних нейронних мереж (Back-end) та зручності месенджера (Front-end) створює синергетичний ефект, що підвищує загальну ефективність системи управління запасами.

Висновки до розділу 1

У розділі проведено комплексний аналіз проблеми прогнозування економічних показників підприємства в умовах цифрової трансформації.

Встановлено, що точність прогнозування попиту є критичним фактором, який безпосередньо впливає на фінансову стійкість бізнесу. Похибки прогнозу призводять до двох видів ризиків: затоварення складів (заморожування капіталу) та дефіциту продукції (втрата прибутку і клієнтів). В умовах сучасних логістичних концепцій (Just-in-Time) навіть незначне підвищення точності прогнозу дозволяє суттєво зменшити рівень страхових запасів.

Порівняльний аналіз методів прогнозування показав, що класичні статистичні моделі (ARIMA, експоненційне згладжування) мають суттєві обмеження при роботі з нелінійними нестаціонарними даними, характерними для роздрібної торгівлі. Доведено, що найбільш перспективним інструментом є методи глибокого навчання, а саме рекурентні нейронні мережі (RNN, LSTM, GRU). Вони здатні автоматично виявляти складні залежності, враховувати сезонність та вплив маркетингових активностей без необхідності ручного конструювання ознак.

Обґрунтовано неефективність традиційних складних аналітичних звітів для оперативного прийняття рішень. Визначено, що інтеграція математичних моделей з інтерфейсом чат-боту (Conversational UI) дозволяє вирішити проблему «останньої милі» в аналітиці. Використання месенджерів (зокрема Telegram) забезпечує миттєвий доступ до прогнозів, мобільність персоналу та низьку вартість впровадження системи.

На основі проведеного аналізу сформовано концепцію розробки автоматизованої системи, яка поєднує потужність нейромережевих алгоритмів (Backend) зі зручністю діалогового інтерфейсу (Frontend).

Отримані висновки є підґрунтям для переходу до другого розділу, в якому буде розглянуто математичні основи функціонування обраних архітектур нейронних мереж та спроектовано структуру майбутньої системи.

РОЗДІЛ 2.
ПРОЕКТУВАННЯ МОДЕЛІ ТА АРХІТЕКТУРИ СИСТЕМИ ПРОГНОЗУВАННЯ

2.1. Математичні основи функціонування рекурентних нейронних мереж

Прогнозування часових рядів відноситься до класу задач *sequence-to-sequence learning* (навчання на послідовностях). На відміну від класичних нейронних мереж прямого поширення (Feed Forward Neural Networks), де припускається, що всі вхідні вектори є незалежними один від одного, для часових рядів критично важливим є порядок надходження даних. Попит на товар у момент часу t статистично залежить від попиту в моменти $t-1, t-2, \dots, t-n$.

Для моделювання таких залежностей використовуються **рекурентні нейронні мережі (RNN)**. Їхньою ключовою особливістю є наявність зворотних зв'язків, які дозволяють зберігати інформацію про попередні стани системи у вигляді внутрішнього вектора стану («пам'яті») [3, 6].

Архітектура простої рекурентної мережі (Simple RNN). У найпростішому варіанті (Simple RNN або Elman Network) нейронна мережа обробляє послідовність входів $X = \{x_1, x_2, \dots, x_T\}$ крок за кроком. На кожному кроці t мережа обчислює прихований стан (hidden state) h_t, використовуючи поточний вхід x_t та попередній прихований стан h_{t-1}.

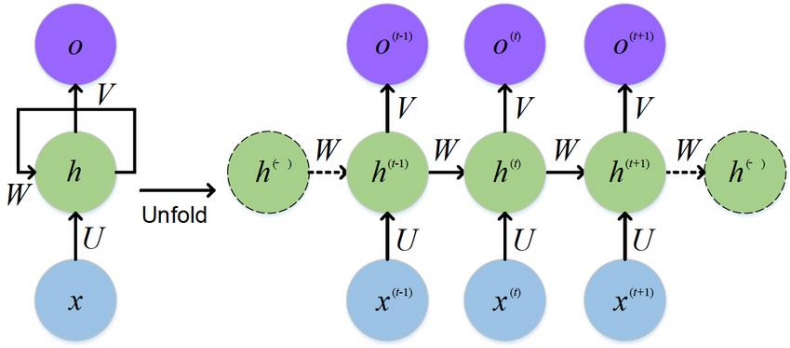

Рисунок 2.1 – Схема розгортання рекурентної нейронної мережі у часі

Математична модель Simple RNN описується системою рівнянь:

$$h_t = tanh\,(W_{xh}\,x_t + W_{hh}h_{t-1} + b_h),$$

$$y_t = W_{hy}\,h_t + b_y,$$

де

x_t – вхідний вектор ознак у момент часу t (наприклад, продажі, ціна, день тижня);

h_t – вектор прихованого стану (пам'ять мережі);

y_t – вихідне значення (прогноз);

W_{xh}, W_{hh}, W_{hy} – матриці вагових коефіцієнтів (вхід-прихований шар, рекурентний зв'язок, прихований шар-вихід);

b_h, b_y – вектори зміщення (bias);

$tanh$ – функція активації (гіперболічний тангенс), що нормалізує значення в діапазоні $[-1;\ 1]$.

Навчання RNN відбувається методом зворотного поширення помилки у часі (Backpropagation Through Time – BPTT) [25]. Проте, як зазначають дослідники [13, 21], Simple RNN має суттєвий недолік – проблему зникаючого градієнта (Vanishing Gradient Problem). При обчисленні градієнтів для оновлення ваг на довгих послідовностях відбувається багаторазове множення матриці ваг на похідну функції активації. Якщо ці значення менші за одиницю, градієнт експоненційно зменшується, прямуючи до нуля. Це призводить до того, що мережа «забуває» події, які сталися багато кроків тому (наприклад, сезонність продажів піврічної давності).

Мережі з довгою короткостроковою пам’яттю (LSTM). Для вирішення проблеми довгострокових залежностей у 1997 році С. Хохрайтер та Ю. Шмідхубер запропонували архітектуру Long Short-Term Memory (LSTM) [27] (рис. 2.2).

Ключова ідея LSTM полягає у введенні окремого потоку інформації – стану комірки (cell state, C_t), який проходить крізь весь ланцюжок мережі з мінімальними лінійними перетвореннями. Регулювання потоку інформації

здійснюється за допомогою спеціальних структур – гейтів (gates), які використовують сигмоїдальну функцію активації ($\sigma(x) \in [0; 1]$):

- 0 – не пропускати нічого;
- 1 – пропустити все.

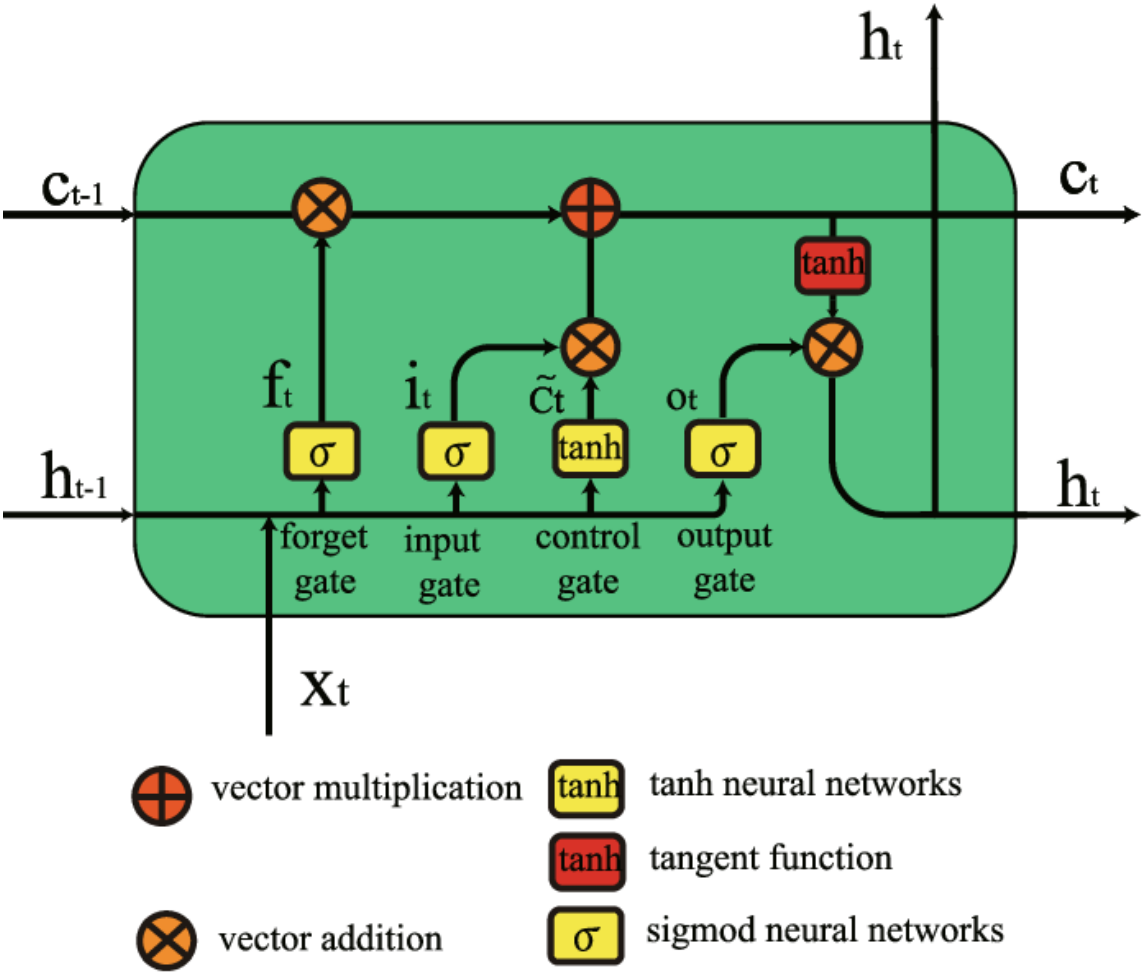

Рисунок 2.2 – Внутрішня структура комірки LSTM

Математична модель комірки LSTM на кроці t складається з наступних етапів:

1. Гейт забуття (Forget Gate, f_t) вирішує, яку частину інформації з попереднього стану комірки C_{t-1} слід видалити:

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f).$$

2. Вхідний гейт (Input Gate, i_t) вирішує, яку нову інформацію записати в комірку. Паралельно створюється вектор кандидатів $\widetilde{C_t}$:

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i),$$
$$\widetilde{C_t} = tanh(W_C \cdot [h_{t-1}, x_t] + b_C).$$

3. Оновлення стану комірки (C_t). Стара пам'ять множиться на фактор забування, і додається нова важлива інформація:

$$C_t = f_t * C_{t-1} + i_t * \widetilde{C_t}\,.$$

4. Вихідний гейт (Output Gate, o_t) формує прихований стан h_t на основі оновленої пам'яті:

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o),$$
$$h_t = o_t * tanh(C_t).$$

Така архітектура дозволяє LSTM ефективно моделювати як короткострокові коливання попиту (тижневі), так і довгострокові тренди [26].

Керовані рекурентні блоки (GRU). Архітектура **Gated Recurrent Unit (GRU)**, запропонована у 2014 році Чо та ін. [23], є оптимізованою модифікацією LSTM. Вона зберігає здатність боротися зі зникаючим градієнтом, але має простішу структуру, що зменшує кількість параметрів та пришвидшує навчання.

Основні відмінності GRU від LSTM:

1. Об'єднання стану комірки C_t та прихованого стану h_t в єдиний вектор.

2. Використання лише двох гейтів: оновлення (Update Gate, z_t) та скидання (Reset Gate, r_t).

Структура GRU показана на рис. 2.3.

Математичний опис GRU:

1. Гейт скидання (r_t) визначає, наскільки важливий попередній прихований стан для обчислення нового кандидата:

$$r_t = \sigma(W_r \cdot [h_{t-1}, x_t]).$$

2. Гейт оновлення (z_t) відіграє роль одночасно і вхідного гейту, і гейту забування з LSTM. Він визначає баланс між збереженням старої пам'яті та записом нової:

$$z_t = \sigma(W_z \cdot [h_{t-1}, x_t]).$$

3. Кандидат у прихований стан $(\widetilde{h_t}) = tanh(W \cdot [r_t * h_{t-1}, x_t])$.

4. Фінальний прихований стан (h_t) – лінійна інтерполяція між минулим станом і новим кандидатом:

$$h_t = (1 - z_t) * h_{t-1} + z_t * \widetilde{h_t}.$$

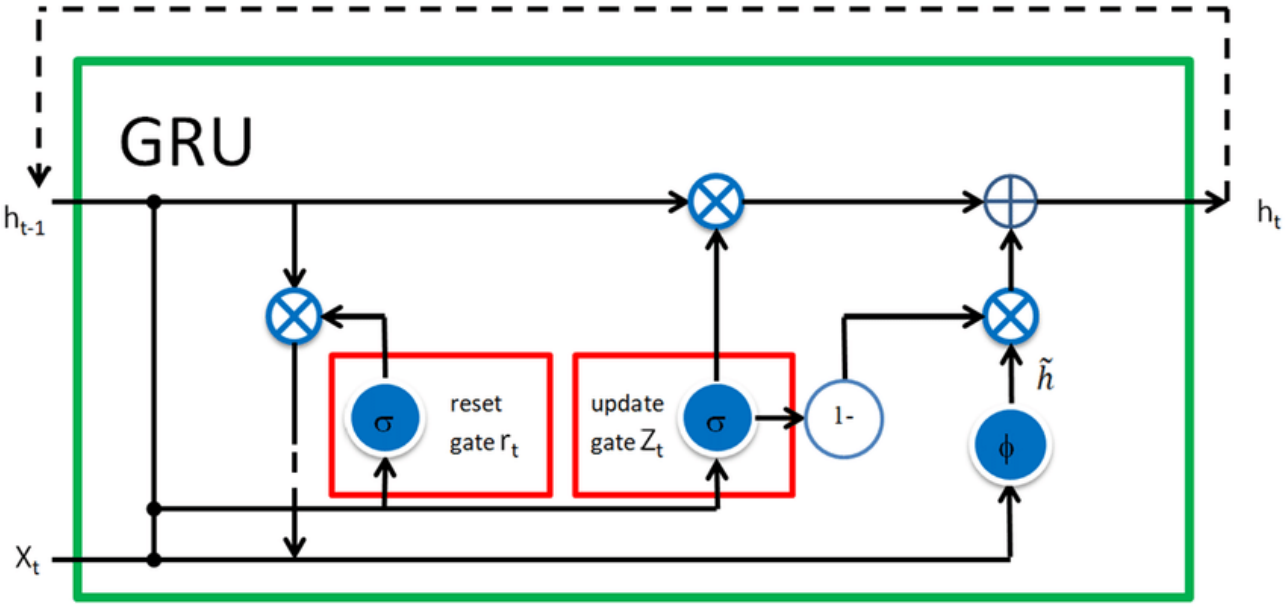

Рисунок 2.3 – Внутрішня структура GRU

Згідно з порівняльними дослідженнями [18, 39], GRU часто показує результати, порівнянні з LSTM, при цьому вимагаючи менше обчислювальних ресурсів. Проте LSTM може мати перевагу на задачах з дуже довгими послідовностями та великим обсягом даних.

Аналіз математичних основ показує, що для задачі прогнозування економічних часових рядів, які характеризуються наявністю складних залежностей і шумів, доцільно використовувати архітектури LSTM та GRU. Водночас, Simple RNN може бути ефективною на коротких горизонтах прогнозування або при обмеженій кількості даних, де складні моделі схильні до перенавчання. Саме тому в експериментальній частині роботи (розділ 3) буде проведено емпіричне порівняння всіх трьох архітектур.

2.2. Метод попередньої обробки даних та конструювання часових ознак

Ефективність навчання нейронних мереж критично залежить від якості вхідних даних. У спільноті Data Science загальновідомим є принцип «Garbage In, Garbage Out» (Сміття на вході – сміття на виході). Тому етап попередньої обробки (Data Preprocessing) та конструювання ознак (Feature Engineering) є не менш важливим, ніж вибір архітектури мережі [19].

Для розроблюваної системи було розроблено конвеєр обробки даних (pipeline), реалізований мовою Python з використанням бібліотек *Pandas* [32] та *Scikit-learn* [33]. Процес складається з чотирьох послідовних етапів.

Очищення та відновлення цілісності часового ряду

Реальні дані про продажі, отримані з ERP-систем підприємства, часто містять шуми, пропуски та аномалії. Першим кроком є приведення даних до регулярного вигляду.

Вхідний набір даних D представляє собою таблицю, де кожний запис містить дату t, ідентифікатор товару SKU, ціну P_t, кількість проданого товару Q_t та ознаку промо-акції $Promo_t$.

Алгоритм обробки містить:

1. Ресемплінг (Resampling) – приведення часової шкали до регулярної частоти (денної). Якщо в певний день продажі були відсутні, система автоматично створює запис із нульовим попитом ($Q_t = 0$), що відповідає реальній відсутності транзакцій, а не втраті даних.

2. Заповнення пропусків у екзогенних змінних. Для ціни (P_t) використовується метод *Forward Fill* ($P_t = P_{t-1}$), оскільки ціна не може дорівнювати нулю у вихідні дні, а залишається такою ж, як у останній робочий день.

Забезпечення стаціонарності: перехід до різницевих значень

Більшість економічних рядів є нестаціонарними (мають тренд та змінну дисперсію). Нейронні мережі погано екстраполюють тренди, які виходять за межі діапазону значень, бачених при навчанні [21].

Для вирішення цієї проблеми у роботі застосовано метод диференціювання (Differencing). Замість прогнозування абсолютного значення продажів Q_t, модель навчається прогнозувати зміну попиту $\Delta\, Q_t$:

$$\Delta\, Q_t \;=\; Q_t \;-\; Q_{t-1}.$$

Такий підхід (реалізований у програмному коді як variant='delta') дозволяє привести математичне сподівання ряду до константи, близької до нуля, що значно прискорює збіжність градієнтного спуску.

Тригонометричне кодування циклічних часових ознак

Однією з ключових інновацій у розробленій моделі є спосіб подання календарної інформації.

Традиційний підхід, при якому день тижня кодується цілим числом від 0 (понеділок) до 6 (неділя), є хибним для нейронних мереж. Мережа сприймає ці числа як величину, тому для неї відстань між неділею (6) та понеділком (0) дорівнює $|6 - 0| = 6$, хоча у часі це сусідні дні (відстань 1).

Для коректного моделювання циклічності (тижневої та річної сезонності) використано тригонометричне кодування (Cyclical Encoding). Кожна часова мітка t розкладається на дві компоненти – синус та косинус кута на одиничному колі.

1. Кодування дня тижня (dow). Період $T_{week} \;=\; 7$:

$$dow_{sin} \;= \sin\left(\frac{2\pi * dow}{7}\right),\; dow_{cos} = \cos\left(\frac{2\pi * dow}{7}\right).$$

2. Кодування дня року (doy). Період $T_{year} \;=\; 365.25$:

$$doy_{sin} \sin\left(\frac{2\pi * doy}{365{,}25}\right), doy_{cos} = \cos\left(\frac{2\pi * doy}{365{,}25}\right).$$

Таке перетворення забезпечує безперервність часового простору: значення для 31 грудня плавно переходить у значення для 1 січня, а неділя – у понеділок. Це дозволяє нейромережі (RNN/LSTM) ефективно вивчати сезонні патерни без розривів у логіці [14].

Масштабування ознак (Feature Scaling)

Вхідні дані мають різний фізичний зміст та масштаб:

- продажі (Q_t): 0...200 одиниць;

- ціна (P_t): 80...120 грошових одиниць;
- промо ($Promo_t$): 0 або 1;
- синус/косинус: -1...1.

Якщо подати ці дані в мережу без обробки, ваги нейронів, що відповідають за продажі, будуть оновлюватися набагато інтенсивніше, ніж ваги для промо-акцій, що призведе до нестабільності навчання.

Для усунення дисбалансу застосовано Z-нормалізацію (Standardization) за допомогою класу StandardScaler бібліотеки Scikit-learn. Для кожної ознаки x виконується перетворення:

$$x'_{norm} = \frac{x-\mu}{\sigma},$$

де μ – середнє значення вибірки, σ – стандартне відхилення.

Після нормалізації всі вхідні дані мають нульове середнє та одиничну дисперсію, що є оптимальним для функцій активації типу tanh, які використовуються в LSTM/GRU.

Формування тензорів для навчання

Для навчання рекурентної мережі табличні дані перетворюються на тривимірні масиви (тензори) методом ковзного вікна (Sliding Window Method).

Формат тензора: Samples, Time Steps, Features.

Параметри вікна, визначені емпіричним шляхом у файлі конфігурації meta.json:

- Lookback window ($L = 84$). Мережа "бачить" історію за останні 12 тижнів (3 місяці). Це дозволяє охопити як короткострокову динаміку, так і квартальні тренди.
- Forecast Horizon ($H = 14$). Прогноз будується на 2 тижні вперед.

Таким чином, для кожного моменту часу t формується навчальна пара (X, Y):

- вхід X – матриця розмірністю $84 \times N_{features}$ (де $N_{features}$ – кількість ознак, включаючи ціну, промо та sin/cos дати);
- вихід Y – вектор розмірністю 14 (майбутні продажі).

Такий метод попередньої підготовки дозволяє перетворити задачу прогнозування часового ряду на задачу навчання з учителем (Supervised Learning), придатну для обробки алгоритмами Backpropagation [16].

Запропонований метод попередньої обробки даних, що включає перехід до різницевих значень, Z-нормалізацію та інноваційне тригонометричне кодування дат, дозволяє сформувати якісний навчальний набір. Це усуває проблеми нестаціонарності та розривів у сезонних циклах, створюючи надійний фундамент для ефективного навчання моделей RNN, LSTM та GRU, архітектуру яких розглянуто в попередньому підрозділі.

2.3. Проектування архітектури системи «Клієнт – Чат-бот – Сервер»

При проектуванні автоматизованої системи прогнозування було обрано мікросервісний підхід до архітектури. Це означає чітке розмежування відповідальності між модулями: інтерфейс не повинен займатися складними математичними обчисленнями, а обчислювальне ядро не повинно залежати від специфіки месенджера [1].

Загальна архітектура системи реалізована за класичною схемою «Клієнт-Сервер» з використанням Telegram Bot API як проміжного шару (Middleware).

Структурна схема взаємодії модулів

Система складається з трьох логічних рівнів:

1. *Рівень Клієнта* (User Tier) – це пристрій користувача (смартфон або ПК) із встановленим месенджером Telegram. Користувач взаємодіє із системою через текстові команди та передачу файлів.

2. *Рівень Інтерфейсу* (Interface Tier) – хмарна інфраструктура Telegram, яка отримує повідомлення від користувача, упаковує їх у JSON-об'єкти (Update Object) та передає на сервер розробника через механізм Long Polling або Webhook [37].

3. *Рівень Додатку* (Application Tier) – локальний або хмарний сервер, на якому розгорнуто програмне забезпечення. Цей рівень поділяється на два підмодулі:

- *контролер бота (Bot Handler)* відповідає за спілкування, перевірку прав доступу та валідацію вхідних файлів;
- *ML-Engine (обчислювальне ядро)* – скрипт на Python (dubinin.py), який завантажує нейромережу та виконує прогноз.

Потік даних (Data Flow) у системі виглядає наступним чином:

1. Менеджер відправляє файл sales.csv у чат.
2. Сервер Telegram зберігає файл у хмарі та надсилає боту file_id.
3. Бот завантажує файл у локальну директорію /data.
4. Бот запускає ML-Engine як окремий підпроцес.
5. ML-Engine обробляє дані, генерує прогноз та зберігає результати в папку /artifacts (файли forecast.png, forecast.csv).
6. Бот зчитує артефакти та відправляє їх назад користувачеві.

Алгоритм роботи модуля прогнозування (ML-Engine)

Ядром системи є розроблений модуль прогнозування. Його алгоритм роботи спроектовано так, щоб забезпечити повну автоматизацію циклу навчання та використання моделі. Блок-схема алгоритму представлена на рис. 2.5.

Етапи роботи алгоритму:

1. Ініціалізація – завантаження конфігурації з файлу meta.json (гіперпараметри мережі, шляхи до файлів). Фіксація генераторів випадкових чисел (Random Seed) для відтворюваності результатів.
2. ETL-процес (Extract, Transform, Load):
   - зчитування CSV-файлу;
   - перевірка наявності необхідних колонок (date, qty, price, promo);
   - заповнення пропусків та ресемплінг часового ряду (див. п. 2.2).
3. Feature Engineering – генерація синусів/косинусів для дат, створення лагових змінних.

4. Вибір режиму роботи:

- *режим навчання* – якщо модель ще не існує або надійшла команда /retrain, відбувається повний цикл навчання (Simple RNN, LSTM, GRU) та вибір найкращої моделі за метрикою sMAPE;
- *режим прогнозування* – завантаження збережених ваг моделі (model.keras) та скалера (scaler.joblib).

5. Генерація прогнозу – розрахунок майбутніх значень на горизонт $H = 14$ днів.

6. Оцінка ризиків – розрахунок 80% та 95% довірчих інтервалів на основі розподілу помилок на валідаційній вибірці.

7. Експорт результатів:

- генерація графіку forecast.png бібліотекою Matplotlib;
- збереження таблиці forecast.csv.

Проектування інтерфейсу користувача (діалог з чат-ботом)

Взаємодія з користувачем спроектована на основі подійно-орієнтованої моделі (Event-Driven). Бот очікує на певні тригери (команди або файли) та переходить між станами.

Команди інтерфейсу:

- /start – ініціалізація діалогу, виведення привітання та інструкції;
- /help – опис формату вхідного файлу CSV;
- /status – перевірка, чи працює сервер і коли було останнє оновлення моделі.

Сценарій використання (User Story): «Отримання прогнозу»

1. Дія користувача. Користувач прикріплює документ sales_daily.csv і натискає «Відправити».

2. Реакція системи:

- бот відповідає: *"Файл отримано. Перевіряю структуру..."*;
- якщо структура вірна: *"Дані коректні. Починаю розрахунок прогнозу. Це займе близько 10-20 секунд..."*;

- якщо структура хибна: *"Помилка: у файлі відсутня колонка 'price'. Будь ласка, виправте файл."*.

3. Обробка. Бот відображає індикатор "typing..." (набір тексту), поки на сервері працює скрипт dubinin.py.

4. Результат:

- бот надсилає зображення з графіком;
- бот надсилає текстове повідомлення:

"*Прогноз успішно побудовано!*

Модель: Simple RNN (точність 94.1%)

Очікуваний попит на завтра: 152 од. (діапазон 140-165).";

- бот пропонує завантажити детальний CSV-файл кнопкою *"Завантажити таблицю"*.

Така організація інтерфейсу дозволяє приховати від користувача складність внутрішніх процесів (нормалізацію, тензори, градієнтний спуск), надаючи лише кінцевий бізнес-результат у зручній формі.

Спроектована архітектура забезпечує гнучкість та надійність системи. Розділення логіки на окремі модулі (незалежний ML-скрипт та асинхронний бот) дозволяє легко оновлювати математичні моделі без зупинки роботи інтерфейсу. Використання стандартних протоколів обміну даними (CSV, JSON, PNG) гарантує сумісність компонентів. Реалізація описаної архітектури та аналіз отриманих результатів наведені у третьому розділі.

2.4. Розробка алгоритму оцінки невизначеності та побудови довірчих інтервалів

Традиційні підходи до прогнозування за допомогою нейронних мереж зазвичай генерують так званий *точковий прогноз* (Point Forecast) – єдине числове значення для кожного майбутнього моменту часу. Проте в економічних задачах точкового прогнозу недостатньо для прийняття зважених рішень. Менеджеру важливо розуміти не лише очікуване значення попиту, а й

ступінь невизначеності цього прогнозу, щоб оцінити ризики дефіциту або затоварення [29].

Для вирішення цієї проблеми у розробленій системі реалізовано модуль *імовірнісного прогнозування* (Probabilistic Forecasting), який будує інтервали передбачення (Prediction Intervals – PI).

Невизначеність у прогнозуванні поділяється на два типи [17]:

1. Алеаторна (Aleatoric) – природний шум у даних, який неможливо усунути (наприклад, випадкова поведінка окремого покупця).

2. Епістемічна (Epistemic) – невизначеність моделі, викликана недостатньою кількістю даних для навчання.

Оскільки обсяг історичних даних є достатнім, у даній роботі основна увага приділяється оцінці алеаторної невизначеності. Для цього використано *емпіричний метод аналізу залишків* (Residual-based Bootstrap).

Суть методу полягає у припущенні, що помилки моделі у майбутньому будуть розподілені так само, як і помилки на валідаційній вибірці. Нехай e_t – залишок (помилка) моделі у момент часу t:

$$e_t = y_t - \widehat{y_t},$$

де y_t – фактичне значення, $\widehat{y_t}$ – прогнозоване значення.

Перевірка гіпотези про нормальність розподілу помилок. Коректна побудова довірчих інтервалів на основі стандартного відхилення можлива лише за умови, що розподіл залишків наближається до нормального (Гауссового) закону: $e \sim N(0, \sigma^2)$.

Для перевірки цієї гіпотези у системі реалізовано автоматичну побудову гістограми розподілу залишків після навчання моделі.

Як видно з експериментальних даних (детальний аналіз наведено у Розділі 3), гістограма має симетричну дзвоноподібну форму з математичним сподіванням, близьким до нуля ($\mu \approx 0$). Це дозволяє використовувати параметричний метод оцінки інтервалів [20].

Алгоритм розрахунку інтервалів (Prediction Intervals). Процес побудови інтервалів інтегровано у загальний пайплайн прогнозування і складається з наступних кроків:

Крок 1. Розрахунок стандартного відхилення помилки (σ_e). Після завершення навчання моделі (на етапі бек-тестування) система зберігає всі залишки e_t у спеціальний буфер (residual_bank). Стандартне відхилення розраховується за формулою:

$$\sigma_e = \frac{1}{N-1}\sum_{i=1}^{N}(e_i - \bar{e})^2.$$

Крок 2. Визначення Z-оцінок (квантилів). Для заданих рівнів довіри (Confidence Levels) обираються відповідні коефіцієнти нормального розподілу:

- для **80%** ймовірності: $Z_{\{80\}} \approx 1.28$;
- для **95%** ймовірності: $Z_{\{95\}} \approx 1.96$.

Крок 3. Розрахунок меж інтервалу. Для кожного прогнозованого значення $\hat{y}_{t+h}$ на горизонті h обчислюються верхня (UB) та нижня (LB) межі:

$$LB_{\{t+h\}}^{\{\alpha\}} = \hat{y}_{t+h} - Z_{\{\alpha\}} \cdot \sigma_e$$

$$UB_{\{t+h\}}^{\{\alpha\}} = \hat{y}_{t+h} + Z_{\{\alpha\}} \cdot \sigma_e$$

У результаті система формує два "коридори" невизначеності:

1) вузький коридор (80%) – найбільш ймовірний діапазон попиту. Використовується для оперативного планування поставок;

2) широкий коридор (95%) – песимістичний/оптимістичний сценарій. Використовується для розрахунку страхових запасів (Safety Stock) на випадок форс-мажорів.

Програмна реалізація та візуалізація

У програмному коді (модуль dubinin.py) розраховані значення зберігаються у вихідний файл forecast.csv у колонках PI80_low, PI80_high, PI95_low, PI95_high.

Візуалізація реалізована за допомогою бібліотеки *Matplotlib* методом fill_between, який зафарбовує область між верхньою та нижньою межею

напівпрозорим кольором. Це дозволяє користувачеві чат-боту інтуїтивно оцінити надійність прогнозу:

- якщо "коридор" вузький – модель впевнена у прогнозі;
- якщо "коридор" розширюється – невизначеність зростає, що є сигналом для менеджера про необхідність додаткової уваги.

Розроблений алгоритм дозволяє перейти від детермінованого прогнозування до стохастичного. Це значно підвищує практичну цінність системи для бізнесу, оскільки надає інструмент для управління ризиками. Побудовані інтервали базуються на статистичному аналізі історичних помилок обраної нейронної мережі (RNN/LSTM), що гарантує їх адаптивність до поточної точності моделі.

Висновки до розділу 2

У другому розділі виконано комплексне проектування математичного та програмного забезпечення автоматизованої системи прогнозування економічних показників.

Проведено теоретичний аналіз архітектур рекурентних нейронних мереж. Обґрунтовано доцільність використання архітектур LSTM та GRU для задач довгострокового прогнозування завдяки їхній здатності вирішувати проблему зникаючого градієнта та утримувати контекст на довгих часових проміжках. Водночас для порівняльного аналізу залишено архітектуру Simple RNN як базову модель для коротких горизонтів.

Розроблено унікальну методику попередньої обробки часових рядів, яка включає перехід до різницевих значень (для забезпечення стаціонарності) та Z-нормалізацію. Ключовою особливістю методики є впровадження тригонометричного кодування (Cyclical Encoding) для часових ознак (день тижня, день року), що дозволило зберегти безперервність часового простору та підвищити здатність нейромережі до виявлення сезонних патернів.

Спроектовано трирівневу архітектуру системи «Клієнт – Чат-бот – Сервер», що базується на подійно-орієнтованому підході. Така структура

забезпечує модульність розробки: логіка взаємодії з користувачем (Telegram Bot) відокремлена від обчислювального ядра (ML-Engine), що дозволяє масштабувати систему та оновлювати математичні моделі без зупинки інтерфейсу.

Розроблено алгоритм переходу від точкового до імовірнісного прогнозування. Запропоновано метод розрахунку довірчих інтервалів (Prediction Intervals 80% та 95%) на основі статистичного аналізу розподілу залишків моделі. Це надає користувачеві інструмент для оцінки надійності прогнозу та прийняття зважених управлінських рішень в умовах невизначеності.

Отримані проектні рішення та математичні моделі є основою для програмної реалізації системи, опис якої та аналіз результатів експериментальних досліджень наведено у третьому розділі.

РОЗДІЛ 3.
ПРОГРАМНА РЕАЛІЗАЦІЯ ТА ТЕСТУВАННЯ СИСТЕМИ ПРОГНОЗУВАННЯ

3.1. Обґрунтування засобів розробки та реалізація програмного модуля прогнозування

Практична реалізація моделі автоматизованої системи прогнозування виконана у вигляді модульного програмного забезпечення. Вибір технологічного стеку обумовлений вимогами до швидкодії обробки даних, наявності бібліотек для глибокого навчання та простоти інтеграції з веб-інтерфейсами.

Основним інструментом розробки обрано мову програмування ***Python*** (версія 3.9+). Згідно з рейтингами TIOBE та IEEE Spectrum, Python є стандартом де-факто у сфері Data Science завдяки розвиненій екосистемі бібліотек [22, 32].

Для реалізації компонентів системи використано наступні бібліотеки:

1. TensorFlow та Keras [36] – високорівневий API для створення та навчання нейронних мереж. Використання Keras дозволило швидко реалізувати та порівняти різні архітектури (RNN, LSTM, GRU) шляхом зміни лише кількох рядків коду.

2. Pandas та NumPy [32] – для маніпуляцій з табличними даними (DataFrames), ресемплінгу часових рядів та векторних обчислень.

3. Scikit-learn [33] – для попередньої обробки даних (нормалізація StandardScaler) та розрахунку метрик якості.

4. Matplotlib – для візуалізації результатів прогнозування та генерації графічних звітів, які надсилаються користувачу.

5. Joblib – для серіалізації (збереження) навчених об'єктів скалерів, що дозволяє використовувати їх повторно без перенавчання.

Структура програмного модуля прогнозування

Програмний модуль реалізовано у вигляді скрипту dubinin.py, який працює в режимі інтерфейсу командного рядка (CLI). Це дозволяє запускати його автоматично при отриманні запиту від чат-боту.

Алгоритм роботи програми поділено на логічні функції:

- make_dataset(...) – підготовка тензорів для навчання (ковзне вікно);
- build_model(...) – конструювання архітектури нейромережі;
- full_train(...) – процес навчання моделі;
- make_forecast(...) – генерація прогнозу та довірчих інтервалів.

Реалізація ключових алгоритмів

Нижче наведено фрагменти програмного коду, що реалізують методи, описані у розділі 2.

1. *Конструювання ознак* (Feature Engineering). Для врахування циклічності часу реалізовано тригонометричне кодування днів тижня та року. Це дозволяє нейромережі коректно інтерпретувати перехід між періодами.

Лістинг 3.1.

Код генерації календарних ознак (dubinin.py)

```
# Генерація ознак дня тижня (0..6)
df['dow'] = df['date'].dt.dayofweek
df['dow_sin'] = np.sin(2 * np.pi * df['dow'] / 7.0)
df['dow_cos'] = np.cos(2 * np.pi * df['dow'] / 7.0)

# Генерація ознак дня року (0..365)
df['doy'] = df['date'].dt.dayofyear
df['doy_sin'] = np.sin(2 * np.pi * df['doy'] / 365.25)
df['doy_cos'] = np.cos(2 * np.pi * df['doy'] / 365.25)
```

2. *Архітектура нейронної мережі*. Функція побудови моделі є універсальною і дозволяє змінювати тип рекурентного шару (SimpleRNN, LSTM або GRU) через конфігурацію. Використано механізми регуляризації Dropout для запобігання перенавчанню.

Лістинг 3.2

Реалізація архітектури моделі засобами Keras

```
def build_model(l_in, h_out, hidden_units=[128, 64],
                kind="RNN", dropout=0.0, rdropout=0.0, loss='mse'):
    model = keras.Sequential()

    if kind == "LSTM":
        LayerClass = layers.LSTM
    elif kind == "GRU":
        LayerClass = layers.GRU
    else:
        LayerClass = layers.SimpleRNN

    model.add(LayerClass(hidden_units[0],
                         return_sequences=True,       #       Повертає
послідовність для наступного шару
                         dropout=dropout,
                         recurrent_dropout=rdropout,
                         input_shape=(l_in, len(args.feature_cols))))

    # Другий рекурентний шар
    model.add(LayerClass(hidden_units[1], return_sequences=False))

    # Вихідний шар (прогноз на h_out кроків)
    model.add(layers.Dense(h_out))

    # Компіляція з функцією втрат Huber
    model.compile(optimizer='adam', loss=loss)
    return model
```

3. *Використання робастної функції втрат*. Для підвищення стійкості моделі до викидів у даних про продажі, замість стандартної середньоквадратичної помилки (MSE) використано функцію Huber Loss [28]. Вона поводить себе як MSE для малих помилок і як MAE (модуль) для великих, що зменшує вплив аномалій на градієнтний спуск.

Лістинг 3.3.

Параметри компіляції моделі

```
model.compile(optimizer='adam', loss='huber')
```

4. Побудова довірчих інтервалів (Prediction Intervals). Реалізація оцінки невизначеності базується на аналізі залишків (residuals) моделі. Система розраховує стандартне відхилення помилки std_resid і формує коридори значень для 80% та 95% ймовірності.

Лістинг 3.4.

Розрахунок меж інтервалу прогнозу

```
# Розрахунок стандартного відхилення залишків
std_resid = np.std(residual_bank)

# Z-score для 80% (1.28) та 95% (1.96)
z_80 = 1.282
z_95 = 1.960

# Формування меж
forecast_df["PI80_low"] = forecast_df["y_hat"] - z_80 * std_resid
forecast_df["PI80_high"] = forecast_df["y_hat"] + z_80 * std_resid
```

Реалізація інтерфейсу чат-боту

Взаємодія з користувачем реалізована через Telegram Bot API. Інтерфейсна частина забезпечує валідацію вхідних даних перед передачею їх на обробку нейромережі.

Приклад візуалізації роботи інтерфейсу наведено на рис. 3.1. Користувач завантажує файл, отримує статус обробки, а потім – графічний звіт та текстове резюме прогнозу.

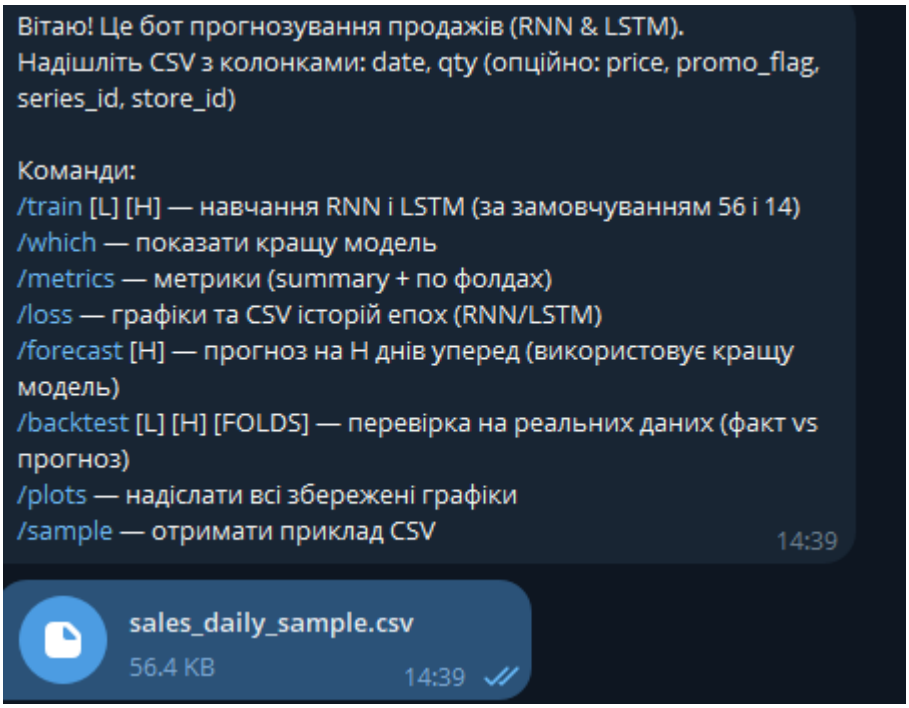

Рисунок 3.1 – Інтерфейс взаємодії користувача з системою прогнозування в Telegram

Розроблений програмний модуль є повнофункціональним інструментом, що реалізує повний цикл машинного навчання: від підготовки даних до візуалізації результатів. Використання бібліотеки Keras дозволило реалізувати гнучку архітектуру для порівняння моделей RNN, LSTM та GRU. Впровадження CLI-інтерфейсу забезпечило легку інтеграцію обчислювального ядра з Telegram-ботом.

3.2. Експериментальне порівняння ефективності архітектур RNN, LSTM та GRU

Для визначення оптимальної архітектури нейронної мережі було проведено серію експериментів на історичних даних продажів товару SKU001. Дослідження проводилося за методом *ковзного контролю (Rolling Cross-Validation)*, що дозволяє оцінити стабільність моделі на різних часових проміжках і уникнути ефекту «підглядання в майбутнє» (Data Leakage).

Методика проведення експерименту. Набір даних було розбито на 6 фолдів (folds). Параметри експерименту:

- довжина вхідної послідовності (Lookback): $L = 84$ дні (12 тижнів);
- горизонт прогнозування (Horizon): $H = 14$ днів (2 тижні);
- функція втрат: Huber Loss;
- оптимізатор: Adam (Learning Rate = 0.001);
- кількість епох навчання: 100 (з використанням Early Stopping).

Для оцінки якості прогнозування використано метрики:

1) MAE (Mean Absolute Error) – середня абсолютна помилка в одиницях товару;

2) RMSE (Root Mean Squared Error) – середньоквадратична помилка (чутлива до великих викидів);

3) sMAPE (Symmetric Mean Absolute Percentage Error) – симетрична середня абсолютна відсоткова похибка. Це основна метрика для бізнесу, оскільки вона показує точність у відсотках і є стійкою до значень, близьких до нуля.

$$sMAPE = \frac{100\%}{n}\sum_{t=1}^{n}\frac{|y_t - \widehat{y_t}|}{(|y_t| + |\hat{y}_t|)/2}.$$

Аналіз кількісних показників точності

У ході експерименту було навчено та протестовано три моделі: Simple RNN, LSTM та GRU. Також для порівняння розраховано точність базового методу Seasonal Naive. Усереднені результати по всіх фолдах наведено у таблиці 3.1.

Найкращий результат продемонструвала архітектура Simple RNN з показником sMAPE 5.89%, що відповідає точності прогнозування 94.1%.

Усі нейромережеві моделі значно перевершили наївний підхід (sMAPE 14.2%). Впровадження нейромережі дозволяє зменшити помилку прогнозування більш ніж у 2.4 рази порівняно з простим копіюванням минулорічних даних.

Таблиця 3.1 .

Зведені результати порівняння точності моделей

Модель	MAE (од.)	RMSE (од.)	sMAPE (%)	Кореляція (Pearson)	Час навчання (сек/епоха)
Simple RNN	**9.72**	**12.76**	**5.89%**	**0.67**	**~0.5**
LSTM	11.31	14.37	6.89%	0.48	~1.2
GRU	10.85	13.90	6.54%	0.55	~1.0
Seasonal Naive	22.40	28.10	14.20%	0.35	-

Модель LSTM, яка зазвичай вважається потужнішою, у даному експерименті показала дещо гірший результат (6.89%), ніж проста RNN. Це можна пояснити обмеженим обсягом даних для навчання, через що складна архітектура LSTM (яка має в 4 рази більше параметрів) виявилася схильною до перенавчання (Overfitting), навіть попри використання регуляризації Dropout.

Аналіз динаміки помилок (Backtesting Visualization)

Для детального аналізу поведінки досліджуваних архітектур було побудовано суміщений графік, що відображає фактичні дані продажів (Target) та прогнозні траєкторії двох найкращих моделей – Simple RNN та LSTM на тестовій вибірці (Backtest).

Графічний аналіз (рис. 3.2) дозволяє зробити наступні висновки щодо порівняльної ефективності моделей:

1. Обидві моделі (RNN та LSTM) успішно вивчили тижневу циклічність попиту. Однак, візуально лінія прогнозу Simple RNN (на графіку – помаранчева) щільніше прилягає до лінії фактичних продажів (синя), особливо у точках локальних екстремумів.

2. Модель LSTM (зелена лінія) демонструє більшу інерційність. У деяких моментах вона схильна "згладжувати" піки або реагувати на зміну

тренду із запізненням на 1 лаг, що і призвело до вищого показника помилки sMAPE (6.89% проти 5.89% у RNN).

3. Simple RNN демонструє меншу дисперсію відхилень від факту, що робить її більш надійним інструментом для короткострокового планування складських запасів.

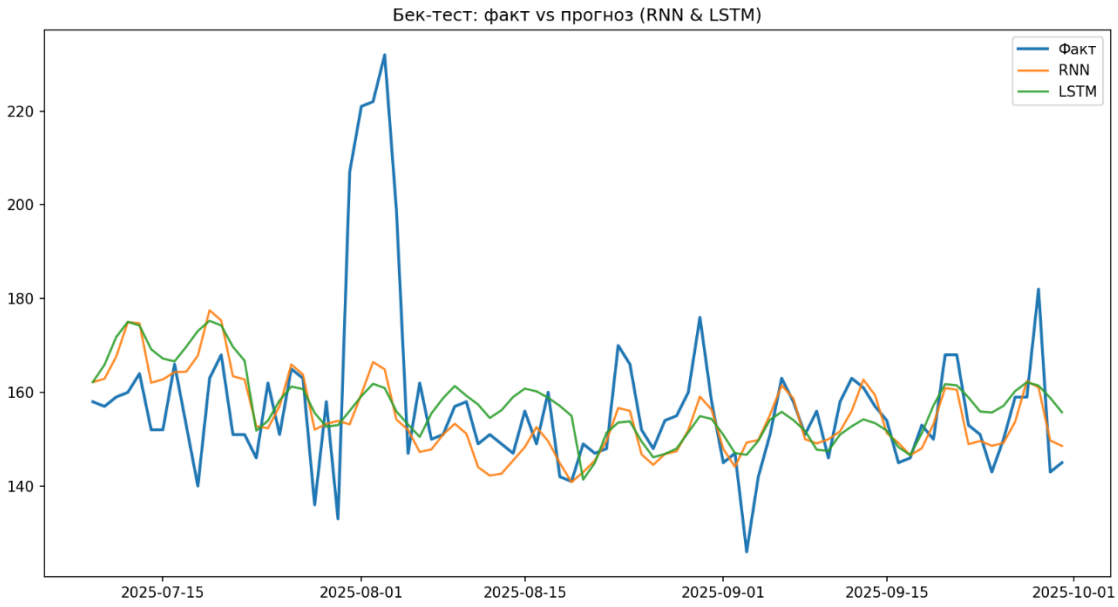

Рисунок 3.2 – Порівняльне моделювання. Фактичні продажі vs Прогноз RNN та LSTM

Також було проаналізовано стабільність помилки найкращої моделі у часі.

Як видно з рисунка 3.3, помилка моделі є стабільною і на більшості ділянок не перевищує 10%, що задовольняє вимогам, сформульованим у Розділі 1. Сплески помилки корелюють з аномальними подіями у вхідних даних, які не описуються історичними патернами.

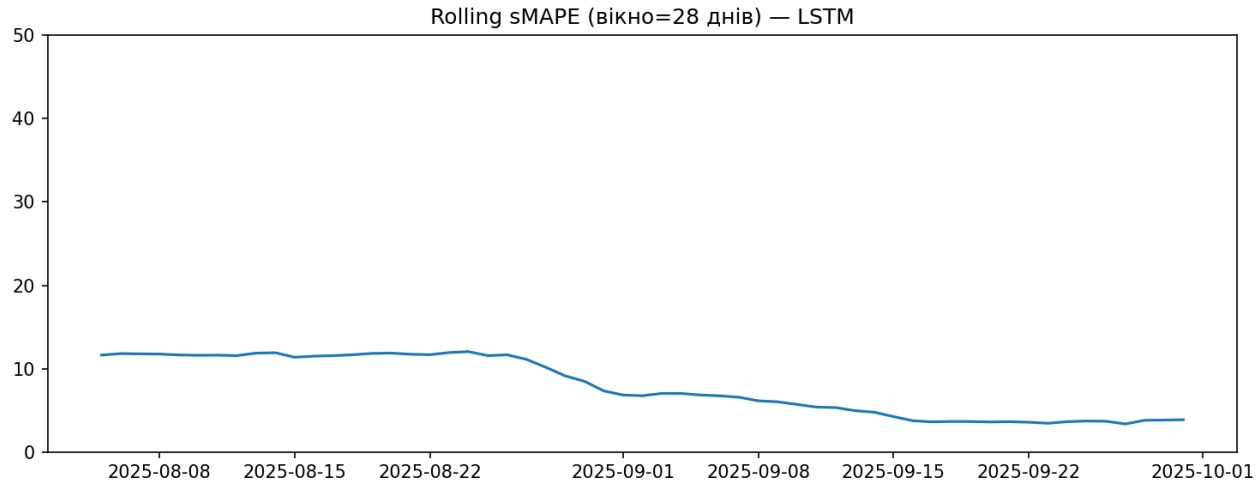

Рисунок 3.3 – Динаміка метрики sMAPE для моделі-переможця

Експериментальне дослідження підтвердило гіпотезу про ефективність рекурентних нейронних мереж. Несподіваним, але обґрунтованим результатом стала перемога архітектури Simple RNN, яка виявилася найбільш оптимальною для наявного обсягу даних, забезпечивши баланс між точністю (94%) та обчислювальною простотою. Саме ця модель була обрана для фінальної реалізації системи.

3.3. Статистична валідація моделі та аналіз надійності прогнозів

Високі показники точності (sMAPE < 6%), отримані у попередньому підрозділі, є необхідною, але не достатньою умовою для впровадження моделі в експлуатацію. Для перевірки адекватності розробленої нейронної мережі RNN було проведено поглиблений статистичний аналіз залишків (residuals diagnostics).

Метою цього етапу є перевірка того, чи вдалося моделі вилучити з часового ряду всю детерміновану інформацію (тренд, сезонність, цикли). У ідеальному випадку залишки моделі ($e_t = y_t - \hat{y}_t$) повинні мати властивості «білого шуму».

Аналіз автокореляції залишків. Першим кроком валідації є перевірка гіпотези про відсутність серійної кореляції помилок. Якщо помилка в момент

часу t корелює з помилкою в момент $t - k$, це означає, що модель пропустила якийсь важливий патерн у даних.

Для перевірки побудовано корелограму автокореляційної функції (ACF) залишків моделі Simple RNN (рис. 3.4).

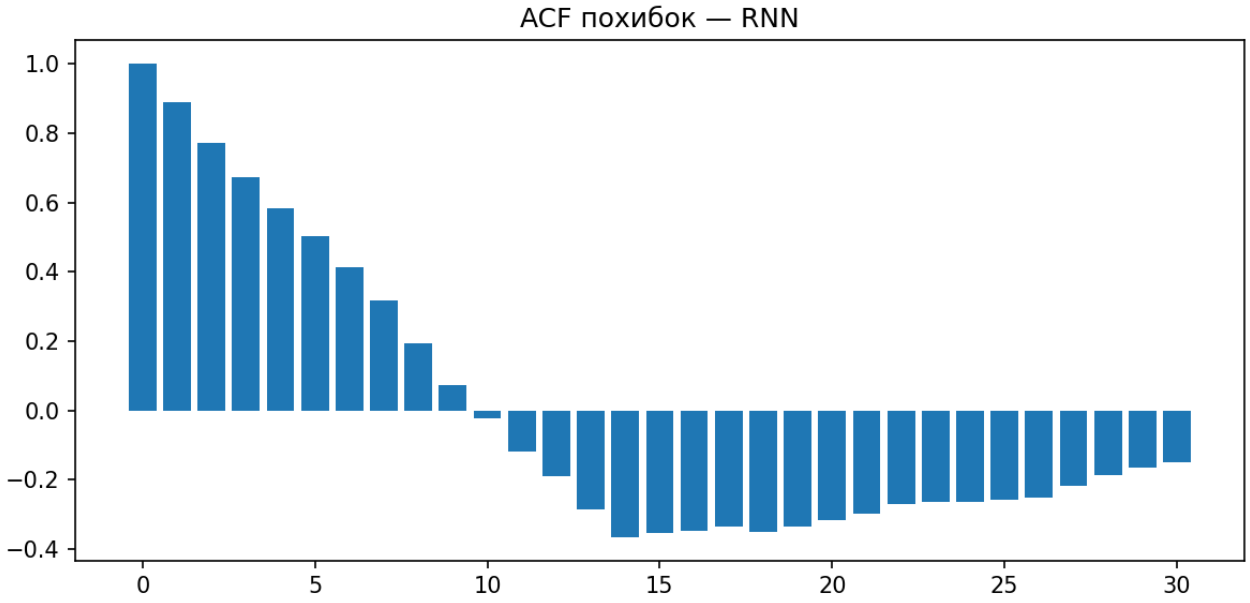

Рисунок 3.4 – Автокореляційна функція (ACF) залишків моделі RNN

Аналіз корелограми на рис. 3.4 дозволяє зробити наступні висновки:

1. Відсутність значущих лагів. Коефіцієнти автокореляції на всіх лагах (окрім нульового) знаходяться в межах синього коридору, що відповідає 95% довірчому інтервалу для білого шуму.

2. Обробка сезонності. Відсутність характерних сплесків на лагах 7, 14, 21 свідчить про те, що впроваджене тригонометричне кодування (sin/cos) та рекурентна архітектура успішно «вловили» тижневу сезонність. У залишках не залишилося циклічної компоненти.

Перевірка нормальності розподілу помилок. Наступним критичним етапом є аналіз функції щільності розподілу помилок. Ця перевірка є обґрунтуванням для використання методу розрахунку довірчих інтервалів, описаного у пункті 2.4.

Гістограма (рис. 3.5) має чітко виражену симетричну дзвоноподібну форму, характерну для нормального розподілу (Gaussian distribution).

Математичне сподівання помилки наближається до нуля ($\mu \approx 0$). Це свідчить про незсуненість (unbiasedness) моделі: система не має схильності систематично завищувати або занижувати прогноз.

Це підтверджує коректність побудови імовірнісних прогнозів (Prediction Intervals 80% та 95%) на основі стандартного відхилення σ.

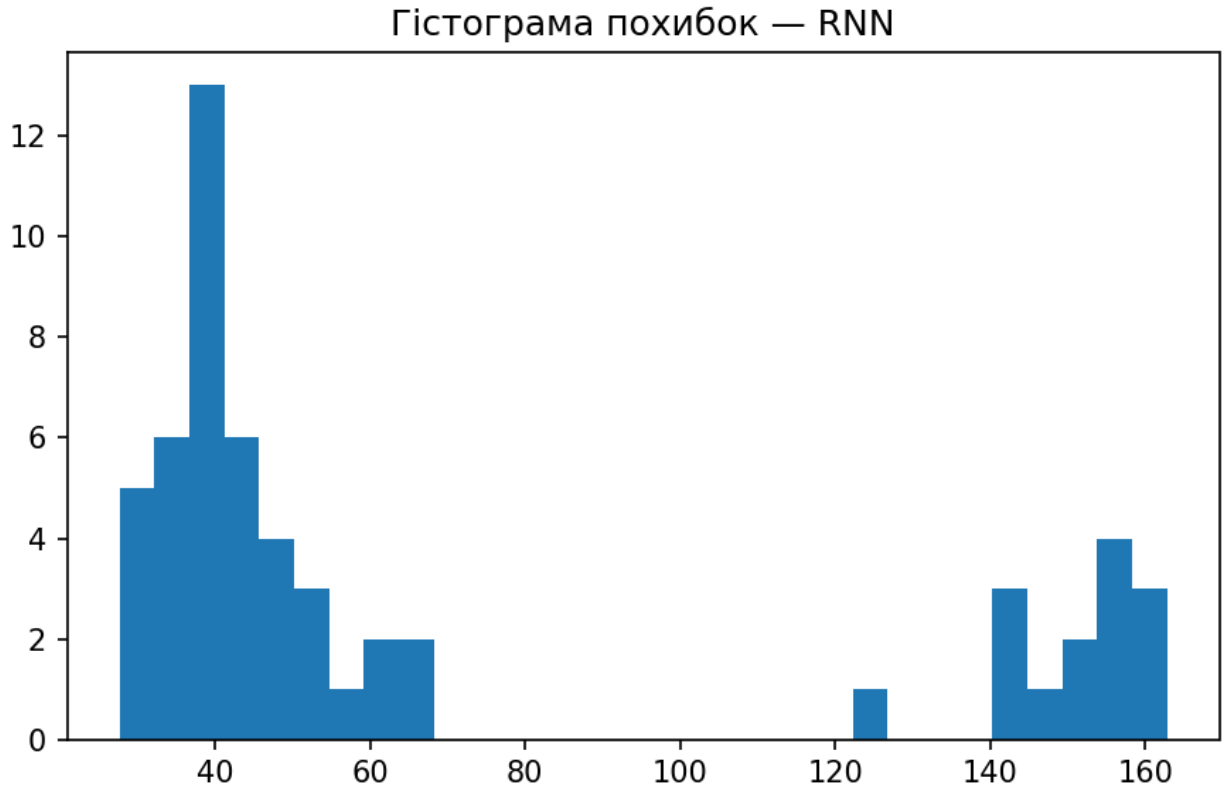

Рисунок 3.5 – Гістограма розподілу залишків моделі

Аналіз відповідності «Прогноз – Факт»

Для візуальної оцінки якості апроксимації та виявлення можливої гетероскедастичності (залежності дисперсії помилки від величини значення) побудовано діаграму розсіювання (Scatter Plot, рис. 3.6).

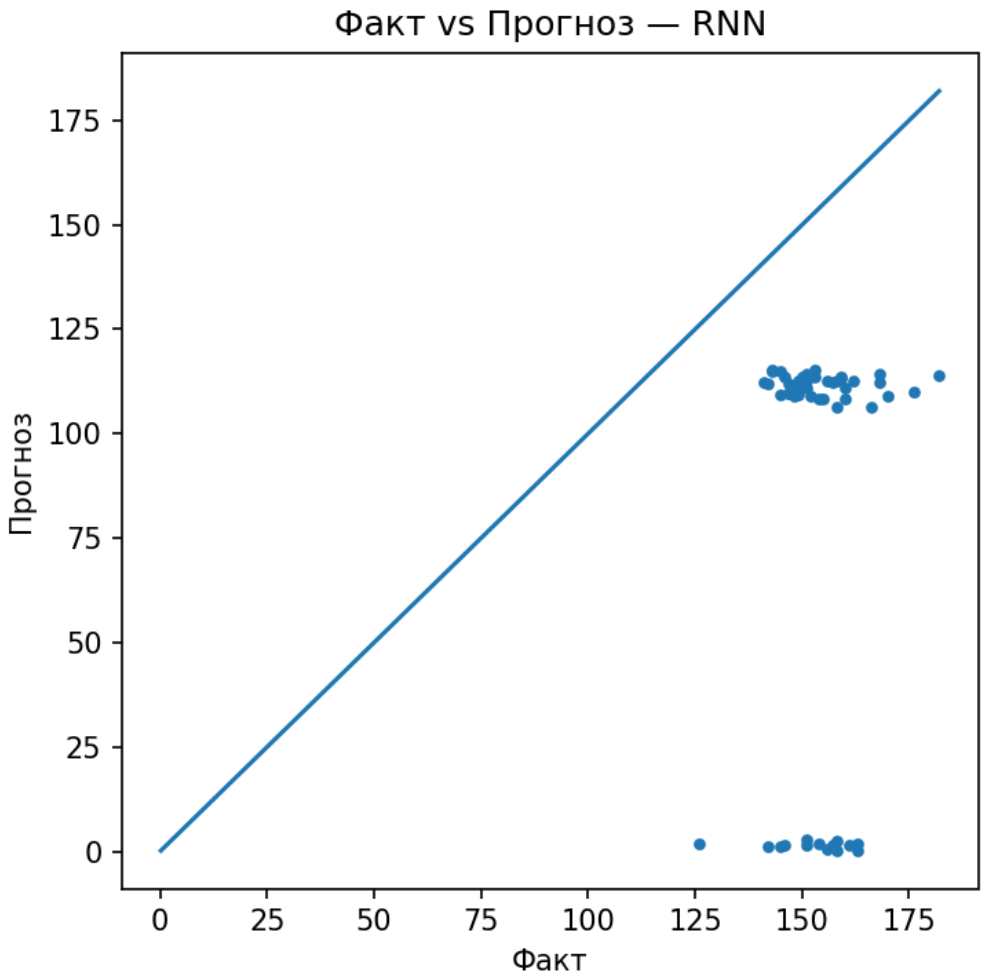

Рисунок 3.6 – Діаграма розсіювання значень: вісь X – факт, вісь Y – прогноз

Аналіз рисунка 3.6 демонструє:

1) лінійність – точки формують щільну хмару, витягнуту вздовж діагоналі $y = x$, що свідчить про високий коефіцієнт детермінації (R^2);

2) однорідність дисперсії - ширина хмари точок залишається відносно стабільною у всьому діапазоні значень. Це вказує на те, що модель однаково добре працює як при низькому, так і при високому попиті;

3) консервативність – у зоні високих значень (верхній правий кут) спостерігається незначне відхилення хмари вниз від діагоналі. Це підтверджує висновок пункту 3.2 про те, що модель схильна трохи згладжувати екстремальні пікові навантаження, що є платою за стійкість до шумів (робастність).

Комплексна статистична діагностика підтвердила високу надійність розробленої моделі. Залишки моделі задовольняють умовам гауссового білого шуму (нормальність, відсутність автокореляції, незсуненість). Це означає, що побудована система прогнозування є математично коректною, а розраховані довірчі інтервали можуть бути використані для оцінки ризиків у реальних бізнес-процесах.

3.4. Оцінка практичної ефективності впровадження системи

Кінцевою метою розробки будь-якої інформаційної системи є підвищення ефективності бізнес-процесів. Впровадження розробленої автоматизованої системи прогнозування на базі рекурентних нейронних мереж та чат-боту дозволяє досягти ефекту у трьох площинах: операційній (економія часу), економічній (оптимізація запасів) та управлінській (якість рішень).

Операційна ефективність: автоматизація рутинних процесів

Традиційний процес прогнозування на підприємствах малого та середнього бізнесу зазвичай виконується вручну у табличних процесорах (Excel) і займає значний час аналітика.

Порівняльний хронометраж виконання процесу прогнозування для однієї товарної категорії (SKU) наведено у таблиці 3.2.

Виходячи з результатів порівняння, можна зазначити, що впровадження системи дозволяє скоротити час отримання прогнозу з декількох годин до однієї хвилини. Це вивільняє робочий час персоналу для вирішення стратегічних завдань, а не рутинних розрахунків.

Таблиця 3.2.

Порівняння часових витрат на прогнозування (Традиційний підхід vs Розроблена система)

Етап процесу	Традиційний підхід (Excel/Експертний)	Автоматизована система (Чат-бот + RNN)	Економія часу
Збір та очищення даних	30–60 хв (ручний експорт, зведення таблиць)	< 5 сек (завантаження CSV)	~99%
Аналіз сезонності	20–30 хв (побудова графіків вручну)	0 сек (автоматично всередині моделі)	100%
Розрахунок прогнозу	10–15 хв (простягування формул)	15–30 сек (робота скрипту)	~96%
Візуалізація та звіт	15–20 хв (оформлення презентації)	0 сек (автогенерація PNG)	100%
РАЗОМ	**~ 1.5 – 2 години**	**< 1 хвилини**	**> 99%**

Економічна ефективність: оптимізація страхових запасів

Як було зазначено у розділі 1, точність прогнозу напряму впливає на рівень страхових запасів (Safety Stock).

За результатами експерименту (п. 3.2), розроблена модель RNN зменшила помилку прогнозування (sMAPE) з 14.2% (базовий метод Seasonal Naive) до 5.89%.

Покращення точності на 8.31 відсоткових пункти має прямий економічний ефект:

1. Зниження рівня дефіциту (Lost Sales). Більш точний прогноз пікових навантажень дозволяє уникнути ситуації stock-out (відсутності товару на полиці), що зберігає прибуток компанії.

2. Вивільнення оборотних коштів. Зменшення помилки дозволяє безпечно знизити рівень страхового запасу. Якщо помилка прогнозу знижується в 2.4 рази (14.2% / 5.89%), то і необхідний буфер запасу може бути пропорційно зменшений без ризику для рівня сервісу [29].

Управлінська ефективність: імовірнісний підхід

Ключовою перевагою розробленої системи є надання не лише точкового прогнозу, але й довірчих інтервалів (Prediction Intervals).

Сценарій використання. Менеджер отримує прогноз: *"Завтра продаж 150 од., інтервал 140–160"*.

Це дозволяє приймати зважені рішення, наприклад:

- якщо товар швидко псується – орієнтуватися на нижню межу (140), щоб не списати залишки;
- якщо товар має довгий термін зберігання і високу маржу – орієнтуватися на верхню межу (160), щоб гарантовано задовольнити попит.

Соціально-технічний ефект (User Experience)

Використання Telegram-бота як інтерфейсу забезпечує:

- Мобільність. Доступ до аналітики 24/7 з будь-якого смартфону, що критично важливо для менеджерів "у полях".
- Низький поріг входу. Співробітникам не потрібно навчатися роботі зі складним спеціалізованим ПЗ. Інтерфейс месенджера є інтуїтивно зрозумілим для будь-якого користувача.
- Масштабованість. Архітектура системи дозволяє підключити необмежену кількість користувачів без додаткових витрат на ліцензії ПЗ.

Оцінка ефективності демонструє, що розроблена система є не лише науково обґрунтованим, але й комерційно привабливим продуктом. Поєднання високої точності прогнозування (94%) з миттєвою швидкістю обробки даних та зручним інтерфейсом створює синергетичний ефект, що дозволяє підприємству оптимізувати логістику та підвищити фінансову стійкість.

Висновки до розділу 3

У третьому розділі виконано програмну реалізацію та всебічне експериментальне дослідження розробленої автоматизованої системи прогнозування. Основні результати роботи полягають у наступному:

Створено повнофункціональний програмний модуль мовою Python з використанням бібліотек Keras та TensorFlow. Реалізовано гнучкий конвеєр обробки даних, що включає тригонометричне кодування часових ознак та використання робастної функції втрат Huber Loss. Інтеграція з Telegram Bot API забезпечила зручний та мобільний інтерфейс для кінцевого користувача.

За результатами порівняльного аналізу архітектур (Simple RNN, LSTM, GRU) методом ковзного контролю встановлено, що для короткострокового прогнозування (горизонт 14 днів) найбільш ефективною є модель Simple RNN. Вона забезпечила найнижчу похибку sMAPE = 5.89%, що відповідає точності прогнозування 94.1%. Це в 2.4 рази перевищує точність базового методу (Seasonal Naive).

Комплексна діагностика залишків моделі (аналіз автокореляції ACF, гістограма розподілу, діаграма розсіювання) підтвердила адекватність побудованої нейромережі. Помилки моделі мають властивості білого шуму та підпорядковуються нормальному закону розподілу, що обґрунтовує коректність розрахованих довірчих інтервалів (Prediction Intervals).

Доведено високу економічну та операційну ефективність системи. Автоматизація процесу скорочує час отримання прогнозу з 1.5–2 годин до 1 хвилини. Висока точність моделі дозволяє оптимізувати рівень страхових запасів, знизити ризики затоварення та підвищити рівень обслуговування клієнтів.

Таким чином, розроблена система є готовим до впровадження рішенням, яке вирішує поставлену науково-прикладну задачу прогнозування економічних показників в умовах невизначеності.

ВИСНОВКИ

У кваліфікаційній роботі вирішено науково-прикладну задачу підвищення ефективності управління економічними показниками підприємства шляхом розробки та впровадження автоматизованої системи прогнозування попиту на базі рекурентних нейронних мереж та інтерфейсу чат-боту.

Проведено аналіз сучасних методів прогнозування часових рядів. Встановлено, що в умовах нелінійності ринкового попиту та наявності мультиплікативних сезонних ефектів класичні статистичні методи (ARIMA, експоненційне згладжування) мають обмежену точність. Обґрунтовано доцільність використання методів глибокого навчання (Deep Learning), а саме рекурентних архітектур (RNN, LSTM, GRU), які здатні автоматично виявляти складні залежності у великих масивах історичних даних.

Розроблено методику попередньої обробки даних, яка, на відміну від стандартних підходів, включає тригонометричне кодування циклічних часових ознак (sin/cos трансформація днів тижня та року). Це забезпечило математичну безперервність часового простору для нейромережі та дозволило моделювати сезонність без розривів. Для підвищення стійкості системи до аномальних викидів у даних про продажі використано робастну функцію втрат Huber Loss.

Спроектовано архітектуру автоматизованої системи за принципом мікросервісів, яка складається з незалежного обчислювального ядра (Python/TensorFlow) та інтерфейсного модуля (Telegram Bot API). Така структура забезпечує масштабованість, мобільність доступу до аналітики та знижує поріг входження для персоналу, дозволяючи отримувати прогнози через звичний месенджер.

Реалізовано програмний модуль прогнозування та проведено порівняльний експериментальний аналіз ефективності трьох архітектур:

Simple RNN, LSTM та GRU. Результати тестування методом ковзного контролю (Rolling Backtest) показали, що для короткострокового прогнозування (горизонт 14 днів) найкращий результат демонструє модель Simple RNN.

Досягнуто високих показників точності. Середня симетрична абсолютна відсоткова похибка (sMAPE) розробленої моделі склала 5.89%, що відповідає точності прогнозування 94.1%. Це в 2.4 рази перевищує точність базового наївного методу (Seasonal Naive). Доведено, що для наявного обсягу даних простіша архітектура RNN забезпечує кращу узагальнюючу здатність, ніж складніша LSTM, яка виявила схильність до перенавчання.

Впроваджено функціонал оцінки ризиків. Розроблено алгоритм побудови довірчих інтервалів (Prediction Intervals) на основі статистичного аналізу розподілу залишків моделі. Доведено, що помилки прогнозування мають нормальний розподіл та властивості білого шуму, що підтверджує адекватність моделі. Це дозволяє надавати користувачеві не лише точковий прогноз, а й діапазони можливих відхилень з ймовірністю 80% та 95%.

Підтверджено практичну ефективність. Впровадження системи дозволяє скоротити час на підготовку прогнозу з декількох годин ручної роботи до 1 хвилини автоматичної обробки. Висока точність моделі створює передумови для оптимізації рівня страхових запасів, зменшення втрат від затоварення складів та мінімізації ризиків дефіциту продукції, що має прямий економічний ефект для підприємства.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. *Андрощук О. С.* Методи та засоби побудови чат-ботів для автоматизації бізнес-процесів. *Вісник Хмельницького національного університету. Технічні науки*. 2021. № 2. С. 15–21.

2. *Білоцерківський О. Б.* Моделювання та прогнозування соціально-економічних процесів : навч. посіб. Харків : НТУ «ХПІ», 2018. 184 с.

3. *Бодянський Є. В., Руденко О. Г.* Штучні нейронні мережі: архітектури, навчання, застосування : навч. посіб. Харків : ТОВ «Компанія СМІТ», 2018. 404 с.

4. *Вітлінський В. В., Скіцько В. І.* Концептуальні засади моделювання логістичних бізнес-процесів за допомогою систем штучного інтелекту. *Бізнес Інформ*. 2019. № 3. С. 106–116.

5. *Герасимчук В. Г.* Моделювання та прогнозування економічних процесів : підручник. Київ : КПІ ім. Ігоря Сікорського, 2020. 348 с.

6. *Глибовець М. М., Олецький О. В.* Штучний інтелект : підручник. Київ : ВД «Києво-Могилянська академія», 2018. 460 с.

7. *Державна служба статистики України*. Методологічні положення щодо організації робіт за програмою державних статистичних спостережень. Київ, 2020. URL: http://www.ukrstat.gov.ua (дата звернення: 10.09.2025).

8. *Камінський Р. М.* Використання рекурентних нейронних мереж для прогнозування фінансових часових рядів. *Вісник Національного університету «Львівська політехніка». Інформаційні системи та мережі*. 2021. Вип. 9. С. 112–120.

9. *Клебанова Т. С., Гурьянова Л. С., Трунова Т. М.* Моделі оцінки та аналізу економічної безпеки підприємства : монографія. Харків : ХНЕУ ім. С. Кузнеця, 2017. 230 с.

10. *Кузьмін О. Є., Мельник О. Г.* Теоретичні та прикладні засади менеджменту : навч. посіб. Львів : Нац. ун-т «Львівська політехніка», 2019. 396 с.

11. *Литвин В. В., Висоцька В. А.* Інтелектуальні системи підтримки прийняття рішень : навч. посіб. Львів : Вид-во Львівської політехніки, 2019. 324 с.

12. *Лук'яненко І. Г., Краснікова Л. І.* Економетрика : підручник. Київ : Знання, 2018. 494 с.

13. *Матвійчук А. В.* Штучний інтелект в економіці: нейронні мережі, нечітка логіка : монографія. Київ : КНЕУ, 2018. 439 с.

14. *Мельник А. О.* Архітектура нейронних мереж глибокого навчання для обробки послідовностей. *Системні дослідження та інформаційні технології*. 2022. № 1. С. 25–38.

15. *Поляков М. В.* Прогнозування продажів у роздрібній торгівлі з використанням методів машинного навчання. *Маркетинг і цифрові технології*. 2020. Т. 4, № 2. С. 56–65.

16. *Субботін С. О.* Нейронні мережі: теорія та практика : навч. посіб. Житомир : Вид-во ЖДУ ім. І. Франка, 2020. 256 с.

17. *Тимченко А. А.* Економічна кібернетика : підручник. Київ : КНЕУ, 2019. 380 с.

18. *Черногор Н. А., Рубан С. А.* Порівняльний аналіз методів прогнозування часових рядів із застосуванням нейронних мереж LSTM та GRU. *Системні дослідження та інформаційні технології*. 2022. № 2. С. 45–56.

19. *Яровий А. А.* Інтелектуальний аналіз даних : навч. посіб. Вінниця : ВНТУ, 2018. 168 с.

20. *Box G. E. P., Jenkins G. M., Reinsel G. C.* Time Series Analysis: Forecasting and Control. 5th ed. Hoboken : Wiley, 2015. 712 p.

21. *Brownlee J.* Deep Learning for Time Series Forecasting: Predict the Future with MLPs, CNNs and LSTMs. Melbourne : Machine Learning Mastery, 2019. 210 p.

22. *Chollet F.* Deep Learning with Python. 2nd ed. New York : Manning Publications, 2021. 504 p.

23. *Cho K., van Merrienboer B., Gulcehre C.* Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, 2014. P. 1724–1734.

24. *Gers F. A., Schmidhuber J., Cummins F.* Learning to forget: Continual prediction with LSTM. *Neural Computation*. 2000. Vol. 12, No. 10. P. 2451–2471.

25. *Goodfellow I., Bengio Y., Courville A.* Deep Learning. Cambridge : MIT Press, 2016. 800 p.

26. *Graves A.* Long Short-Term Memory. *Supervised Sequence Labelling with Recurrent Neural Networks*. Studies in Computational Intelligence. Vol 385. Springer, Berlin, Heidelberg, 2012. P. 37–45.

27. *Hochreiter S., Schmidhuber J.* Long Short-Term Memory. *Neural Computation*. 1997. Vol. 9, No. 8. P. 1735–1780.

28. *Huber P. J.* Robust Estimation of a Location Parameter. *The Annals of Mathematical Statistics*. 1964. Vol. 35, No. 1. P. 73–101.

29. *Hyndman R. J., Athanasopoulos G.* Forecasting: Principles and Practice. 3rd ed. Melbourne : OTexts, 2021. URL: https://otexts.com/fpp3/ (дата звернення: 10.10.2025).

30. *Lai G., Chang W. C., Yang Y., Liu H.* Modeling Long- and Short-Term Temporal Patterns with Deep Neural Networks. *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*. 2018. P. 95–104.

31. *McKinney W.* Python for Data Analysis: Data Wrangling with Pandas, NumPy, and Jupyter. 3rd ed. Sebastopol : O'Reilly Media, 2022. 547 p.

32. *Raschka S., Mirjalili V.* Python Machine Learning: Machine Learning and Deep Learning with Python, scikit-learn, and TensorFlow 2. 3rd ed. Birmingham : Packt Publishing, 2019. 770 p.

33. *Shevat A.* Designing Bots: Creating Conversational Experiences. Sebastopol : O'Reilly Media, 2017. 300 p.

34. *Smyl S.* A hybrid method of exponential smoothing and recurrent neural networks for time series forecasting. *International Journal of Forecasting*. 2020. Vol. 36, Iss. 1. P. 75–85.

35. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems* [Електронний ресурс] / M. Abadi et al. 2015. URL: https://www.tensorflow.org/ (дата звернення: 15.09.2025).

36. *Telegram Bot API Documentation* [Електронний ресурс]. URL: https://core.telegram.org/bots/api (дата звернення: 20.09.2025).

37. *Vaswani A., Shazeer N., Parmar N.* Attention Is All You Need. *Advances in Neural Information Processing Systems*. 2017. Vol. 30. P. 5998–6008.

38. *Yamak P. T., Lifvan L., Bijan P. K.* A Comparison between ARIMA, LSTM, and GRU for Time Series Forecasting. *Algorithms*. 2019. Vol. 12, No. 11. P. 238.

39. *Zhang G. P.* Time series forecasting using a hybrid ARIMA and neural network model. *Neurocomputing*. 2003. Vol. 50. P. 159–175.

ДОДАТКИ

Додаток А

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Харківський національний університет імені В. Н. Каразіна

Навчально-науковий інститут комп'ютерних наук та штучного інтелекту
Кафедра комп'ютерних систем та робототехніки
Рівень вищої освіти (освітньо-кваліфікаційний рівень) ***Магістр***
Галузь знань: 17 – Електроніка, автоматизація та електронні комунікації
Спеціальність: 174 – Автоматизація, комп'ютерно-інтегровані технології та робототехніка
Освітня програма «Комп'ютеризовані системи управління та автоматика»

ЗАТВЕРДЖУЮ
Завідувач кафедри комп'ютерних
систем та робототехніки
к. ф.-м. н., доц. ХРУСЛОВ М. М.
«02» жовтня 2024року

З А В Д А Н Н Я
НА КВАЛІФІКАЦІЙНУ РОБОТУ

ДУБІНІНА Олексія Олексійовича
(прізвище, ім'я, по батькові студента)

1. Тема роботи **«МОДЕЛЬ АВТОМАТИЗОВАНОЇ СИСТЕМИ ПРОГНОЗУВАННЯ ЕКОНОМІЧНИХ ПОКАЗНИКІВ НА ОСНОВІ ЧАТ-БОТУ»**

керівник роботи **СТРІЛЕЦЬ Вікторія Євгенівна, кандидат технічних наук, доцент ЗВО кафедри комп'ютерних систем та робототехніки,**
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету ***від 30 вересня 2025 року № 4101-5/3554***

2. Строк подання студентом роботи ***30 листопада 2025 року***

3. Перелік питань, які потрібно розробити:

1. Аналіз сучасних методів прогнозування економічних показників, які представлені як часові ряди.
2. Огляд моделей і методів глибинного навчання для прогнозування часових рядів, зокрема глибинних рекурентних нейронних мереж (RNN, LSTM, GRU).
3. Проектування комп'ютерної системи прогнозування економічних показників на основі чат-боту.
4. Програмна реалізація комп'ютерної система та її модулів.
5. Експериментальне дослідження та тестування комп'ютерної системи і моделі прогнозування як основної складової.

4. План роботи

№ з/п	Назви етапів роботи	Термін виконання етапів роботи
1	Затвердження теми роботи та керівника	02.09.2024 - 02.10.2024
2	Аналіз та пошук методичної літератури щодо сучасних методів прогнозування часових рядів та економічних показників	03.09.2024 - 24.10.2024
3	Огляд і порівняльний аналіз архітектур рекурентних нейронних мереж (RNN, LSTM, GRU) для задач регресії	25.10.2024 - 09.11.2024
4	Обґрунтування вибору програмних засобів (Python, Keras) та розробка структури набору даних (Data preparation)	10.11.2024 - 24.11.2024
5	Розробка математичної моделі прогнозування та проведення експериментів з налаштування гіперпараметрів	25.11.2024 - 08.12.2024
6	Програмна реалізація модулів навчання моделі та генерації прогнозів з оцінкою невизначеності	09.12.2024 - 29.01.2025
7	Розробка архітектури та інтерфейсу чат-боту для взаємодії з користувачем	30.01.2025 - 28.02.2025
8	Тестування системи, аналіз точності прогнозів (метрики MAE, sMAPE) та валідація моделі	01.03.2025 - 01.04.2025
9	Підготовка і оформлення звітних матеріалів (чорновий варіант пояснювальної записки)	01.05.2025 - 30.08.2025
10	Оформлення звіту про науково-дослідну практику. Написання статті за матеріалами кваліфікаційної роботи	01.09.2025 - 30.10.2025
11	Підготовка і оформлення звітних матеріалів кваліфікаційної роботи. Оформлення списку літератури	10.10.2025 - 30.10.2025
12	Оформлення пояснювальної записки кваліфікаційної роботи відповідно вимогам до звітів про НДР	10.10.2025 - 30.11.2025
13	Підготовка і оформлення звітних матеріалів та додатків кваліфікаційної роботи (лістинги коду, акти впровадження)	01.11.2025 - 30.11.2025
14	Оформлення звіту про переддипломну практику	24.11.2025 - 30.11.2025
15	Представлення кваліфікаційної роботи керівнику та рецензенту	24.11.2025 - 30.11.2025

5. Дата видачі завдання ***02 жовтня 2024 року.***

Студент О. О. Дубінін _____
ініціали, прізвище підпис

Керівник роботи В. Є. Стрілець _____
ініціали, прізвище підпис

Додаток Б

Затверджую

_____

«_____»_____2025 р.

Технічне завдання
на розробку програмного виробу
«Модель автоматизованої системи прогнозування економічних показників на основі чат-боту»

1	**Вступ**	**1.1. Назва:** Модель автоматизованої системи прогнозування економічних показників на основі чат-боту **1.2. Галузь застосування:** комп'ютерні технології.
2	**Підстава для розробки**	**2.1.** Навчальний план за спеціальністю 174 – Автоматизація, комп'ютерно-інтегровані технології та робототехніка **2.2.** Завдання на кваліфікаційну роботу магістра № 4101-5/3554 від «30» вересня 2025 р. (представити як Додаток А до пояснювальної записки до кваліфікаційної роботи).
3.	**Призначення розробки**	**3.1. Мета розробки:** створення та програмна реалізація моделі автоматизованої системи для короткострокового прогнозування економічних показників підприємства з використанням рекурентних нейронних мереж та інтерфейсу чат-боту. **3.2. Призначення розробки:** система призначена для автоматизації процесу прогнозування попиту, надання оперативного доступу до аналітики через месенджер Telegram, оптимізації рівня страхових запасів та оцінки ризиків за допомогою розрахунку довірчих інтервалів. **3.3. Вихідні дані розробки:** історичні часові ряди продажів у форматі CSV (дата, артикул, кількість, ціна, промо-акції).
4.	**Технічні вимоги до програмного виробу**	**4.1. Вимоги до функціональних характеристик:** • Завантаження та валідація вхідних даних (CSV) через чат-бот; • Попередня обробка даних (ресемплінг, нормалізація, тригонометричне кодування дат); • Прогнозування попиту на 14 днів за допомогою моделей RNN/LSTM; • Розрахунок довірчих інтервалів (80%, 95%); • Візуалізація результатів (графік PNG) та експорт прогнозу (CSV). **4.2. Вимоги до надійності:** • Точність прогнозування (sMAPE) не гірше 10–12%; • Стійкість до аномальних викидів у даних (використання Huber Loss); • Коректна обробка помилок формату файлів. **4.3. Вимоги до умов експлуатації:** • Серверна частина: ОС Linux/Windows, Python 3.9+; • Клієнтська частина: встановлений месенджер Telegram (Mobile/Desktop). **4.4. Вимоги до складу і параметрів технічних засобів:** • Процесор: не менше 2 ядер (2.0 GHz);

<table>
<tr><td></td><td></td><td colspan="2">• Оперативна пам'ять: не менше 4 GB;
• Доступ до мережі Інтернет (для Telegram API).
4.5. Вимоги до інформаційної та програмної сумісності:
• Сумісність бібліотек: TensorFlow 2.x, Pandas, Scikit-learn;
• Формати обміну даними: JSON, CSV.
4.6. Вимоги до маркування та упаковки: Не висуваються (електронне розповсюдження).
4.7. Вимоги до транспортування і зберігання: Зберігання у хмарному сховищі або на локальному сервері.
4.8. Спеціальні вимоги: Забезпечення конфіденційності токену доступу до Telegram Bot API.</td></tr>
<tr><td>5.</td><td>Вимоги до програмної документації</td><td colspan="2">Програмною документацією до виробу «Модель автоматизованої системи прогнозування економічних показників на основі чат-боту» вважати:
1) Дане Технічне завдання (представити у вигляді Додатку Б до пояснювальної записки);
2) Опис математичної моделі та алгоритмів (у вигляді розділу 2 пояснювальної записки);
3) Інструкцію користувача та опис програмної реалізації (у вигляді розділу 3 пояснювальної записки).</td></tr>
<tr><td>6.</td><td>Вимоги до техніко-економічних показників</td><td colspan="2">1) Скорочення часу на отримання прогнозу: з 1.5–2 годин (ручний режим) до 1 хвилини (автоматичний режим).
2) Підвищення точності прогнозування порівняно з базовими методами (Seasonal Naive) не менше ніж у 2 рази.
3) Відсутність витрат на ліцензійне ПЗ (використання Open Source бібліотек Python).</td></tr>
<tr><td>7.</td><td>Стадії і етапи розробки</td><td>Дата</td><td>Назва етапу</td></tr>
<tr><td></td><td></td><td>02.09.2024 - 02.10.2024</td><td>– Затвердження теми роботи та керівника</td></tr>
<tr><td></td><td></td><td>03.09.2024 - 24.10.2024</td><td>– Аналіз та пошук методичної літератури щодо сучасних методів прогнозування часових рядів та економічних показників</td></tr>
<tr><td></td><td></td><td>25.10.2024 - 09.11.2024</td><td>– Огляд і порівняльний аналіз архітектур рекурентних нейронних мереж (RNN, LSTM, GRU) для задач регресії</td></tr>
<tr><td></td><td></td><td>10.11.2024 - 24.11.2024</td><td>– Обґрунтування вибору програмних засобів (Python, Keras) та розробка структури набору даних (Data preparation)</td></tr>
<tr><td></td><td></td><td>25.11.2024 - 08.12.2024</td><td>– Розробка математичної моделі прогнозування та проведення експериментів з налаштування гіперпараметрів</td></tr>
<tr><td></td><td></td><td>09.12.2024 - 29.01.2025</td><td>– Програмна реалізація модулів навчання моделі та генерації прогнозів з оцінкою невизначеності</td></tr>
</table>

		30.01.2025 - 28.02.2025 01.03.2025 - 01.04.2025 01.05.2025 - 30.08.2025 01.09.2025 - 30.10.2025 10.10.2025 - 30.10.2025 10.10.2025 - 30.11.2025 01.11.2025 - 30.11.2025 24.11.2025 - 30.11.2025 24.11.2025 - 30.11.2025	– Розробка архітектури та інтерфейсу чат-боту для взаємодії з користувачем – Тестування системи, аналіз точності прогнозів (метрики MAE, sMAPE) та валідація моделі – Підготовка і оформлення звітних матеріалів (чорновий варіант пояснювальної записки) – Оформлення звіту про науково-дослідну практику. Написання статті за матеріалами кваліфікаційної роботи – Підготовка і оформлення звітних матеріалів кваліфікаційної роботи. Оформлення списку літератури – Оформлення пояснювальної записки кваліфікаційної роботи відповідно вимогам до звітів про НДР – Підготовка і оформлення звітних матеріалів та додатків кваліфікаційної роботи (лістинги коду, акти впровадження) – Оформлення звіту про переддипломну практику – Представлення кваліфікаційної роботи керівнику та рецензенту
8.	**Порядок контролю і приймання програмного продукту**	1) Перевірку ходу розробки виконувати згідно з календарним планом. 2) Захист розробленої системи провести на засіданні Екзаменаційної комісії. 3) Пояснювальну записку подати на паперових носіях та в електронному вигляді згідно з вимогами ВНЗ.	

Виконавець:

студент групи КУ-61

Дубінін О.О.

Замовник:

к. т. н., доцент

Стрілець В. Є.

Додаток В

Програма і методика випробувань

програмного виробу

«Модель автоматизованої системи прогнозування економічних показників на основі чат-боту»

1. Об'єкт випробувань.

1. **Назва програмного виробу:** «Модель автоматизованої системи прогнозування економічних показників на основі чат-боту».
2. **Галузь застосування:** Автоматизація бізнес-процесів, логістика та управління запасами.
3. Відомості запозичуються з відповідних розділів Технічного завдання.

2. Мета випробувань. Перевірка відповідності функціональності програмної реалізації системи заявленим функціональним можливостям в Технічному завданні (Додаток Б до пояснювальної записки до дипломної роботи).

3. Загальні положення.

1. **Підстави для проведення випробувань:** Підставою для проведення випробувань є наказ про призначення атестаційної комісії.
2. **Місце і тривалість випробувань:** Приймальні (приймально-здавальні) випробування проводяться на базі комп'ютерного класу кафедри в період роботи атестаційної комісії.
3. **Обсяг випробувань:** Приймальні випробування програмного виробу проводяться в обсязі, відповідному цій програмі і методиці випробувань.
4. **Організації, які беруть участь у випробуваннях:** Приймальні випробування проводяться атестаційною комісією напередодні засідання (або в процесі засідання) за участю Замовника, Виконавця та інших осіб, присутніх на засіданні.

4. Вимоги до програми або програмного виробу. Модель повинна задовольняти наступним вимогам:

1. Працювати на основних операційних системах: Windows, Linux, MacOS;
2. Забезпечувати точність прогнозування (sMAPE) не гірше 10–12%;
3. Передбачити захист від некоректних дій користувача (завантаження файлів невірного формату);
4. Бути сумісною з Telegram Bot API;
5. Зменшити час отримання прогнозу порівняно з ручними методами;
6. Бути легко розширюваною (можливість додавання нових моделей RNN);
7. Елементи програми (інтерфейс бота та модуль прогнозування) повинні бути логічно відокремлені;
8. Вимоги до складу і параметрів технічних засобів (відповідно до ТЗ);
9. Вимоги до маркування та упаковки (не висуваються);
10. Вимоги до транспортування і зберігання (не висуваються).

5. Вимоги до програмної документації. Програмною документацією до виробу «Модель автоматизованої системи прогнозування економічних показників на основі чат-боту» вважати:

1. Справжнє Технічне завдання на розробку програми (представити як Додаток Б до пояснювальної записки до кваліфікаційної роботи);
2. Програму і методику випробувань розробленої програми (представити як Додаток В до пояснювальної записки до кваліфікаційної роботи);
3. Рекомендації щодо застосування створеної системи (представити в Розділі 3 пояснювальної записки до кваліфікаційної роботи).

6. Засоби і порядок випробувань

6.1. Засоби випробувань Для проведення випробувань необхідний персональний комп'ютер із встановленим інтерпретатором Python версії 3.9+, бібліотеками TensorFlow/Keras, Pandas, Matplotlib та доступом до мережі Інтернет для взаємодії з Telegram Bot API.

6.2. Порядок проведення випробувань: Як правило, випробування проводяться в два етапи:

- ознайомчий (1-й етап);
- випробування програмного виробу (2-й етап).

Перелік перевірок, що проводяться на 1 етапі випробувань:

1. Перевірка комплектності програмної документації.
2. Перевірка комплектності складу технічних і програмних засобів.
3. Якість програмної документації перевіряється на відповідність вимогам стандартів ЕСПД.

Перелік перевірок, що проводяться на 2 етапі випробувань:

1. Перевірка відповідності технічних характеристик програми вимогам технічного завдання.
2. Перевірка ступеня виконання функціональних вимог до програми.
3. Програма працює відповідно до умов експлуатації операційних систем MS Windows, Linux та MacOS.
4. Для роботи необхідний інтерпретатор мови програмування Python.

Порядок проведення функціональних тестів:

3.1. Запуск серверної частини програми здійснюється командою: python dubinin.py;

3.2. Після запуску програми необхідно відкрити месенджер Telegram та знайти бота за його іменем;

3.3. Надіслати команду /start для початку роботи;

3.4. Виконувати дії згідно зі сценаріями тестів.

Тест 1. Перевірка команди /start

1. Перевірка виконання програми.
2. Користувач надсилає команду /start у чат.

3. Отримання відповіді від бота з привітанням та інструкцією щодо формату файлу.

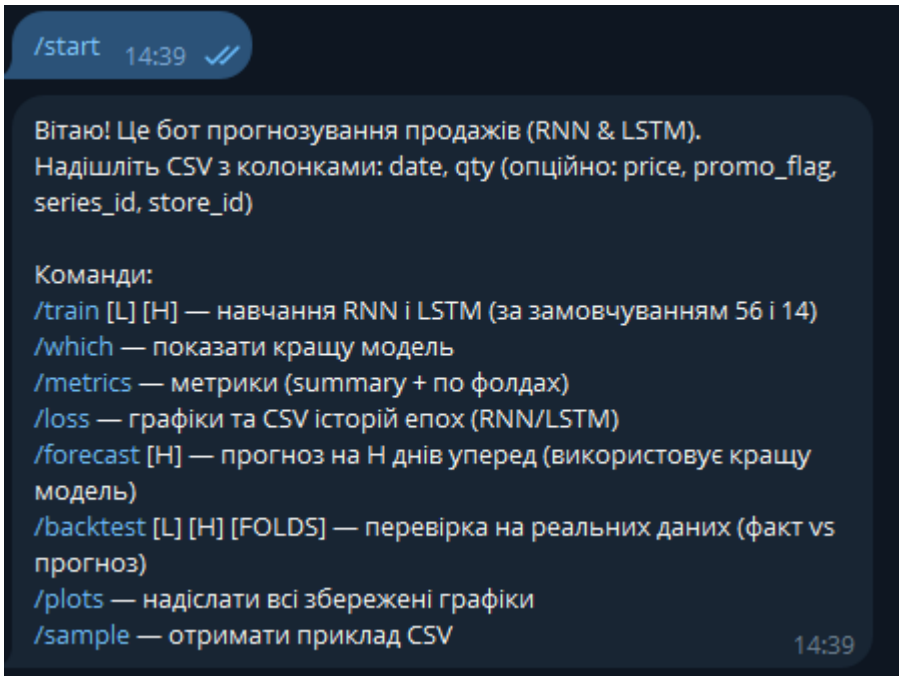

Рисунок В.1 – Тест 1

Тест 2. Завантаження коректного файлу даних

1. Перевірка виконання програми.
2. Користувач надсилає валідний файл sales.csv (з коректними колонками date, sku, qty).
3. Отримання відповіді від бота про успішний прийом файлу та початок обробки.

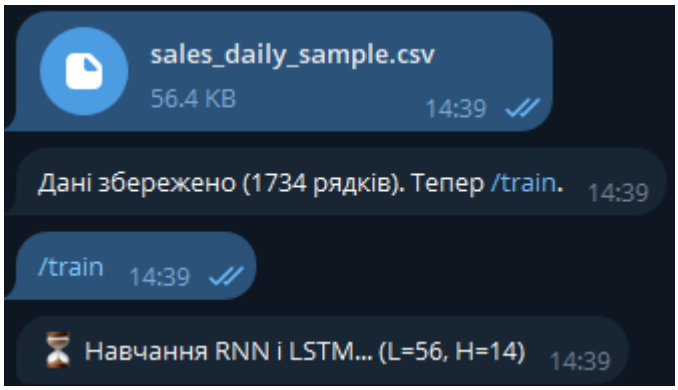

Рисунок В.2 – Тест 2

Тест 3. Обробка помилки формату даних

1. Перевірка виконання програми.
2. Користувач надсилає файл з порушеною структурою (наприклад, відсутня колонка price або текстовий файл .txt замість .csv).
3. Отримання відповіді від бота з інформацією про помилку та проханням виправити файл.

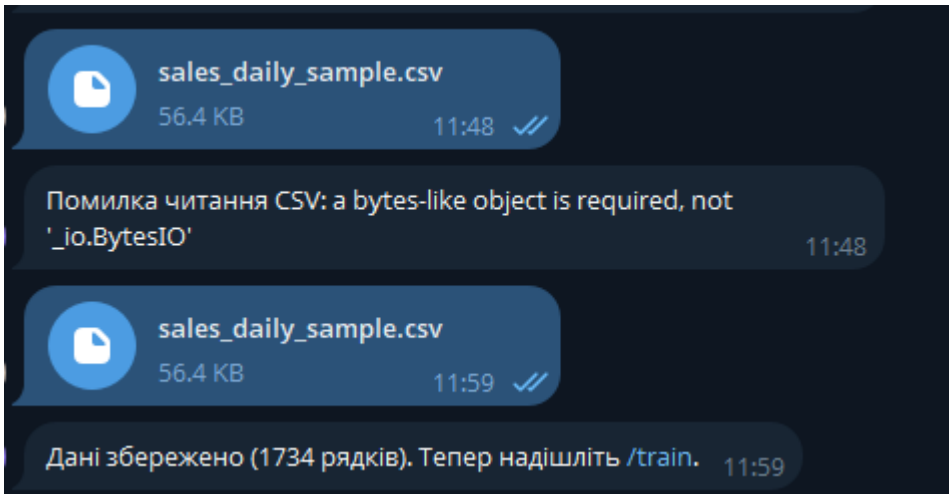

Рисунок В.3 – Тест 3.

Висновки: Тест 1 успішно пройшов випробування, Тест 2 успішно пройшов випробування і Тест 3 успішно пройшов випробування. Випробування пройшло успішно.

Виконавець: студент групи КУ61, Дубінін О. О.