НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»
Навчально-науковий інститут прикладного системного аналізу
Кафедра штучного інтелекту

До захисту допущено:
В. о. завідувачки кафедри
_____ Ірина ДЖИГИРЕЙ
«___»_____20__ р.

Дипломна робота

на здобуття ступеня бакалавра
за освітньо-професійною програмою «Системи і методи штучного інтелекту»
спеціальності 122 «Комп'ютерні науки»
на тему: «Система підтримки прийняття рішень в'їзного туризму Великої Британії рекурентними нейронними мережами глибокого навчання»

Виконала:
студентка IV курсу, групи КІ-03
Мітюк Дар'я Сергіївна _____

Керівник:
професор кафедри штучного інтелекту, д.т.н., професор
Данилов Валерій Якович _____

Консультант з економічного розділу:
доцент кафедри економічної кібернетики ФММ,
доктор філософії з економіки, доцент
Мажара Гліб Анатолійович _____

Консультант з нормоконтролю:
фахівець першої категорії кафедри штучного інтелекту,
Кравець Павло Володимирович _____

Рецензент:
директор НН ФТІ, д.т.н., професор
Новіков Олексій Миколайович _____

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.
Студентка _____

Київ – 2024 року

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Навчально-науковий інститут прикладного системного аналізу
Кафедра штучного інтелекту

Рівень вищої освіти – перший (бакалаврський)

Спеціальність – 122 «Комп'ютерні науки»

Освітньо-професійна програма «Системи і методи штучного інтелекту»

ЗАТВЕРДЖУЮ
В. о. завідувачки кафедри
_____ Ірина ДЖИГИРЕЙ
«31» січня 2024 р.

ЗАВДАННЯ
на дипломну роботу студенту
Мітюк Дар'ї Сергіївні

1. Тема роботи «Система підтримки прийняття рішень в'їзного туризму Великої Британії рекурентними нейронними мережами глибокого навчання», керівник роботи Данилов Валерій Якович, професор кафедри ШІ, д.т.н., професор, затверджені наказом по університету від « » 20 р. № _____

2. Термін подання студентом роботи «10» червня 2024 року.

3. Вихідні дані до роботи: історичні дані про загальну кількість туристів Великої Британії; кількість туристів з метою відпочинку; кількість туристів, що прибули авіатранспортом; кількість українців, що прибули до Великої Британії після початку повномасштабного вторгнення.

4. Зміст роботи: аналіз предметної області дослідження та даних, вибір доцільних методів та моделей для роботи, моделювання прогнозу обраними методами, оцінка та аналіз отриманих результатів.

5. Перелік ілюстративного матеріалу (із зазначенням плакатів, презентацій тощо): блок-схеми, таблиці, графіки історії тренування моделей.

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Економічний	доцент, доктор філософії з економіки, доцент кафедри економічної кібернетики ФММ Мажара Гліб Анатолійович		

7. Дата видачі завдання «05» лютого 2024 року.

Календарний план

№ з/п	Назва етапів виконання дипломної роботи	Термін виконання етапів роботи	Примітка
1	Затвердження теми ДР	15.04.2024 – 21.04.2024	Виконано
2	Ознайомлення зі структурою ДР згідно з Положенням про державну атестацію студентів НТУУ «КПІ»	22.04.2024 – 28.04.2024	Виконано
3	Ознайомлення з ДСТУ 3008:2015 та стандартами ЄСПД	29.04.2024 – 05.05.2024	Виконано
4	Проведення дослідження за темою ДР	05.05.2024 – 19.05.2024	Виконано
5	Проведення роботи над теоретичною частиною ДР	20.05.2024 – 26.05.2024	Виконано
6	Проведення роботи над програмним продуктом	27.05.2024 – 02.06.2024	Виконано
7	Оформлення ДР та аналіз отриманих результатів	03.06.2024 – 10.06.2024	Виконано

Студентка Дар’я МІТЮК

Керівник Валерій ДАНИЛОВ

РЕФЕРАТ

Дипломна робота: 86 с., 25 рис., 7 табл., 27 посилань, 1 додаток.

ШТУЧНИЙ ІНТЕЛЕКТ, МАШИННЕ НАВЧАННЯ, НЕЙРОННІ МЕРЕЖІ, ФРАКТАЛЬНА ІНТЕРПОЛЯЦІЯ, ТУРИСТИЧНІ ПОТОКИ.

У цій роботі розглянуто тему використання нейронних мереж з довгою короткочасною пам’яттю для прогнозування туристичних потоків, перспективи їх використання в сфері туризму.

Об’єктом дослідження стала історія кількості туристів, що відвідали Велику Британію впродовж 1980 – 2019 рр.

Предмет дослідження – нейронні мережі з довгою короткочасною пам’яттю для прогнозування змін туристичних потоків.

Мета роботи – визначити потенціал використання нейронних мереж з довгою короткочасною пам’яттю для прогнозування туристичних потоків.

У першому розділі досліджено предметну область завдання. Розглянуто різні аспекти ринку туризму Великої Британії, його особливості та наявні методи передбачення розвитку туристичних потоків. В другому розділі проведено огляд фрактальної інтерполяції як методу оптимізації роботи нейронних мереж з довгою короткочасною пам’яттю. В третьому розділі розроблено програмний продукт для дослідження ефективності нейронних мереж з довгою короткочасною пам’яттю для прогнозування туристичних потоків та проаналізовано його результати. В четвертому розділі оцінено характеристики програмного продукту та проведено функціонально-вартісний аналіз. Підбито підсумок про потенціал використання нейронних мереж з довгою короткочасною пам’яттю у сфері туризму, проведено аналіз моделі та описано способи її подальшого покращення.

ABSTRACT

Master's thesis: 86 p., 25 figures, 7 tables, 27 references, 1 appendix.

ARTIFICIAL INTELLIGENCE, MACHINiE LEARNING, NEURAL NETWORKS, FRACTAL INTERPOLATION, TOURISM FLOWS.

This research explains the usage of long short-term memory (LSTM) neural networks for tourism flows forecast and its prospects.

The object of research is the time series of overseas visits to United Kingdom for period 1980 – 2019.

The subject of research is a long short-term memory neural network for tourism flows forecast.

The purpose of the work is to determine the potential of using long short-term memory neural networks for forecasting tourism flows.

In the first chapter, the subject area of the task is examined. Various aspects of the UK's tourism market, its characteristics, and existing methods for forecasting tourism flows are considered. In the second chapter, a review of fractal interpolation as a method for optimizing the performance of long short-term memory neural networks is conducted. The third chapter develops a software product to study the effectiveness of LSTM neural networks for forecasting tourism flows and analyzes its results. In the fourth chapter, the characteristics of the software product are evaluated, and a functional-cost analysis is performed. A summary of the potential use of LSTM neural networks in the tourism sector is provided, along with an analysis of the model and descriptions of ways to further improve it.

ЗМІСТ

ВСТУП

Туризм є важливою складовою сучасної глобальної економіки. Він відіграє ключову роль у розвитку багатьох країн. Це багатогранне явище, що поєднує економічні, соціальні, культурні та екологічні аспекти, має значний потенціал для сталого розвитку. Туризм сприяє зростанню валового внутрішнього продукту (ВВП), створенню робочих місць, стимулює розвиток інфраструктури та сприяє культурному обміну між народами.

Велика Британія – це батьківщина сучасного туризму як способу проведення дозвілля. Вона приваблює туристів з усього світу своєю багатовіковою культурою, багатою історією та унікальним шармом і колоритом. А за бюджетними надходження в економіку країни галузь туризму поступається лише хімічній промисловості та фінансовому сектору.

У цій роботі буде досліджено потенціал використання рекурентних нейронних мереж глибокого навчання, а саме нейронних мереж з довгою короткочасною пам'яттю для прогнозування часових рядів туристичних потоків на прикладі Великої Британії. Буде також досліджена фрактальна інтерполяція, як метод оптимізації нейронних мереж та порівняно результати роботи додатку на різних наборах даних.

Цей проєкт є важливим для розробки та оптимізації вже наявних методів прогнозування часових рядів у сфері туризму. Результати роботи будуть корисними для прогнозування та планування туризму в країні з урахуванням системного підходу.

РОЗДІЛ 1 ДОСЛІДЖЕННЯ ВНУТРІШНЬОГО ТА В'ЇЗНОГО ТУРИЗМУ ВЕЛИКОЇ БРИТАНІЇ

1.1 Визначення поняття та структури туризму

В наш час можна знайти багато визначень поняттю «туризм», але найповнішим є опис туризму як тимчасових виїздів громадян та осіб без громадянства у вільний час з постійного місця проживання з метою оздоровлення, пізнання, професійного розвитку, спорту, релігії, бізнесу, освіти та інших цілей на термін від 24 годин до 6 місяців без можливості займатися оплачуваною діяльністю в місці тимчасового перебування [1].

Найважливішою ознакою туризму є вільний час людини, що включає відпустку, канікули, вихідні, святкові дні та час після виходу на пенсію. Другою важливою ознакою є мета подорожі, яка може бути оздоровчою, пізнавальною, професійною, спортивною, релігійною, діловою або освітньою тощо. Третя ознака встановлює часові рамки перебування, які повинні бути не менше 24 годин і не більше 6 місяців, що є важливим для туристичної статистики та економіки туристичної індустрії. Четверта ознака передбачає, що турист не може займатися оплачуваною діяльністю у місці тимчасового перебування, що означає наявність вільних грошових коштів для витрат на відпочинок та відновлення сил.

1.1.1 Типи туризму

В наш час туризм проявляється у різних поєднаннях та зв'язках, а тому дуже необхідна його класифікація. Зазвичай туризм поділяють за типами, видами та формами. Основними типами туризму відповідно до СОТ (Світової Організації Торгівлі) є внутрішній туризм та міжнародний туризм, який у свою

чергу включає в себе в’їзний та виїзний туризм. Блок-схему цієї структури показано на рис. 1.1.

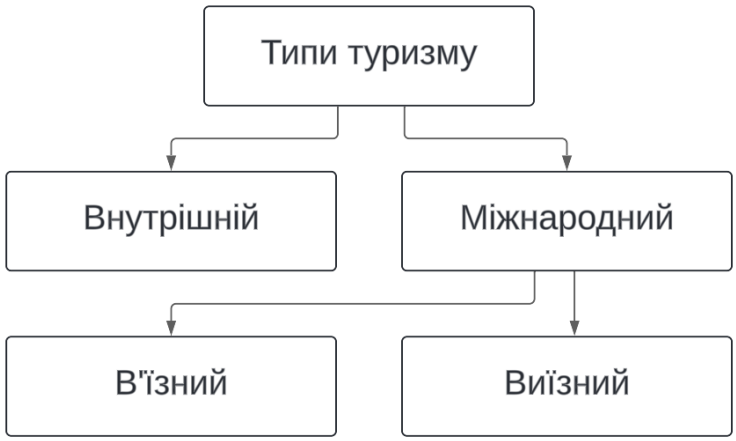

Рисунок 1.1 – Типи туризму

Внутрішній туризм є переміщенням людей у межах своєї країни з туристичними цілями. У таких подорожах використовуються національна валюта для розрахунків і рідна мова для спілкування, що спрощує організацію поїздок. За деякими оцінками, внутрішній туризм становить 80-90% усіх туристичних подорожей, а загальні витрати на нього в 5-10 разів перевищують витрати на міжнародні поїздки. Внутрішній туризм сприяє розвитку місцевої економіки, підтримує національні туристичні об'єкти та зменшує вплив на навколишнє середовище порівняно з міжнародним туризмом.

Міжнародний туризм включає поїздки людей за межі своєї країни з туристичними цілями. Перетин державного кордону супроводжується певними формальностями, такими як оформлення закордонних паспортів і віз, проходження митних процедур, валютний і медичний контроль. Ці правила запроваджуються державою для боротьби з незаконною міграцією, міжнародним тероризмом, торгівлею наркотиками, проституцією тощо, а також для забезпечення встановленого порядку в'їзду та виїзду з країни. Міжнародний туризм поділяють на в’їзний та виїзний.

В'їзний туризм - це вид туризму, коли люди відвідують інші країни для відпочинку, ознайомлення з культурою, відвідування пам'яток або інших туристичних атракцій.

Виїзний туризм - це вид туризму, коли люди подорожують зі своєї країни в інші країни для відпочинку, екскурсій, бізнесу або будь-яких інших цілей.

Усі типи туризму можуть по-різному поєднуватися між собою, утворюючи кілька категорій:

– туризм у межах однієї країни поєднує внутрішній та в'їзний туризм;
– національний туризм поєднує виїзний та внутрішній туризм;
– міжнародний туризм включає в'їзний та виїзний туризм.

Вказані категорії можуть стосуватися конкретної країни, регіону всередині цієї країни або регіону, що охоплює декілька країн.

1.1.2 Види туризму

Розрізняють дуже багато видів туризму за різними категоріями, такими як: мета поїздки, тривалість подорожі, інтенсивність відвідування, вікова категорія, джерела фінансування, вид туристичної діяльності, спосіб переміщення тощо, тому розглянемо найпоширеніші (рис. 1.2).

За метою подорожі туризм поділяють на рекреаційний, пізнавальний, лікувально-оздоровчий, діловий, спортивний, етнічний, релігійний, транзитний, освітній тощо.

Рекреаційний туризм – це форма туризму, спрямована на відновлення фізичних і психологічних сил, відпочинок, релаксацію та оздоровлення. Цей вид туризму зазвичай включає активності, які допомагають людям зняти стрес, покращити своє здоров'я і підвищити загальний рівень благополуччя.

Пізнавальний туризм – це форма туризму, орієнтована на отримання нових знань, вражень та досвіду через ознайомлення з культурними,

історичними, природними та науковими об'єктами. Цей вид туризму передбачає активне залучення туристів у пізнавальні активності та вивчення різних аспектів місця призначення.

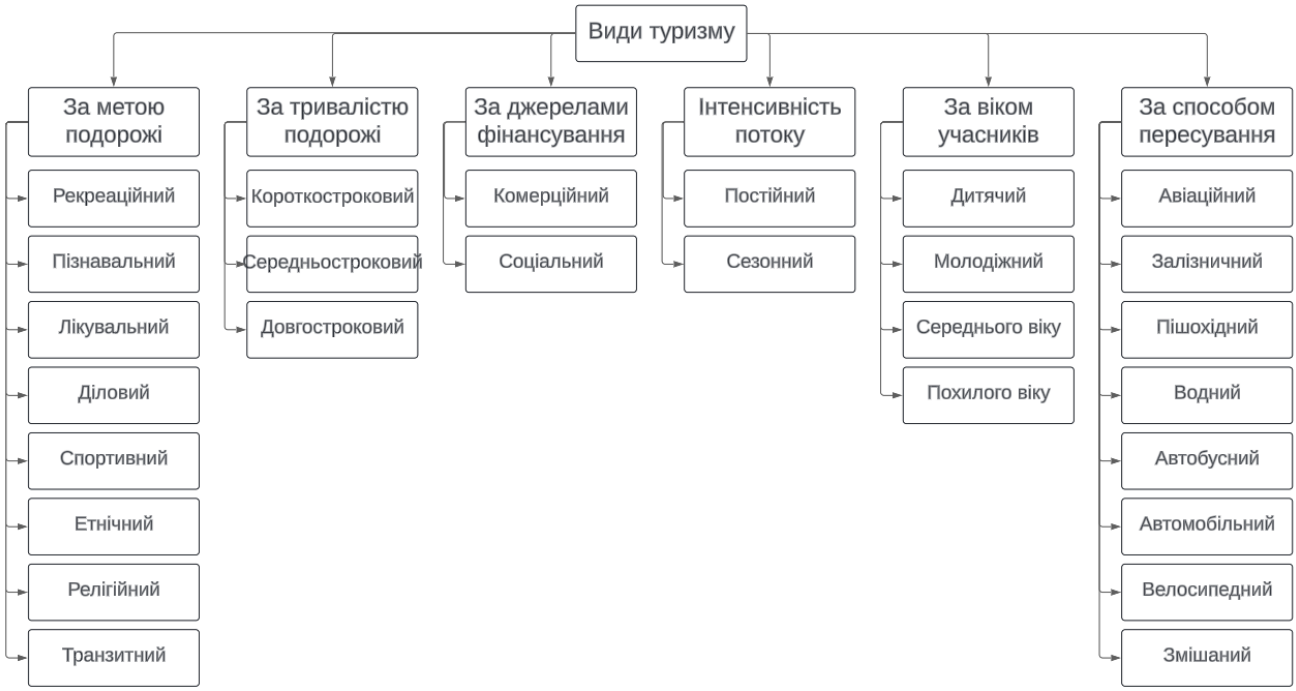

Рисунок 1.2 – Види туризму

Лікувально-оздоровчий туризм спрямований на покращення здоров'я та профілактику захворювань через використання природних ресурсів і спеціалізованих медичних послуг. Цей вид туризму поєднує відпочинок з лікувальними процедурами і є важливим елементом медичного та рекреаційного туризму.

Діловий туризм пов'язаний з поїздками для виконання службових обов'язків, участі в бізнес-заходах та професійних зустрічах. Цей вид туризму охоплює різноманітні види діяльності, що мають на меті професійний розвиток, встановлення ділових контактів та вирішення робочих завдань.

Спортивний туризм – це форма туризму, пов'язана з участю в спортивних заходах, змаганнях або активному відпочинку, що включає заняття різними видами спорту. Цей вид туризму орієнтований як на професійних спортсменів,

так і на аматорів, які прагнуть активно проводити час, розвивати свої фізичні здібності і підтримувати здоровий спосіб життя.

Етнічний туризм спрямований на знайомство з культурою, традиціями, звичаями та способом життя різних етнічних груп. Він передбачає відвідування місць, де зберігається унікальна етнічна спадщина, та взаємодію з місцевим населенням з метою глибшого розуміння їх культурних особливостей.

Релігійний туризм – це форма туризму, спрямована на відвідування місць, які мають релігійне значення, та участь у релігійних обрядах і святах. Він включає паломництво, відвідування святих місць, храмів, монастирів, участь у релігійних заходах і церемоніях.

Транзитний туризм – це форма туризму, при якій подорожуючі перетинають територію однієї або декількох країн (або регіонів) без тривалого перебування в них, зазвичай на шляху до кінцевого пункту призначення. Транзитні туристи зазвичай проводять короткий час в транзитній зоні, зупиняючись лише для пересадки або короткого відпочинку.

Освітній туризм передбачає подорожі з метою здобуття знань, професійного розвитку, навчання або участі в освітніх заходах. Такий туризм може включати участь у курсах, семінарах, конференціях, мовних школах або навчальних обмінах.

Також існують екологічний, гастрономічний, пляжний, культурний, пригодницький, сімейний, гірський, екстремальний, науковий, паломницький, ностальгічний, екзотичний, елітарний, спортивний туризм тощо.

За тривалістю подорожі виділяють короткостроковий туризм (поїздки до 7 днів та тури “вихідного дня”), середньостроковий туризм (поїздки від 9 до 12 днів) та довгостроковий туризм (поїздки тривалістю від 15 до 30 днів).

Залежно від **способу пересування** туризм поділяють на: авіаційний, залізничний, пішохідний, водний (річковий та морський), автобусний, автомобільний, велосипедний, змішаний тощо.

За інтенсивністю туристичного потоку туризм поділяється на постійний та сезонний. *Постійний туризм* описує регулярне відвідування

туристичних регіонів та об'єктів протягом усього календарного року без великих коливань. *Сезонний туризм*, натомість, відноситься до відвідування туристичних місць, частота якого змінюється залежно від кліматичних умов та особливостей конкретного місця. До сезонних видів можна віднести рекреаційний туризм та конгресно-виставковий туризм.

Сезонність туризму також визначається типом самого туризму. Вона поділяється на *високий, середній і низький сезони*. Високий сезон відзначається максимальним використанням туристами доступних туристичних ресурсів. Чинники, що впливають на сезонність, включають природно-кліматичні умови, загальнонаціональні або релігійні свята, канікули та масові заходи.

Залежно від віку туристів виділяють наступні види туризму: дитячий, молодіжний, середнього віку та похилого віку. Вікова градація туристів, запропонована Всесвітньою туристичною організацією, включає такі категорії: діти, які подорожують з батьками (до 15 років), молодь (15 - 24 роки), економічно активна молодь (25 - 44 роки), економічно активні туристи середнього віку (45 - 64 роки) та туристи похилого віку (65 років і старші). Мотивація подорожей та фінансові можливості туристичних поїздок часто залежать від віку туристів, що робить вік важливим фактором для планування туризму.

За джерелами фінансування виділяють комерційний та соціальний туризм. *Комерційний туризм* спрямований на отримання прибутку туристичними підприємствами та є основним джерелом розвитку сфери послуг. З метою максимізації прибутку туристичні компанії постійно розглядають оптимальне поєднання витрат і ціни турпродукту. Такі послуги переважно спрямовані на клієнтів із середнім та високим рівнем доходу. *Соціальний туризм* фінансується за рахунок коштів, що призначені для соціальних потреб, з метою створення умов для подорожей певних категорій громадян. Фінансування може здійснюватися як з державних, так і з недержавних фондів, а також через благодійні організації. В Манільській декларації визначено, що

"соціальний туризм - це мета, яку суспільство повинно ставити перед собою в інтересах менш забезпечених громадян".

1.1.3 Форми туризму

У контексті туризму "форми" можна розглядати як способи, за якими туристи організовують свої подорожі та відпочинок, залежно від їхніх потреб, уподобань, обмежень та інших факторів (рис. 1.3).

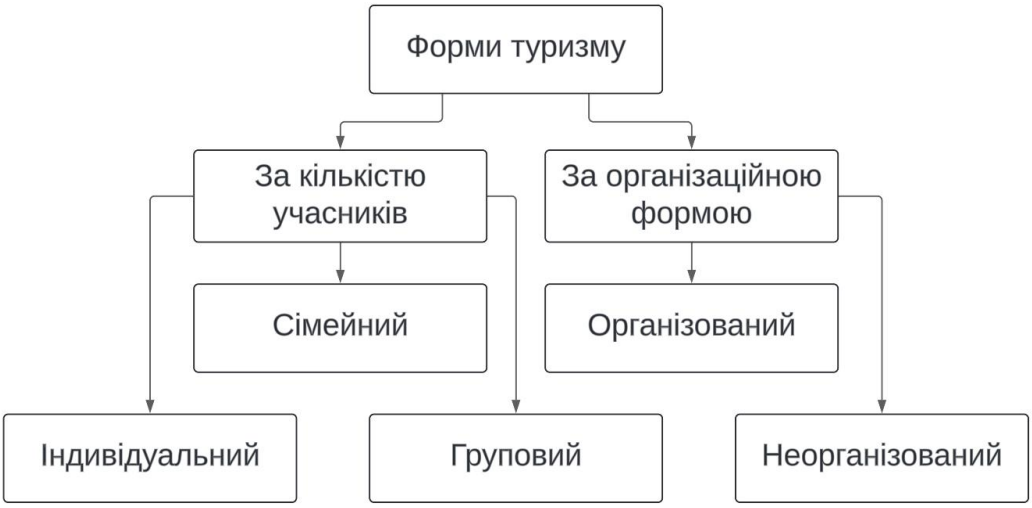

Рисунок 1.3 – Форми туризму

Залежно від кількості учасників туризм поділяють на індивідуальний, сімейний та груповий.

Індивідуальний туризм - це форма подорожей, де поїздка організована однією особою або для одного туриста. Тут кожен турист самостійно планує свою поїздку, включаючи маршрут, місце проживання та програму відпочинку.

Сімейний туризм - це вид подорожей, де учасниками є члени однієї сім'ї. Цільовими місцями можуть бути курорти, місця відпочинку або екскурсійні

тури. Планування та організація подорожі зазвичай відбувається за участі всіх членів сім'ї.

Груповий туризм - це колективна форма подорожей, де група людей вирушає разом на відпочинок або екскурсію. Група може бути складена з друзів, колег по роботі, студентів або учасників туристичної програми. Цей вид туризму зазвичай включає заздалегідь спланована програму та організований графік екскурсій та відпочинку.

За організаційною формою існують організований та неорганізований туризм.

Організований туризм - це форма подорожей, яка включає у себе організацію і керівництво подорожжю професійними туристичними агентствами або туроператорами. У цьому типі туризму весь процес від планування до виконання подорожі здійснюється спеціалізованими організаціями. Організатори можуть розробляти туристичні програми, включаючи визначення маршрутів, бронювання готелів, організацію екскурсій, транспортні послуги тощо. Вони також можуть забезпечувати супровід групи, надавати інформаційну підтримку та різні послуги для забезпечення комфортної та безпечної подорожі. Організований туризм може бути як груповим, так і індивідуальним, в залежності від потреб та уподобань туристів.

Неорганізований туризм – це форма подорожей, в якій туристи самостійно організовують свої поїздки без використання послуг професійних туристичних агентств або туроператорів. У цьому типі туризму люди самостійно планують свої маршрути, бронюють готелі або інші види проживання, організовують транспорт, вибирають екскурсії та відвідують різноманітні атракції. Цей тип туризму може бути привабливим для тих, хто шукає більшу свободу та незалежність у плануванні та виконанні своїх подорожей, а також для тих, хто бажає досліджувати нові місця самостійно і власними силами. Однак він також може вимагати більшого зусилля та витрат у порівнянні з організованим [2].

1.2 В'їзний туризм Великої Британії

Кількість відвідувачів.

У 2022 році було зареєстровано 31.2 мільйона подорожей до Великої Британії іноземцями, що є більшою кількістю, ніж у 2021 та 2020 роках, коли пандемія Covid-19 суттєво обмежила міжнародні подорожі. Хоча ця цифра на 10 мільйонів менше, ніж показник у 2019 році, який становив 40.9 мільйона відвідувань, більш пізні дані за квартали вказують на 9.6 мільйона відвідувань лише у останньому кварталі 2022 року, що майже рівно кількості подорожей у тому ж періоді 2018 та 2019 років [3].

Офіс національної статистики надає дані з 1980 року, коли до Великої Британії прибуло 12.4 мільйони іноземних мешканців. Цей показник зріс до 20 мільйонів у 1994 році, 30 мільйонів у 2006 році, а найвищий рівень досяг у 2017 році з показником 41.1 мільйони відвідувань. Ці дані зображено на рис. 1.4.

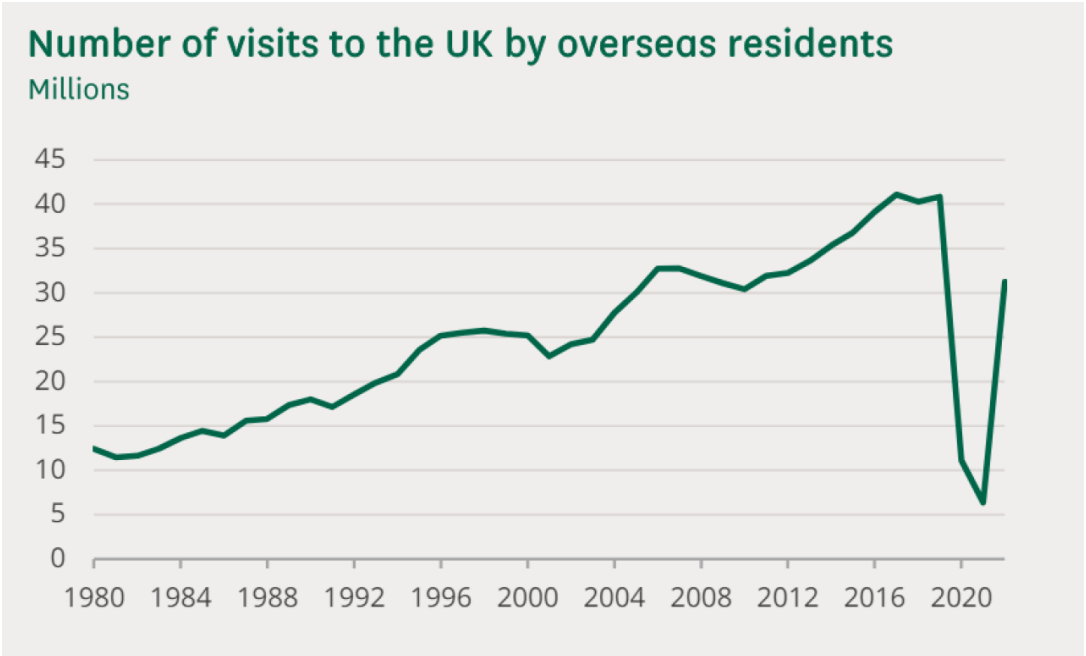

Рисунок 1.4 – Кількість відвідувань Великої Британії іноземними мешканцями

Хоча з середини 1990-х до 2019 років кількість іноземних відвідувачів у Великій Британії зростала, її частка в глобальних прибутках зменшилася [3]. Після пандемії кількість іноземних відвідувачів у Великій Британії в 2022 році була на 24% нижча, ніж у 2019 році, перевершуючи глобальне падіння на 34%, але поступаючись 19% падінням в Європі протягом того ж періоду часу [4]. Однак дані за перший квартал 2023 року є більш позитивними, з падінням кількості відвідувачів на 8% порівняно з тим самим періодом 2019 року, що подібно до 9% падіння в Європі.

Країна походження відвідувачів.

У 2020 році 31.2 мільйони іноземців відвідало Велику Британію, 21 мільйон з яких - жителі Європи, 5.5 мільйонів - Північної Америки та 4.8 мільйони - решти світу. Топ-3 країни, жителі яких відвідують Британію найчастіше – це США (4.6 млн візитів), Франція (2.8 млн візитів) та Ірландія (2.5 млн візитів). На рис. 1.5 зображено країни, жителі яких відвідали Велику Британію найбільше у 2022 році.

Хоча кількість туристів з Північної Америки прирівнялась до допандемічної, кількість відвідувачів з Європи знизилася більш ніж на 6 мільйонів, а відвідувачів з інших країн - на майже 3.5 мільйонів у 2020 році порівняно з 2019 роком. Близько 1 мільйона жителів Китаю відвідало Сполучене Королівство у 2019 році, але у 2020 їхня кількість зменшилась до менш ніж 100 тисяч через жорстку політику зовнішнього туризму в країні. Китай не відновлював прямих рейсів до Британії до серпня 2023 року [5].

Інші помітні зміни сталися у кількості туристів з Німеччини (знизилася на 1 мільйон або 31%), Італії (знизилася на 900 тисяч або 40%) та Франції (знизилася на 700 тисяч або 21%).

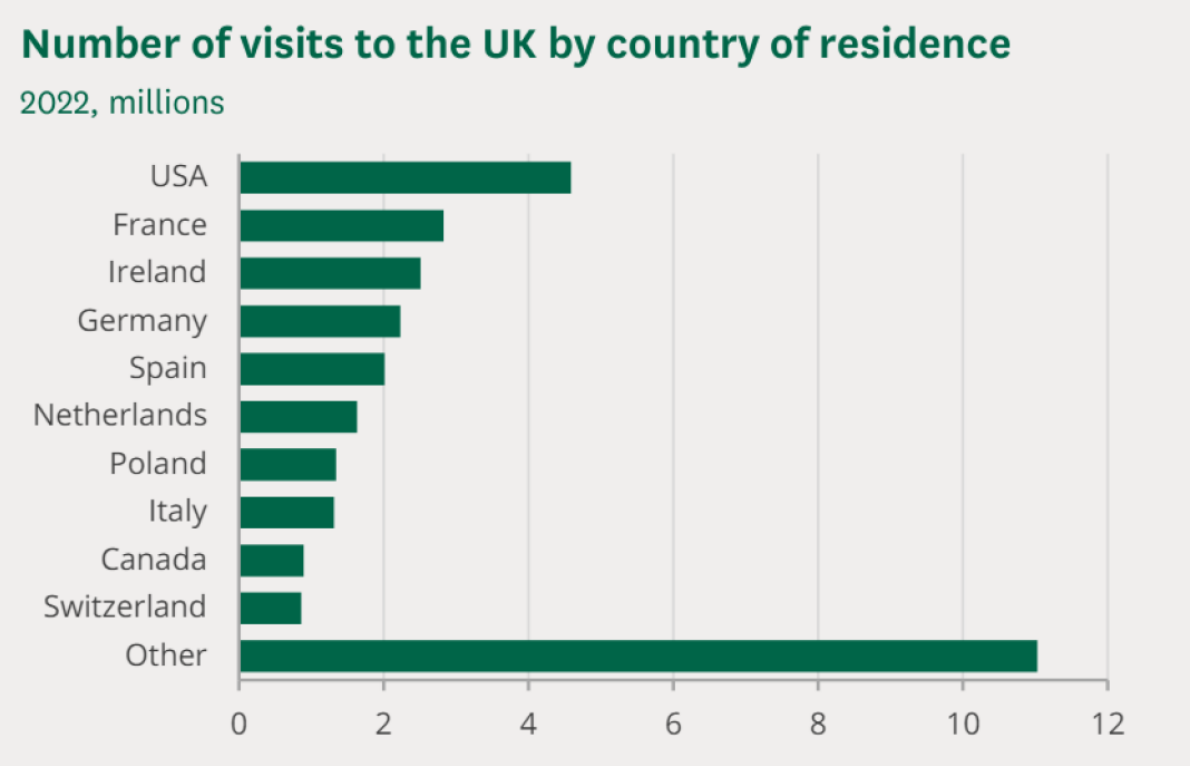

Рисунок 1.5 – Кількість відвідувань за країнами походження

Тривалість візиту.

Іноземні туристи провели у Британії 263 мільйони днів у 2020 році, що є нижчим за 290 мільйони днів у 2019 році. Тим не менш, кількість днів у фінальному кварталі (72 мільйони) зросла порівняно з таким самим кварталом у 2018 році (63 мільйони) та у 2019 році (67 мільйонів).

Середня тривалість візиту склала 8.4 дні у 2022 році та зросла порівняно з 7.1 днів у 2019 році і 7.2 днів у 2018 році. Жителі Європи провели у середньому найменше часу у Великій Британії - 6.1 ночей, за ними йдуть жителі Північної Америки - 8.9 ночей та інших країн світу - 17.9 ночей.

Основні напрямки.

У 2022 році було здійснено понад 16 мільйонів візитів до Лондона, що становить понад половину (52%) всіх візитів до Британії; 13 мільйонів поїздок було до інших частин Англії, 3 мільйони - в Шотландію і 700 000 - до Уельсу. Дані для Північної Ірландії не були опубліковані. Ці цифри перевищують

загальну кількість візитів у 31.2 мільйона, оскільки візит може включати ночівлі у кількох регіонах. Після Лондона найпопулярнішими місцями для відвідування були Единбург (1.8 мільйона відвідувань), Манчестер (1.2 мільйона) та Бірмінгем (800 тисяч) [6]. Це кількість ночівель у цих місцях, а не кількість візитів, оскільки турист може здійснити одноденну поїздку, скажімо, до Бірмінгема, залишаючись на ніч у Лондоні або навпаки.

Мета подорожі.

У 2022 році 39% туристів приїхало до Великої Британії на відпочинок, 38% - відвідати друзів чи родичів, 16% - у справах, а 7% - з інших причин (рис. 1.6).

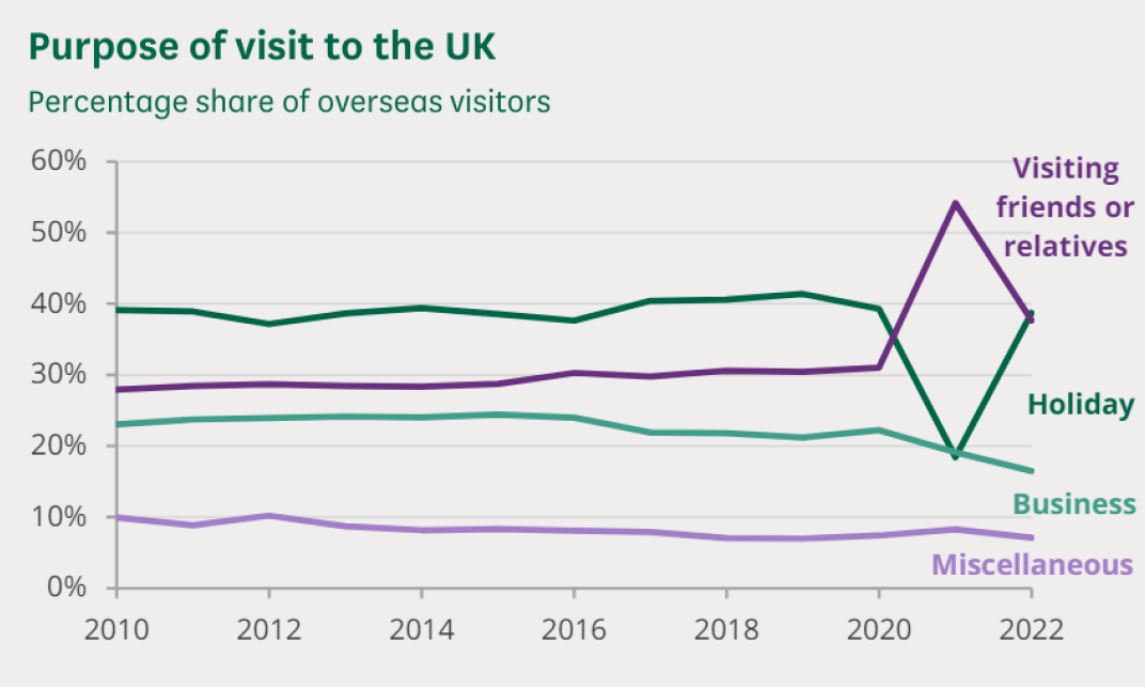

Рисунок 1.6 – Кількість відвідувань за метою візиту

У 2022 мандрівники частіше приїжджали з метою відвідати друзів та родичів і рідше по робочих справах. У 2021 ця різниця була ще більшою з 54% тих, хто приїхав відвідати рідних. Мандрівники, що приїжджають з метою відвідин проводять у середньому більше часу в країні (11.1 ночей в 2022 році), ніж ті, хто приїжджає на канікули (6.2 ночей) чи по роботі (4.6 ночей). Хоча

кількість відвідувачів у 2022 році була значно нижчою, ніж у 2019, кількість проведених у Великій Британії закордонними відвідувачами днів була більш схожою. Це у певній мірі пояснюється тим, що відвідувачі у 2022 році були схильніші подорожувати, щоб відвідати родину та друзів, а люди, які відвідують родичів і друзів, в середньому проводять найдовше часу у Великій Британії.

Мешканці Північної Америки ймовірніше відвідуватимуть Велику Британію для відпочинку (46%), ніж мешканці Європи чи решти світу (37%). Мешканці Європи найбільш схильні відвідувати з причин бізнесу (19%), а мешканці решти світу найбільш схильні відвідувати родичів та друзів (45%).

Витрати відвідувачів.

Туристи витратили 26.5 мільярдів фунтів стерлінгів у Британії у 2022 році, що є нижчим за 28.5 мільярдів у 2019 році та схожим на кількість витрат у 2018 році (26.5 мільярдів). Ці витрати подані у валютному еквіваленті і не враховують зміни внаслідок інфляції. Дорожні витрати до і з Сполученого Королівства не включені у статистику [7].

Враховуючи інфляцію, у 2022 році туристи витратили на 18% менше, ніж у 2019. В цілому, 6.9 мільярди фунтів було зароблено завдяки відвідувачам з Північної Америки, 11.7 мільярди - відвідувачам з Європи та 7.9 мільярди - завдяки решти країн. Відвідувачі з решти світу в середньому витрачають найбільше за візит - £1,650, за ними слідують мешканці Північної Америки з £1,250 і європейці з £550, хоча це в певній мірі відображає середню тривалість перебування кожної з цих груп у країні. На день європейці та відвідувачі з решти світу витрачають приблизно однакові суми (£90 і £92 відповідно), тоді як мешканці Північної Америки витрачають £141 на день.

Відвідувачі, які приїжджають з причин бізнесу, витрачають найбільше на день (£209), за ними слідують відпочивальники (£153), тоді як ті, хто відвідують родичів і друзів, витрачають £58 на день. Відвідувачі з материкового Китаю та Близького Сходу (окрім ОАЕ) в середньому витрачають найбільше на поїздку - £3,000 та £2,900 відповідно.

1.3 Внутрішній туризм Великої Британії

У 2022 році жителі Британії здійснили 126 мільйонів кількаденних поїздок у межах країни, провівши у цілому 383 мільйонів днів та витративши при цьому £32.9 мільярди. Це більше, ніж витрати міжнародних туристів у тому ж році (£26,5 мільярдів) [8].

Найбільш відвідуваною частиною Великої Британії виявилася Англія (107 млн поїздок), за нею йде Шотландія (13 мільйонів) та Уельс (9 мільйонів поїздок). Середні витрати на поїздку та за один день у Англії та Шотландії були досить схожими, проте витрати в Уельсі були трохи нижчими (відповідно £220 загалом та £74 на день порівняно з національними середніми величинами в £262 загалом та £86 на день). Відвідування родичів та друзів становило 36% від усіх поїздок, за якими слідували відпустки (34%), інші причини (25%) та бізнес (6%).

Далі ми проаналізуємо основні напрямки туризму всередині країни.

Англія

На відміну від в'їзного туризму, внутрішній туризм по Англії розподілений більш рівномірно. З 107 мільйонів відвідувань, які були здійснені в межах Англії, 18 мільйонів (16%) були до південно-західної частини, 16 мільйонів (15%) - до південно-східної, і по 15 мільйонів (14%) - до Лондона та північно-західної частини Англії. Лондон дійсно забезпечив найбільші прибутки (5.1 мільярдів фунтів стерлінгів або 18%), за якими слідували південно-західна частина (5 мільярдів фунтів стерлінгів або 18%) та північно-західна (4.2 мільярдів фунтів стерлінгів або 15%). На рис. 1.7 зображено розподіл кількості подорожей та витрат туристів станом на 2020 рік за регіонами країни.

Вельс

Південно-східний регіон Вельсу був найбільш популярним напрямом у 2022 році, куди було здійснено 2.7 мільйони поїздок. За ним слідував північний

Вельс (2.3 мільйони), південно-західний Вельс (1.8 мільйони) та центральний Вельс (0.8 мільйони) [9].

Відвідувачі Вельсу були більш схильні відвідати сільську місцевість, ніж в інших регіонах Великої Британії в середньому. Понад половина резидентів відвідала мале містечко, село або сільську місцевість, близько 21% відвідало узбережжя, а 29% - велике місто чи містечко. 6.2 мільйони (71%) усіх поїздок до Уельсу здійснили жителі Англії, 2.2 мільйони (25%) - жителі Уельсу, а решта 0.6 млн (4%) - Шотландії. Відвідувачі Уельсу, ймовірніше, будуть молодшими за середній показник по країні, адже половина з них віком від 16 до 35 років.

Шотландія

Понад половина поїздок у Шотландії були до міст і великих містечок [11]. 36% усіх поїздок були на захід Шотландії (включно з Глазго), 34% - до східного району (включно з Единбургом), 24% - на північ, а 6% - на південь Шотландії. Ці цифри відображають основні відвідувані регіони, оскільки туристи могли подорожувати до більш ніж одного регіону. Близько половини кількаденних подорожей по Шотландії було здійснено жителями цього регіону, трохи менше половини – жителями Англії та 1% - жителями Вельсу.

Північна Ірландія

На жаль, The Great Britain Visitor Survey не надає інформацію про внутрішній туризм у Північній Ірландії, починаючи з 2019 року, проте основні дані станом на 2019 рік:

- жителі Великої Британії здійснили 1.5 мільйони поїздок до Північної Ірландії, витративши при цьому 370 мільйонів фунтів стерлінгів[12];
- жителі Північної Ірландії здійснили 2.3 мільйони поїздок в межах регіону, витративши при цьому 310 мільйонів фунтів [13];
- жителі Північної Ірландії здійснили 1.1 мільйони поїздок до Великої Британії.

Domestic overnight visits and spending by region

2022

Region	Visits (millions)	Spend (£millions)
Great Britain	125.7	£32,882
England	106.9	£27,554
London	15.4	£5,087
North East	3.8	£932
North West	14.7	£4,187
Yorkshire and the Humber	10.1	£2,535
West Midlands	8.4	£1,679
East Midlands	8.5	£1,932
East of England	10.4	£2,294
South West	17.6	£4,983
South East	15.8	£3,521
Other/unspecified England	2.0	£403
Scotland	13.5	£3,410
North of Scotland	2.9	£783
East of Scotland	4.1	£1,079
South of Scotland	0.7	£160
West of Scotland	4.4	£1,172
Wales	8.7	£1,919
South East Wales	2.7	£556
North Wales	2.3	£571
South West Wales	1.8	£440
Mid Wales	0.8	£110

Рисунок 1.7 – Розподіл кількості подорожей та витрат туристів за регіонами Великої Британії станом на 2020 рік [10]

1.4 Постановка задачі

Задача дослідження передбачає використання рекурентної нейронної мережі глибокого навчання для прогнозування часових рядів туристичних потоків. Метою є створення нейронної мережі, яка може навчатися на основі недорогих або навіть безкоштовних даних без необхідності анотацій експертами. Дослідження передбачає використання методу фрактальної розмірeності часових рядів для покращення результатів роботи нейронної мережі з довгою короткочасною пам’яттю. В результаті ми порівняємо роботу нейронної мережі на звичайних даних та фрактально- інтерпольованих.

Висновки до розділу 1

У розділі 1 було розглянуто загальне поняття туризму та його структури, а також проведено аналіз внутрішнього та в’їзного туризму Великої Британії. У першому підпункті ми дослідили розподіл туристичних потоків за типами, видами та формами, що є важливим фактором для планування шляхів розвитку цього сектору економіки. У другому підпункті було детально проаналізовано такі метрики в’їзного туризму, як кількість іноземних відвідувачів, країни їхнього походження, середня тривалість візиту, мета подорожі, витрати та основні напрямки. У третьому підпункті наведено відсотковий розподіл відвідувачів за регіонами проживання, основні напрямки туризму, витрати тощо всередині країни за такими регіонами, як Англія, Вельс, Шотландія та Північна Ірландія. Збір, дослідження та аналіз даної інформації важливий для розробки стратегії розвитку сфери туризму в обраній для проєкту країні.

РОЗДІЛ 2 АНАЛІЗ МЕТОДІВ ОПТИМІЗАЦІЇ НЕЙРОННИХ МЕРЕЖ

Останнім часом багато дослідників використовують штучний інтелект, а саме машинне навчання та глибоке навчання, для прогнозу та аналізу історичних даних. Прогнозування має великий спектр застосувань, як от оцінка кількості майбутнього населення, передбачення епілептичних нападів, прогноз цін на фондовому ринку тощо. Результати таких досліджень показують перспективу у фізиці твердого тіла або ж для прогнозування сонячної радіації. У біології застосування включають геноміку, протеоміку та еволюцію, а у медицині нейронні мережі можуть використовуватися для ідентифікації фібриляції передсердь та застійної серцевої недостатності. Що стосується сільського господарства, машинне навчання може бути використане для прогнозування врожайності та оцінки статусу азоту або інших параметрів ґрунту.

Усі ці дослідження базуються на історичних даних, а це означає, що складність та фактична кількість даних є дуже важливим показником для роботи алгоритму. Розрізняють дві основні причини, пов’язані з якістю даних, через які алгоритм машинного навчання може працювати погано:

- недостатня кількість даних: для часових рядів це означає, що дані або недостатньо деталізовані, або недостатньо тривалі;
- випадкові дані: незважаючи на наявність достатньої кількості даних, внутрішній шум у даних заважає моделі робити точні прогнози.

Один зі способів отримати більш деталізовані дані – це провести їхню інтерполяцію [14]. Це можна зробити за допомогою лінійної або ж фрактальної інтерполяції. У більшості випадків реальні дані є нелінійними та походять зі складних систем із багатьма невідомими взаємодіями, тому підхід лінійної інтерполяції не відображає складності системи. Підхід, який враховує складність реальних систем при інтерполяції даних, полягає у використанні

фрактальної інтерполяції. На відміну від традиційних методів інтерполяції, які базуються на елементарних функціях, таких як поліноми, фрактальна інтерполяція базується на системах ітераційних функцій. Оскільки системи ітераційних функцій здатні генерувати фрактальні та мультифрактальні структури, внутрішня складність фрактально-інтерпольованих даних ближче до реальних даних, ніж традиційно інтерпольованих.

Загалом складність даних, що досліджуються, впливає на якість прогнозів. Зазвичай для цієї задачі використовується міра складності, така як показник Херста, фрактальна розмірність або міра ентропії.

Метою цього дослідження є визначення того, наскільки поєднання фрактальної інтерполяції та мір складності може бути використане для покращення прогнозів за допомогою базової реалізації нейронної мережі з довгою короткочасною пам’яттю (LSTM).

Ми застосували фрактальний метод інтерполяції до чотирьох наборів даних для підвищення їхньої деталізації. Також фрактальну інтерполяцію було адаптовано для відповідності складності оригінальних даних за допомогою показника Херста. Усі набори даних є реальними складними часовими рядами. Нейронну мережу типу LSTM навчають на одній частині даних, а потім налаштовують на невідомій частині. Це виконується для кожного набору даних.

2.1 Опис набору даних

Для цього дослідження було обрано чотири набори даних, що є реальними часовими рядами та описують різні метрики стану сфери туризму. Дані було узято з офіційного сайту статистики Великої Британії Office for National Statistics.

1. Загальна кількість поїздок. Цей набір даних включає кількість відвідувань Великої Британії іноземними мешканцями помісячно із

січня 1980 року до грудня 2023 року та не є сезонно-скоригованими. Для дослідження було взято лише дані до грудня 2019 року через суттєвий вплив пандемії Covid-19 на якість датасету. Робочий часовий ряд має 480 спостережень [15].

2. Кількість подорожей з метою відпочинку. Другий датасет включає в себе помісячну кількість туристів, що прибули до Великої Британії з метою відпочинку. Дані подані з січня 2015 року до грудня 2019 року та нараховують 60 спостережень.
3. Кількість туристів, що прибули авіатранспортом. Третій набір даних надає інформацію про кількість іноземних мешканців, що прибули до Великої Британії авіаційним транспортом. Дані подані поквартально з 2009 до 2019 року та нараховують 44 спостереження.
4. Кількість біженців з України. Четвертий набір даних надає інформацію про кількість українців, що прибули до Великої Британії після повномасштабного вторгнення за схемою Ukraine Sponsorship Scheme. Дані подаються потижнево з 28 березня 2022 року до 27 березня 2023 року та в цілому нараховують 53 спостереження [16].

2.2 Дослідження методу фрактальної інтерполяції

Для фрактальної інтерполяції було використано метод, описаний у роботі P. Manousopoulos “Curve Fitting by Fractal Interpolation” [17]. Традиційні методи інтерполяції базуються на елементарних функціях, таких як поліноми. На противагу цьому, фрактальна інтерполяція базується на системах ітераційних функцій (IFS). Система ітераційних функцій визначається як повний метричний простір X з відстаневою функцією h та скінченним набором стискальних

відображень, $\{w_n : X \to X$ для $n = 1, 2, \ldots, N\}$. Фрактально-інтерпольовані функції відтворюють дані зі складністю, ближчою до реальних даних, у порівнянні з поліномами.

Часовий ряд подається як набір точок даних, де $\{ (u_m, v_m) \in \mathrm{R}^2 \} : m = 0, 1, \ldots, M$. Інтерполяція потім застосовується до підмножини цих даних, тобто до точок інтерполяції $\{ (x_i , y_i) \in \mathrm{R}^2 : i = 0, 1, \ldots, N \}$. Обидва ряди лінійно впорядковуються зважаючи на їхню абсцису, тобто $u_0 < u_1 < \ldots < u_M$ та $x_0 < x_1 < \ldots < x_N$. Таким чином, набір точок інтерполяції розділяє набір точок даних на інтервали інтерполяції. У нашому застосуванні інтервали інтерполяції обираються рівновіддалено. Чим більше точок інтерполяції використовується, тим краще інтерполяція відповідає початковим даним, але більше точок інтерполяції призводить до меншого коефіцієнта стиснення (співвідношення оригінальних до інтерпольованих даних), оскільки для опису функції інтерполяції потрібно більше інформації.

$\{\mathrm{R}^2 ; w_n , n = 1, 2, \ldots, N\}$ – це система ітераційних функцій (IFS) з афінними перетвореннями:

$$\omega_n \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} a_n & 0 \\ c_n & s_n \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} d_n \\ e_n \end{bmatrix},$$

яка має задовольняти рівняння:

$$\omega_n \begin{bmatrix} x_n \\ y_n \end{bmatrix} = \begin{bmatrix} x_{n-1} \\ y_{n-1} \end{bmatrix} \quad \text{та} \quad \omega_n \begin{bmatrix} x_N \\ y_N \end{bmatrix} = \begin{bmatrix} x_n \\ y_n \end{bmatrix},$$

для кожного $n = 1, 2, \ldots, N$.

Розв’яжемо ці рівняння:

$$a_n = \frac{x_n - x_{n-1}}{x_N - x_0},$$

$$d_n = \frac{x_N x_{n-1} - x_0 x_n}{x_N - x_0},$$
$$c_n = \frac{y_n - y_{n-1}}{x_N - x_0} - s_n \frac{y_N - y_0}{x_N - x_0},$$
$$e_n = \frac{x_N y_{n-1} - x_0 y_n}{x_N - x_0} - s_n \frac{x_N y_0 - x_0 y_N}{x_N - x_0}.$$

Тут дійсні числа a_n, d_n, c_n, e_n визначаються точками інтерполяції, а s_n є вільним параметром, що називається коефіцієнтом вертикального масштабування. Щоб система ітераційних функцій (IFS) була гіперболічною з урахуванням відповідної метрики, s_n обмежений умовою $|s_n| < 1$. Параметр s_n буде використаний пізніше для дослідження складності інтерпольованої кривої.

2.3 Результати роботи методу фрактальної інтерполяції

Нижче описаний алгоритм було застосовано до кожного часового ряду для знаходження фрактальної інтерполяції, що відтворює реальні комплексні дані часових рядів:

1. Часовий ряд було поділено на *m* підмножин довжини *l*.
2. Для кожної підмножини і було пораховано показник Херста H_i.
3. Для кожної підмножини цей алгоритм було застосовано *k* разів, де *k* було попередньо встановлено рівним 200 для кожної підмножини:
   - використали метод фрактальної інтерполяції, що був описаний у розділі 2.2 з випадковим параметром s_n, що є сталим для всієї підмножини;
   - порахували показник Херста H_i для інтерпольованого часового ряду.

На рис. 2.1 – рис. 2.4 зображено результати фрактальної інтерполяції чотирьох часових рядів: перший описує кількість подорожей, здійснених іноземними мешканцями до Великої Британії, другий – кількість подорожей з

метою відпочинку, третій – кількість подорожей, здійснених авіатранспортом, а четвертий часовий ряд описує кількість українців, що прибули до Великої Британії після початку повномасштабного вторгнення.

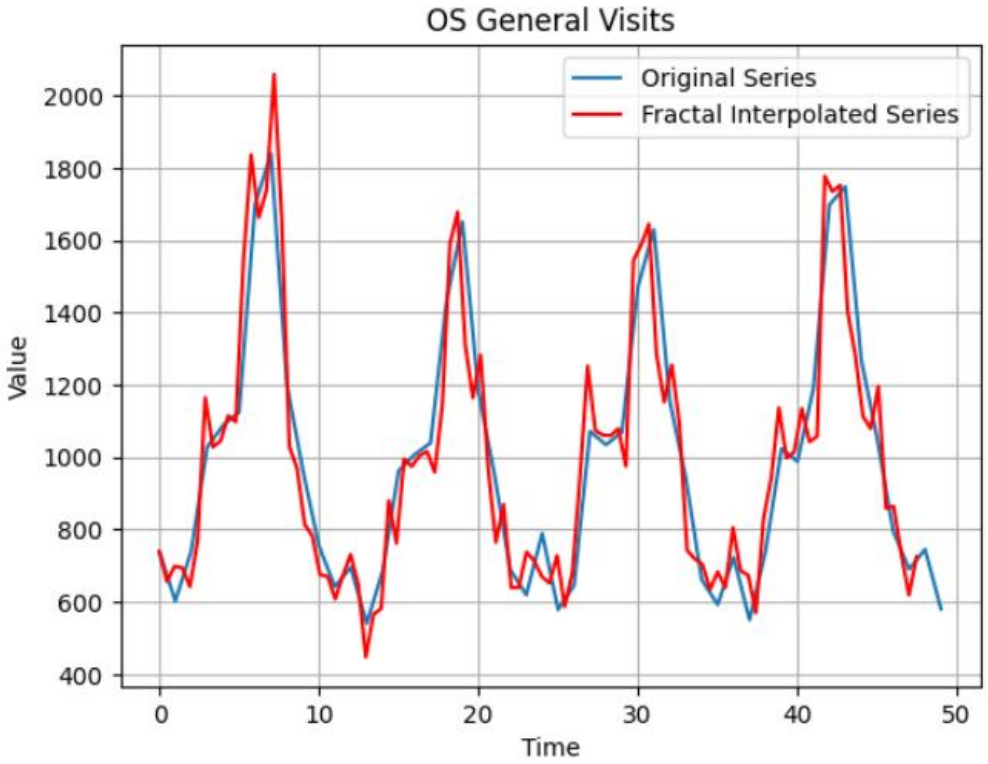

Рисунок 2.1 – Результат роботи фрактальної інтерполяції для першого набору даних

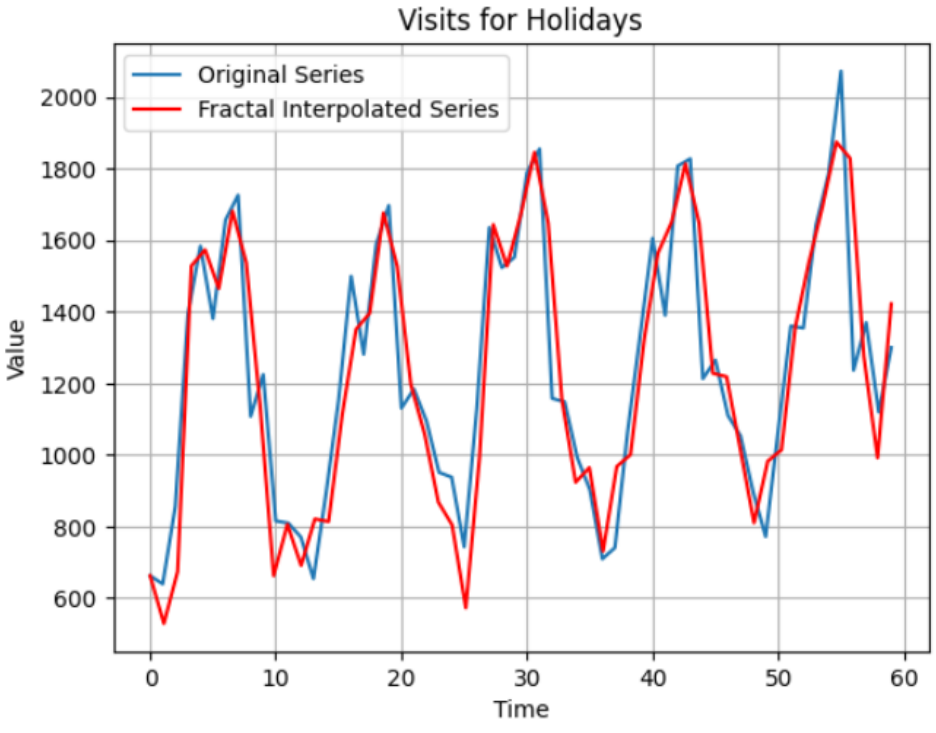

Рисунок 2.2 – Результат роботи фрактальної інтерполяції для другого набору даних

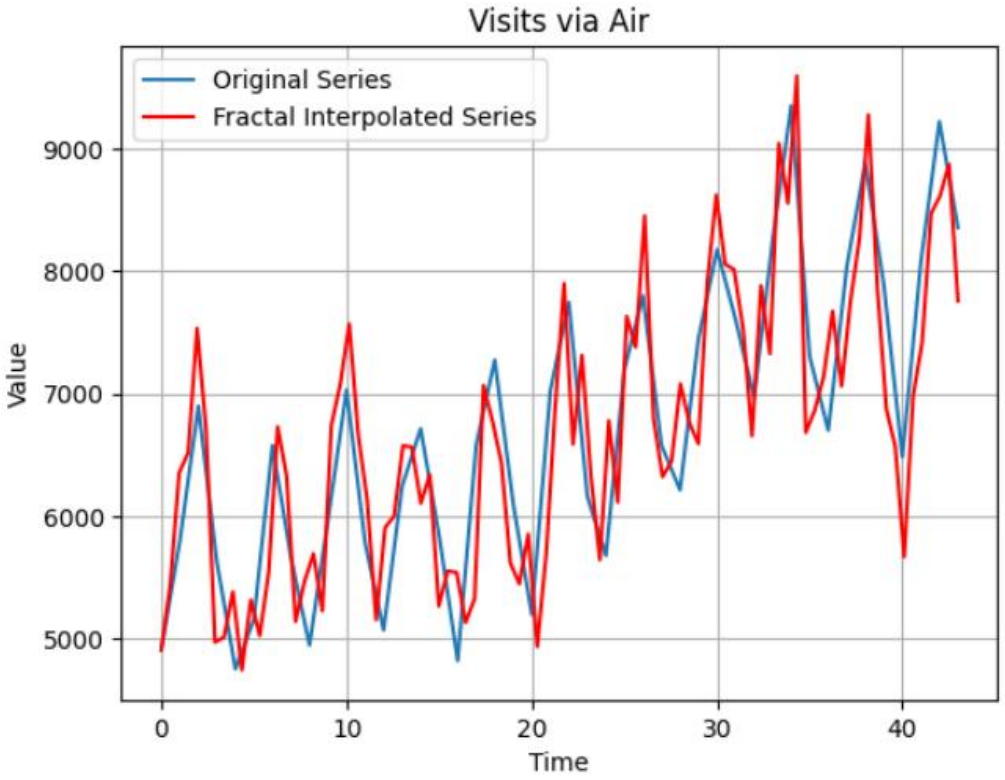

Рисунок 2.3 – Результат роботи фрактальної інтерполяції для третього набору даних

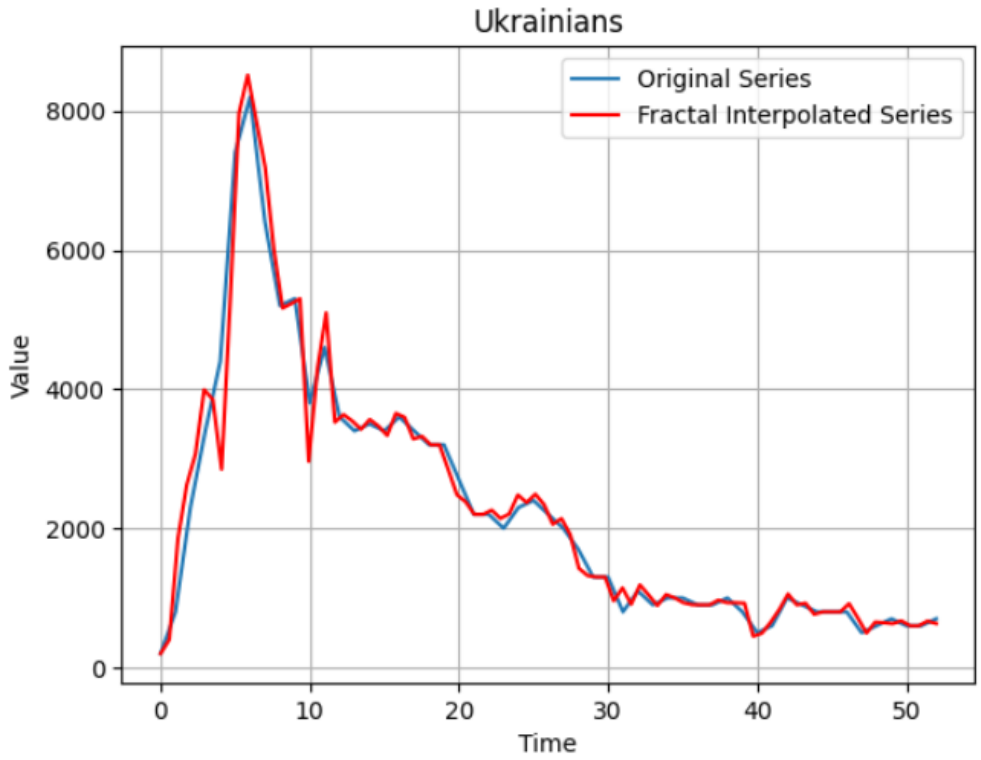

Рисунок 2.4 – Результат роботи фрактальної інтерполяції для четвертого набору даних

2.4 Показник Херста

Аналіз перерозподіленого діапазону (R/S аналіз) є чутливим методом для виявлення довготривалих кореляцій у випадкових процесах. Ця інформація подається у вигляді одного параметра: показника Херста ‘‘H’’ [18].

Заданий часовий ряд $X(t)$, визначений у дискретні моменти часу $t = v$, $2v$, $3v$, …, kv. Середнє значення за період τ (ціле кратне v) визначається як:

$$\langle \mathrm{X} \rangle_{\tau} = \frac{\mathrm{v}}{\tau} \sum_{\mathrm{k}=1}^{\frac{\tau}{\mathrm{v}}} \mathrm{X}\,(\mathrm{kv})$$

Різниця між максимальними та мінімальними значеннями $X(t)$ у інтервалі $[v, T]$ називається діапазоном R:

$$R(\tau) = \max[X(t)]_{v \le t \le \tau} - \min[X(t)]_{v \le t \le \tau}$$

Показник Херста H визначається за допомогою властивості масштабування відношення:

$$\frac{R(\tau)}{S(\tau)} \propto \left(\frac{\tau}{v}\right)^{H},$$

де $S(\tau)$ – це стандартне відхилення, яке розраховується за формулою:

$$S(\tau) = \sqrt{\frac{v}{\tau} \sum_{k=1}^{\frac{\tau}{v}} * [X(kv) - \langle X \rangle_{\tau}]^2}$$

Як було доведено Гарольдом Херстом та Вільямом Феллером, асимптотична поведінка для будь-якого незалежного випадкового процесу зі скінченною дисперсією описується як:

$$\frac{R(\tau)}{S(\tau)} = \left(\frac{\pi}{2v}\tau\right)^{\frac{1}{2}}.$$

Це підтверджує, що $H = 1/2$, що є правдою лише для повністю випадкових процесів. Для реальних даних $H \neq 1/2$, оскільки реальні процеси зазвичай характеризуються довготривалими кореляціями. Для таких процесів масштабування модифікується, і R/S асимптотично задається рівнянням показника Херста.

Значення показника Херста може бути в межах $0 < H < 1$. Значення $H < 0{,}5$ вказує на антиперсистентність, що означає, що низьке значення слідує за високим, а потім низьке значення за високим, тобто ряд постійно коливається, але не є випадковим. Значення, дуже близькі до 0, є характерними для вираженої антиперсистентності, а значення, близькі до 0,5, є характерними для слабкої антиперсистентності. З іншого боку, значення $H > 0{,}5$ вказує на персистентність, а значення, близькі до 1, вказують на високу стійкість. Для R/S аналізу та розрахунку показника Херста був використаний алгоритм, що знаходиться у пакеті Python nolds.

2.5 Фрактальна розмірність Хігучі

Фрактальна розмірність часового ряду є мірою складності сигналу [19]. Ідея цього методу полягає у тому, що часовий ряд (як двовимірний графік) лежить на рівномірній сітці, і треба дослідити кількість «коробок», необхідних для його покриття. Отже, фрактальна розмірність - це співвідношення між кількістю «коробок», що покривають часовий ряд, та загальною площею графіка. Цей процес відомий як підрахунок коробок (box-counting).

Фрактальну розмірність можна також можна описати як міру заповнення

простору часового ряду, яка вказує, як по-різному масштабується фрактал залежно від простору. Фрактальна розмірність може бути дробовим числом, а також вона тісно пов'язана з показником Херста. Фрактальна розмірність самоподібного часового ряду може мати значення $1 < D < 2$. Значення, близьке до 1, є типовим для часових рядів з дуже малою кількістю коливань, тобто дуже простого і нескладного сигналу. Значення, близьке до 2, є типовим для дуже складних сигналів. Значення приблизно 1.5 є індикатором випадкового сигналу. Ми використовували алгоритм, розроблений Т. Хігучі, для розрахунку фрактальної розмірності часового ряду. Для обчислення використовувалась вже наявна реалізація на мові Python з пакету hfda.

Пораховані для кожного часового ряду показник Херста та фрактальну розміреність можна знайти у табл. 2.1.

Таблиця 2.1 – Показники комплексності часових рядів

Набір даних	*Показник Херста*	*Фрактальна розмірність*
1, початкові дані	0,1693	1,4892
1, фрактально інтерпольовані дані	0,8515	1,4667
2, початкові дані	0,7860	1,4436
2, фрактально інтерпольовані дані	0,8284	1,3576
3, початкові дані	0,2315	1,9544
3, фрактально інтерпольовані дані	0,7332	1,4703
4, початкові дані	0,7907	1,3842
4, фрактально інтерпольовані дані	0,8864	1,3241

2.6 Мережі з довгою короткочасною пам’яттю

Для аналізу часових рядів було застосовано нейронну мережу з довгою короткочасною пам'яттю (LSTM) [20]. Кожний часовий ряд був розділений на дві частини: одну для навчання моделі та одну для перевірки її ефективності на раніше невідомих даних. Частина для навчання становила дві третини, а невідома частина - останню третину даних в хронологічному порядку.

LSTM-мережі належать до категорії рекурентних нейронних мереж (RNN). RNN – це тип штучних нейронних мереж, що призначений для обробки послідовних даних. Вони використовуються для завдань, де важливі контекстуальні залежності, наприклад, для прогнозування часових рядів або обробки природної мови. RNN мають зворотні зв'язки, що дозволяють інформації передаватися мережею в зворотньому напрямку. Ця особливість дозволяє RNN зберігати інформацію про попередні стани (рис. 2.5).

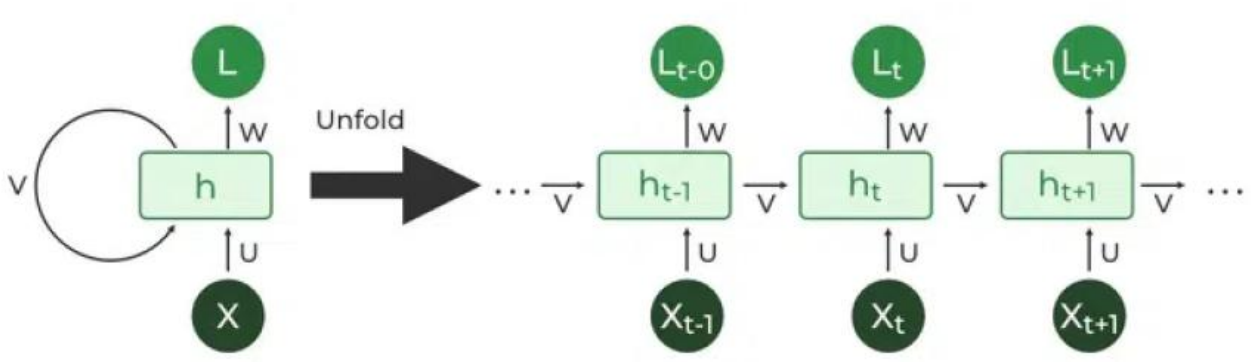

Рисунок 2.5 – Рекурентна нейронна мережа

Хоча стандартні рекурентні нейронні мережі (RNN) й призначені для аналізу та прогнозування часових рядів, вони все ж мають проблему того, що певні вхідні дані, проходячи через рекурентні зв’язки, можуть згасати або експоненційно зростати. LSTM-мережі були спеціально розроблені для вирішення цієї проблеми, а також для вивчення вбудованих довгострокових

залежностей. У складі LSTM-мереж є компонент (блок пам'яті), що використовується для покращення їхньої здатності моделювати довгострокові залежності. Блок пам'яті є рекурентною підмережею, яка містить два функціональні модулі: комірку пам’яті та відповідні вентилі (gates). Завдання комірки пам’яті - запам'ятовувати тимчасовий стан нейронної мережі. Вентилі, у свою чергу, відповідають за контроль потоку інформації та складаються з мультиплікативних одиниць. Існує три типи вентилів: входовий вентиль (input gate), виходовий вентиль (output gate) та забувальний вентиль (forget gate). Входовий вентиль контролює потік інформації в комірку, де забувальний вентиль відповідає за те, як інформація залишається в комірці пам’яті. Виходовий вентиль контролює, скільки інформації використовується для активації виходу, тим самим вирішуючи, що буде повернено решті нейронної мережі.

Була використана базова архітектура нейронної мережі LSTM з одним dense-layer та різною кількістю прихованих шарів. Ми не застосовували регуляризатори та інші зміни для dense-layer. Алгоритм був оптимізований методом проб та помилок зі спостереженням за кривою втрат під час навчання та загальною продуктивністю. В якості оптимізатора було використано Adam. Для функції втрат використовувалася середньоквадратична помилка.

Висновки до розділу 2

У розділі 2 було досліджено метод фрактальної інтерполяції часових рядів та важливість його застосування для покращення роботи рекурентних нейронних мереж глибокого навчання. У першому підпункті було детально описано чотири набори даних для проєкту. У другому підпункті було математично обґрунтовано метод фрактальної інтерполяції, а у третьому підпункті – показано результати його використання на наших наборах даних.

Далі було детально описано та досліджено показники комплексності часових рядів: показник Херста та фрактальна розмірeність. Ці показники ми порівняли для кожного набору даних – початкового та інтерпольованого та записали у таблицю. В останньому пункті було описано проблематику використання рекурентних нейронних мереж для прогнозування часових рядів, а також спосіб її вирішення за допомогою фрактальної інтерполяції даних.

РОЗДІЛ 3 ОПИС ПРОГРАМНОГО ПРОДУКТУ

3.1 Обґрунтування вибору мови програмування та бібліотек

У цьому проєкті мовою програмування було обрано Python через його читабельність та простоту взаємодії з кодом [21]. Більше того, Python надає доступ до багатьох бібліотек та фреймворків для роботи зі штучним інтелектом та машинним навчанням. Його гнучкість, стабільність, незалежність від платформи та велике співтовариство робить його однією з найбільш популярних мов програмування. Python також має велику кількість різноманітних бібліотек для обробки даних, візуалізації, обробки природної мови та багато іншого, що робить його ідеальним вибором для проєктів, пов'язаних з аналізом даних та штучним інтелектом.

Для розробки нейронної мережі була використана відкрита бібліотека для машинного навчання TensorFlow [22]. Ця бібліотека, розроблена компанією Google, надає зручний інтерфейс для побудови та взаємодії з різними типами нейронних мереж, включно з LSTM, що була використана у цій роботі. TensorFlow допомагає проводити ефективні розрахунки на графічних процесорах, що, у свою чергу, прискорює процес навчання моделі. Більше того, бібліотека TensorFlow має широку спільноту користувачів, а також регулярно оновлюється, впроваджуючи нові функції, що робить її надійним інструментом для розробки нейронних мереж.

Використання бібліотеки Pandas для обробки даних було зумовлено її потужними інструментами для роботи з табличними даними та їх аналізу [23]. Pandas – це відкрита бібліотека мови програмування Python, яка забезпечує високоефективні, прості у використанні структури даних і функції для роботи з метаданими. Вона дозволяє легко завантажувати, очищати, обробляти та аналізувати великі обсяги даних. Зокрема, Pandas надає зручний інтерфейс для роботи з даними в табличному форматі, підтримує операції зведення, фільтрації, групування та об'єднання, що значно спрощує підготовку даних для

машинного навчання. Завдяки своїй гнучкості та можливості інтеграції з іншими бібліотеками Python, такими як NumPy та Matplotlib, Pandas є незамінним інструментом для ефективної обробки та аналізу даних у цьому проєкті.

NumPy – це бібліотека мови програмування Python, яка забезпечує підтримку великих багатовимірних масивів і матриць, разом із численними високоефективними математичними функціями для їх обробки [24]. NumPy пропонує широкі можливості для виконання елементних операцій, лінійної алгебри, статистики та інших наукових обчислень, що робить її ключовим інструментом для числового аналізу. Бібліотека значно покращує продуктивність обчислень завдяки реалізації на основі оптимізованого коду C. Завдяки своїй сумісності з іншими бібліотеками для обробки та аналізу даних, такими як Pandas і TensorFlow, NumPy є фундаментальною частиною екосистеми Python для наукових обчислень і машинного навчання, що дозволяє ефективно підготовляти, обробляти та аналізувати числові дані.

Для візуалізації даних було використано бібліотеку Matplotlib [25]. Matplotlib - це відкрита бібліотека мови програмування Python, яка забезпечує гнучкий та зручний інтерфейс для побудови двовимірних графіків. Вона підтримує широкий спектр графічних форматів, таких як лінійні графіки, гістограми, кругові діаграми, коробкові діаграми, теплові карти та багато іншого, що дозволяє ефективно візуалізувати результати аналізу даних. Завдяки своїй гнучкості, Matplotlib дозволяє налаштовувати всі аспекти графіків, включаючи стиль, кольори, осі, підписи та легенди. Крім того, бібліотека інтегрується з іншими інструментами Python, такими як NumPy та Pandas, що спрощує процес побудови графіків на основі даних, оброблених за допомогою цих бібліотек. Matplotlib є незамінним інструментом для ефективної візуалізації та аналізу даних у моєму проєкті, оскільки надає можливість наочно представляти результати та взаємозв’язки між різними змінними.

Jupyter Notebook було використано як платформу для роботи з проєктом завдяки його інтерактивності та зручності [26]. Jupyter Notebook - це відкритий

веб-застосунок, який дозволяє створювати та ділитися документами, що містять живий код, рівняння, візуалізації та текстові описи. Ця платформа підтримує понад 40 мов програмування, включаючи Python, і забезпечує зручний інтерфейс для написання, виконання та налагодження коду в реальному часі. Jupyter Notebook особливо корисний для експериментування з кодом, проведення аналізу даних, створення інтерактивних візуалізацій та документування процесу досліджень. Завдяки можливості інтеграції з такими бібліотеками, як NumPy, Pandas, Matplotlib та TensorFlow, Jupyter Notebook стає потужним інструментом для розробки, тестування та демонстрації моделей машинного навчання.

Дані для роботи були взяті з офіційного сайту статистики Великої Британії, National Office for Statistics [27]. Цей ресурс надає достовірні та всебічні статистичні дані, що охоплюють різні аспекти економіки, соціального розвитку, здоров'я та демографії. Завдяки високій якості та актуальності наданої інформації, дані з National Office for Statistics слугують надійною основою для аналізу та моделювання. Використання цих даних дозволяє забезпечити точність та достовірність результатів, а також сприяє проведенню обґрунтованих висновків і прийняттю рішень, базованих на реальних статистичних показниках.

3.2 Будова програми

Програма складається з декількох частин, кожна з яких відповідає за окрему операцію чи функцію. Спочатку програма імпортує бібліотеки для математичних обчислень, структуризації даних, створення нейронних мереж та візуалізації результатів. Також в застосунку використовується бібліотека для знаходження фрактальної розмірeності, збереження та завантаження натренованої моделі

Далі написано код для завантаження даних з файлу Visitors.xls, в якому міститься інформація про щомісячний обсяг туристів. Ці дані оброблено, перш за все видалено інформацію за час пандемії, адже вона погано впливає на передбачення моделі в період без негативного впливу на туризм, також реалізовано функції для нормалізації та стандартизації даних. З цієї вибірки даних за допомогою фрактальної інтерполяції створено більший часовий ряд, в даному дослідженні буде порівняно моделі натреновані на цих та звичайних даних. Для створення прикладів для тренування нейронної мережі часовий ряд поділено на менші відрізки довжиною по 60 записів з кроком зміщення в один запис.

В програмному коді реалізована функція компіляції LSTM моделі з такими параметрами: кількість нейронів в шарі, кількість шарів, функція активації та ймовірність DropOut. Для визначення оптимальних параметрів для тренування нейронної мережі натреновано моделі з різними параметрами для їх порівняння, створено графіки для візуалізації результатів. Узгоджено такі параметри моделей:

- n_a = 200;
- drop_prob = 0.1;
- n_layers = 3;
- activation = linear.

Оскільки нейронна мережа показує кращий результат на більшій кількості даних, для роботи було обрано саме 3 шари. Використано лінійну функцію активації, адже це задача регресії.

Фінальний блок програмного продукту присвячений візуалізації результатів. Для кожної моделі створено графік історії середньої абсолютної похибки тренування та проведено порівняння передбачених даних з реальними.

3.3 Результати роботи додатка

Тренування моделі, яка навчалась на звичайних даних тривало близько 250 секунд, в той час як тренування моделі з фрактально інтерпольованими даними тривало близко 1020 секунд. В кожної моделі 808251 параметр. Нижче зображено результати роботи програмного продукту (рис. 3.1 – рис 3.8).

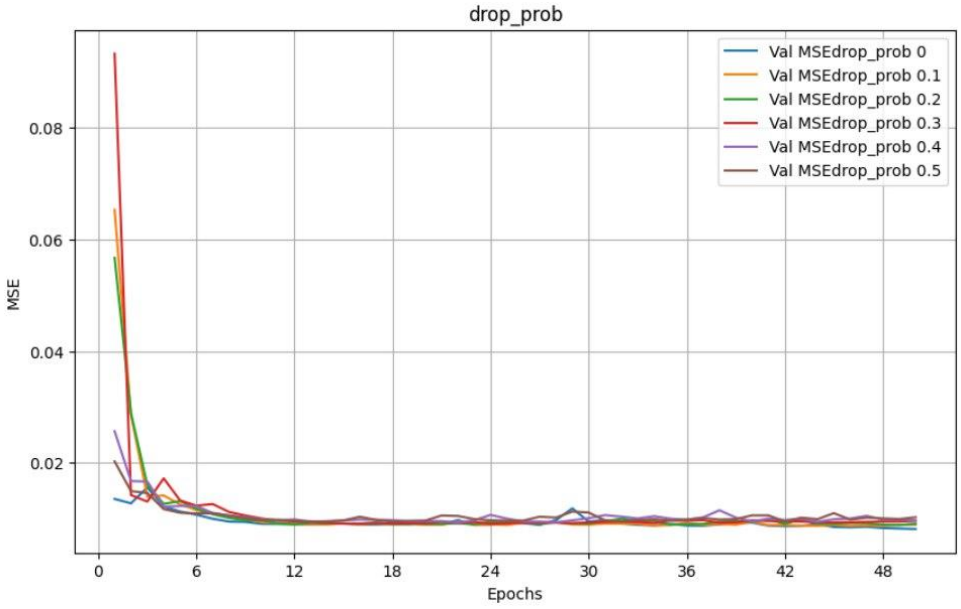

Рисунок 3.1 – Графік порівняння різних значень параметру ймовірності DropOut

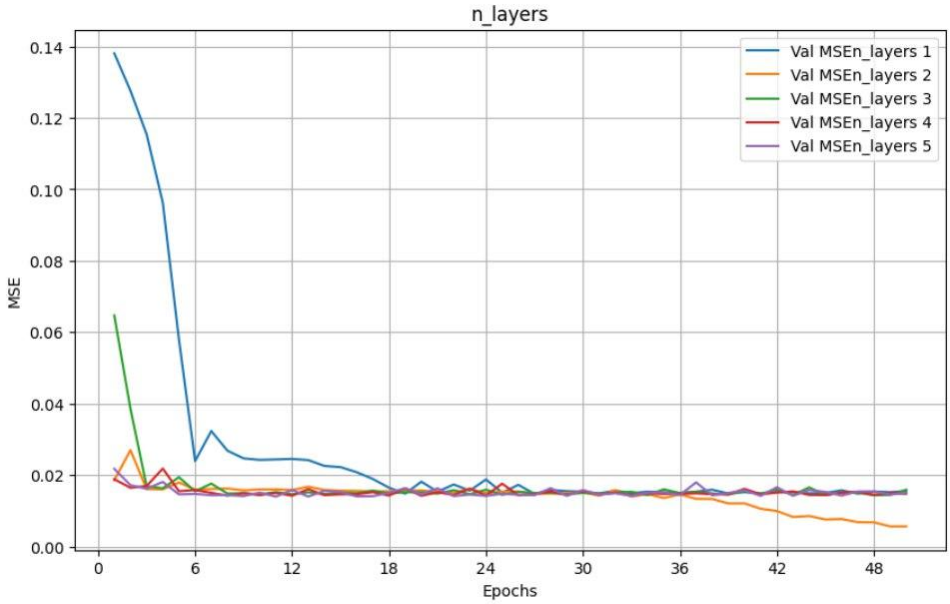

Рисунок 3.2 – Графік порівняння різних значень параметру кількості шарів LSTM

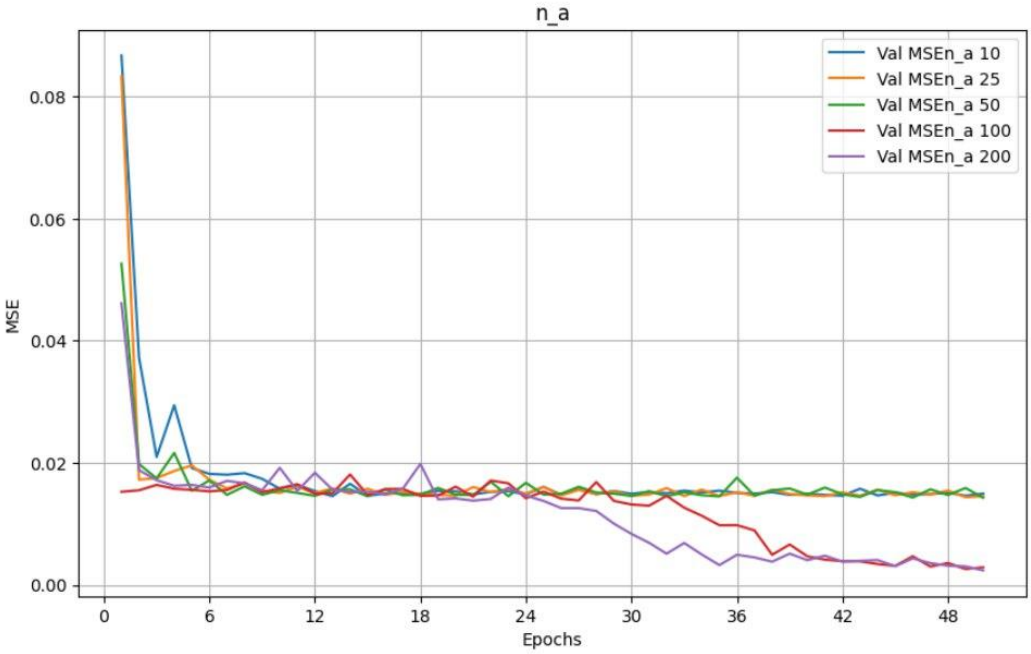

Рисунок 3.3 – Графік порівняння різних значень параметру кількості нейронів в шарі

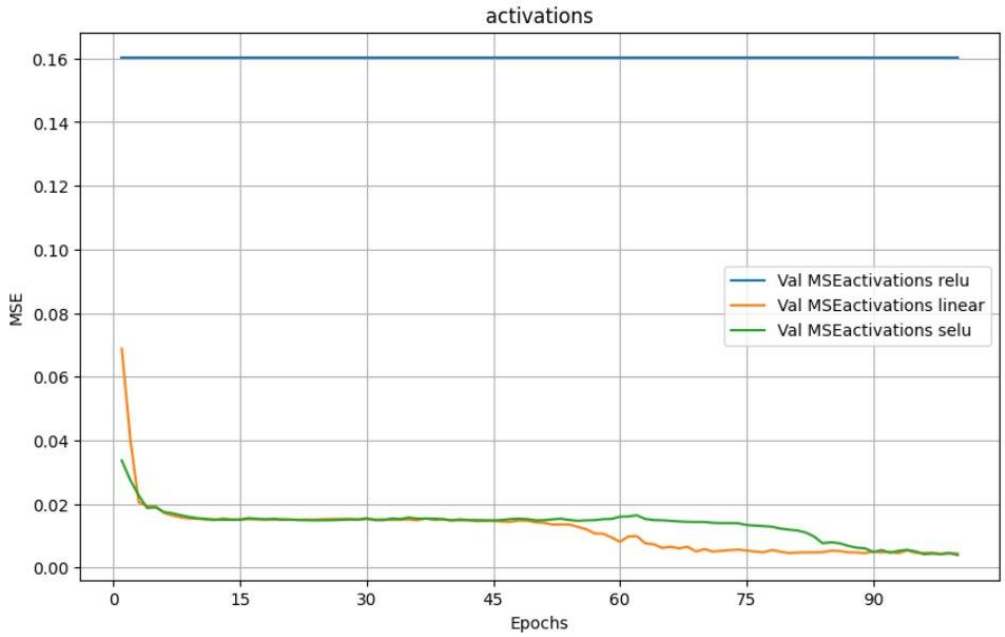

Рисунок 3.4 – Графік порівняння різних значень параметру функції активації

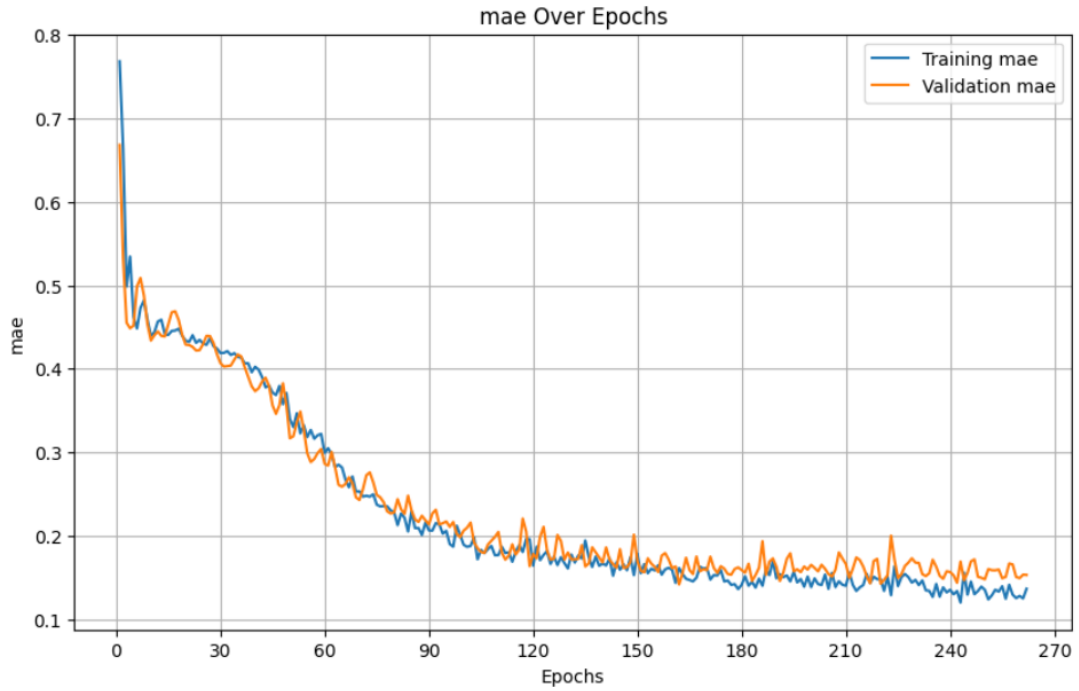

Рисунок 3.5 – Графік MAE для кожної ітерації історії тренування моделі на звичайних даних

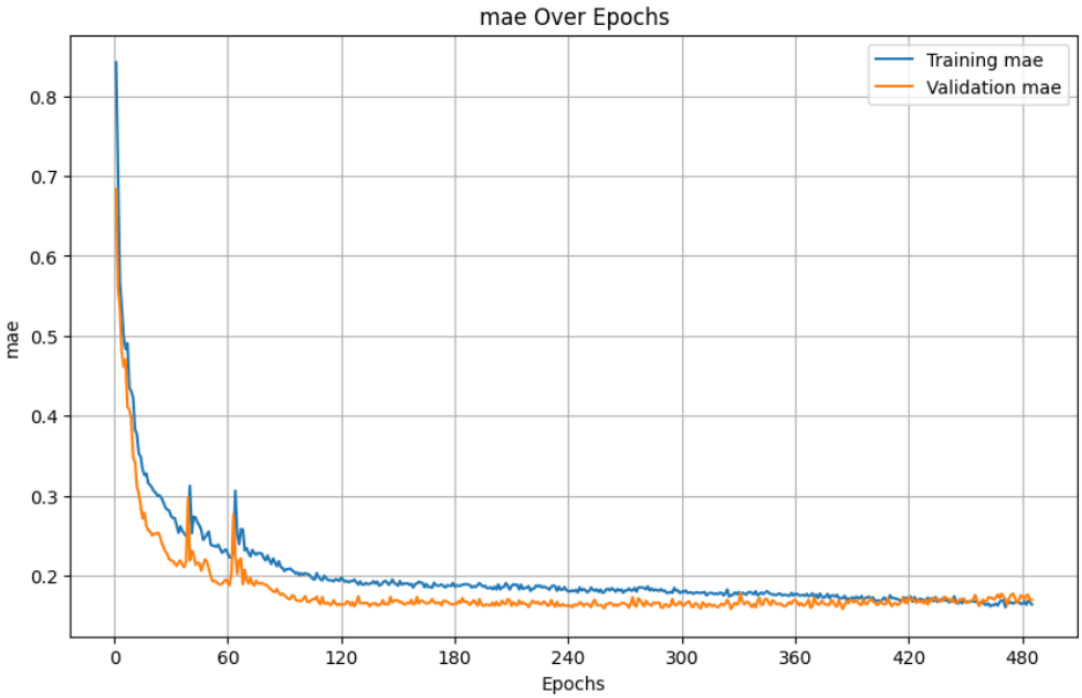

Рисунок 3.6 – Графік MAE для кожної ітерації історії тренування моделі на фрактально інтерпольованих даних

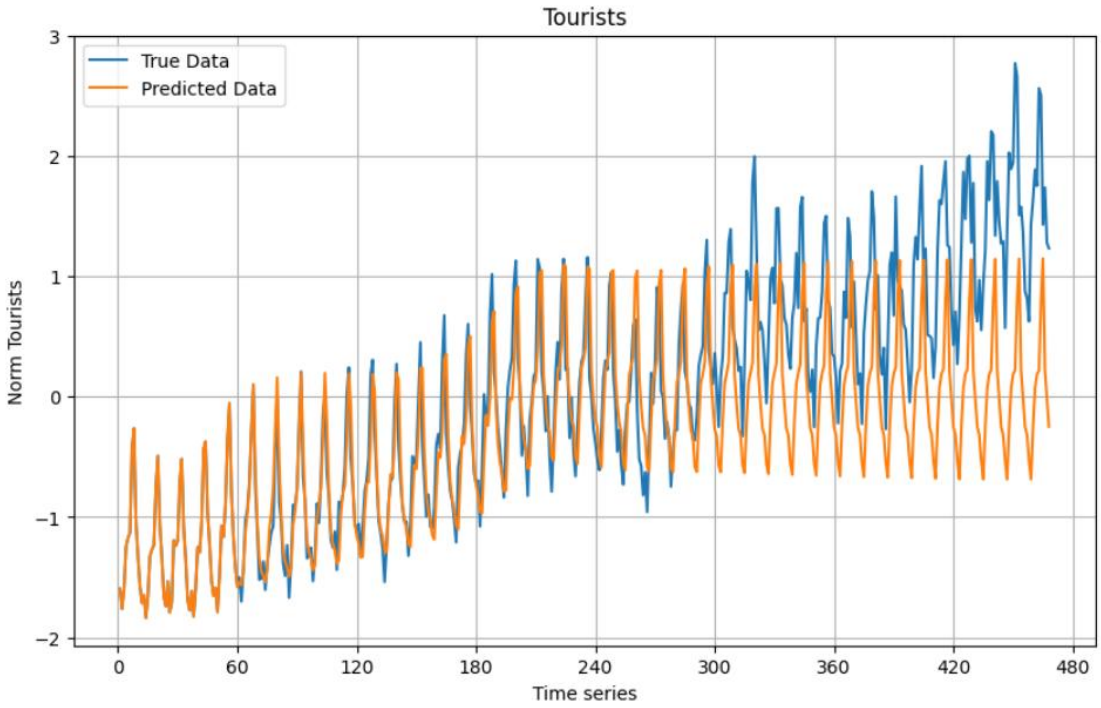

Рисунок 3.7 – Графік порівняння реальних даних з передбаченням моделі навченої на звичайних даних

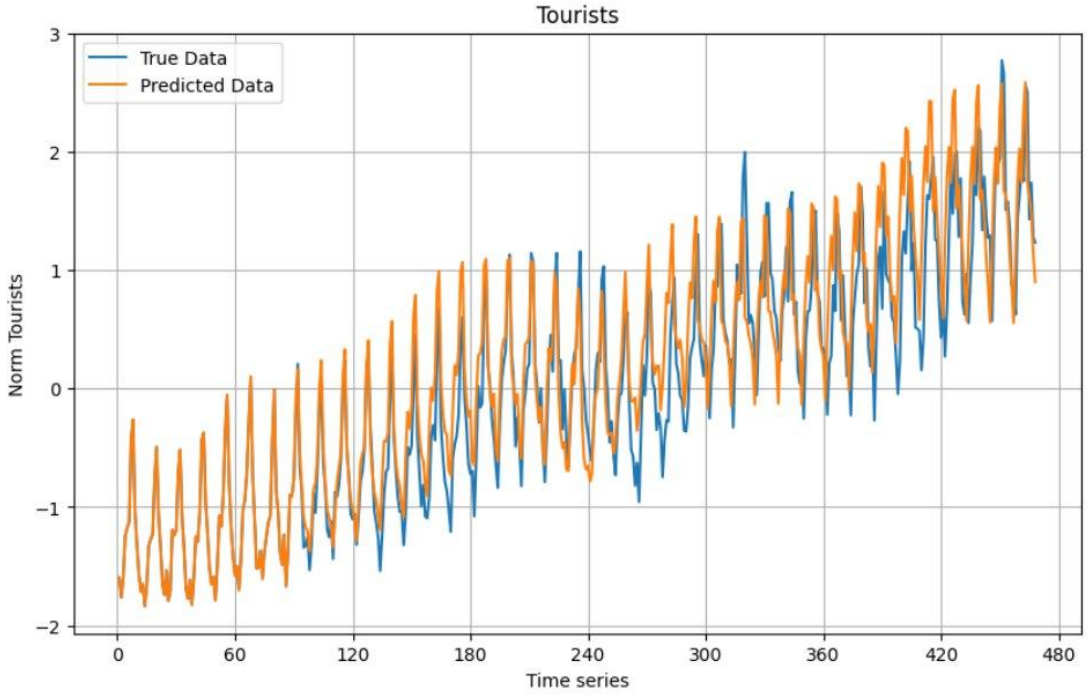

Рисунок 3.8 – Графік порівняння реальних даних з передбаченням моделі навченої на фрактально інтерпольованих даних

Висновки до розділу 3

У третьому розділі було детально описано та обгрунтовано переваги використання мови програмування Python та її бібліотек TensorFlow для роботи з нейронними мережами, Pandas для обробки даних, NumPy для взаємодії з масивами та матрицями та Matplotlib для візуалізації даних. Платформою для роботи використовувався Jupyter Notebook, а часові ряди були отримані з офіційного сайту статистики Великої Британії Office for National Statistics. У роботі було показано та обгрунтовано вибір структури додатка завдяки порівнянню різних параметрів, а також результати його роботи на реальних даних та фрактально інтерпольованих.

РОЗДІЛ 4 ФУНКЦІОНАЛЬНО-ВАРТІСНИЙ АНАЛІЗ ПРОГРАМНОГО ПРОДУКТУ

У цьому розділі ми проведемо оцінку основних характеристик програмного продукту, розробленого для вирішення задач прогнозування стану сфери туризму за допомогою рекурентних нейронних мереж глибокого навчання.

Нижче наведено аналіз різних варіантів реалізації модулю з метою вибору оптимальної, з огляду при цьому як на економічні фактори, так і на характеристики продукту, що впливають на продуктивність роботи і на його сумісність з апаратним забезпеченням. Для цього було використано апарат функціонально-вартісного аналізу.

Функціонально-вартісний аналіз (ФВА) – це технологія, яка дозволяє оцінити реальну вартість продукту або послуги незалежно від організаційної структури компанії. ФВА проводиться з метою виявлення резервів зниження витрат за рахунок ефективніших варіантів виробництва, кращого співвідношення між споживчою вартістю виробу та витратами на його виготовлення. Для проведення аналізу використовується економічна, технічна та конструкторська інформація.

Алгоритм функціонально-вартісного аналізу включає в себе визначення послідовності етапів розробки продукту, визначення повних витрат (річних) та кількості робочих часів, визначення джерел витрат та кінцевий розрахунок вартості програмного продукту.

4.1 Постановка задачі проєктування

У роботі застосовується метод ФВА для проведення техніко-економічного аналізу розробки системи підтримки прийняття рішень для в'їзного туризму Великої Британії. Фактичний аналіз - це аналіз функцій програмного продукту призначеного для того, щоб проаналізувати доцільність використання нейронних мереж з самонавчанням для прогнозування часових рядів у сфері туризму.

Вимоги до ПП (програмного продукту):

- функціонування на персональних комп'ютерах із стандартним набором компонентів;
- швидкість обробки даних та доступ до інформації в реальному часі;
- точність прогнозування;
- мінімальні витрати на впровадження програмного продукту.

4.2 Обґрунтування функцій програмного продукту

Головна функція F_0 – розробка програмного продукту, призначеного для того, щоб проаналізувати доцільність використання нейронних мереж з самонавчанням у сфері туризму. Це основна функція, але також можна виділити такі:

- F_1 – вибір мови програмування;
- F_2 – вибір фреймворку для створення нейронних мереж;
- F_3 – вибір середовища розробки.

В кожної з цих функцій є декілька варіантів реалізації.

Функція F_1:

– Python;

– Java.

Функція F_2:

– Tensorflow;

– Torch.

Функція F_3:

– Jupyter Notebook;

– Visual Studio Code.

Морфологічну карту системи з варіантами реалізації основних функцій зображено на рис. 4.1.

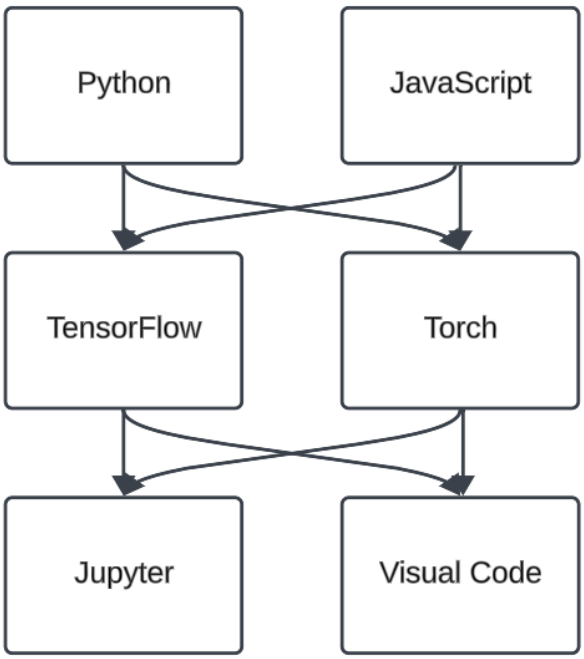

Рисунок 4.1 – Морфологічна карта

В морфологічній карті відображається множина всіх можливих варіантів основних функцій.

Побудуємо позитивно-негативну матрицю варіантів основних функцій (табл. 4.1). Нам варто відкинути деякі варіанти, адже вони не задовольняють потреби програмного продукту. Ці варіанти відзначені у морфологічній карті.

Таблиця 4.1 – Позитивно-негативна матриця

Функції	*Варіанти реалізації*	*Переваги*	*Недоліки*
F_1	*А*	Швидка розробка програми, доступність бібліотек, кросплатформеність	Відносно низька швидкість роботи, особливо на великій кількості даних
	Б	Швидкість виконання коду	Більше часу витрачається для розробки програми
F_2	*А*	Надійно працює зі складними проєктами	Не підтримується багатьма мовами
	Б	Добре працює на малих проєктах	Погано працює з великим об’ємом даних
F_3	*А*	Підтримується багатьма мовами програмування, легко запускається на будь-якому сервері	Відсутня можливість роботи без інтернету
	Б	Багато інструментів, безпечна	Підтримує одночасно лише одну мову програмування

Функція F_1.

Перевагу надаємо швидкості вивчення, простоті використання та наявності стандартних бібліотек для обчислення. Для спрощення роботи по написанню коду варіант Б має бути відкинутий.

Функція F_2.

Обидва варіанти використовуються для розробки нейронних мереж, тому можна використати варіант А чи Б.

Функція F_3.

Віддаємо перевагу варіанту А в разі вибору мови програмування Python.

Таким чином, будемо розглядати такий варіант реалізації ПП:

$$F_1\text{а} - F_2\text{а} - F_3\text{а},$$

$$F_1\text{а} - F_2\text{б} - F_3\text{а}.$$

Для оцінювання якості розглянутих функцій обрана система параметрів, описана нижче.

4.3 Обгрунтування системи параметрів ПП

На основі даних, розглянутих вище, визначаються основні параметри вибору, які будуть використані для розрахунку коефіцієнта технічного рівня.

Для того, щоб охарактеризувати програмний продукт, будемо використовувати наступні параметри:

– *X*1 – швидкодія мови програмування;

– *X*2 – об’єм пам’яті для обчислень та збереження даних;

– *X*3 – час навчання даних;

– *X*4 – потенційний об’єм програмного коду.

Гірші, середні і кращі значення параметрів вибираються на основі вимог замовника й умов, що характеризують експлуатацію ПП як показано у табл. 4.2.

Таблиця 4.2 – Основні параметри програмного продукту

Назва параметра	*Умовні позначення*	*Одиниці виміру*	*Значення параметра*		
			гірші	середні	кращі
Швидкодія мови програмування	*X*1	оп/мс	10000	16000	19000
Об’єм пам’яті	*X*2	Мб	40	20	10
Тривалість навчання моделі	*X*3	с	3000	2500	1800
Приблизний об’єм програмного коду	*X*4	кількість рядків коду	350	260	220

За даними табл. 4.2 будуються графічні характеристики параметрів – рис. 4.2 – рис. 4.5.

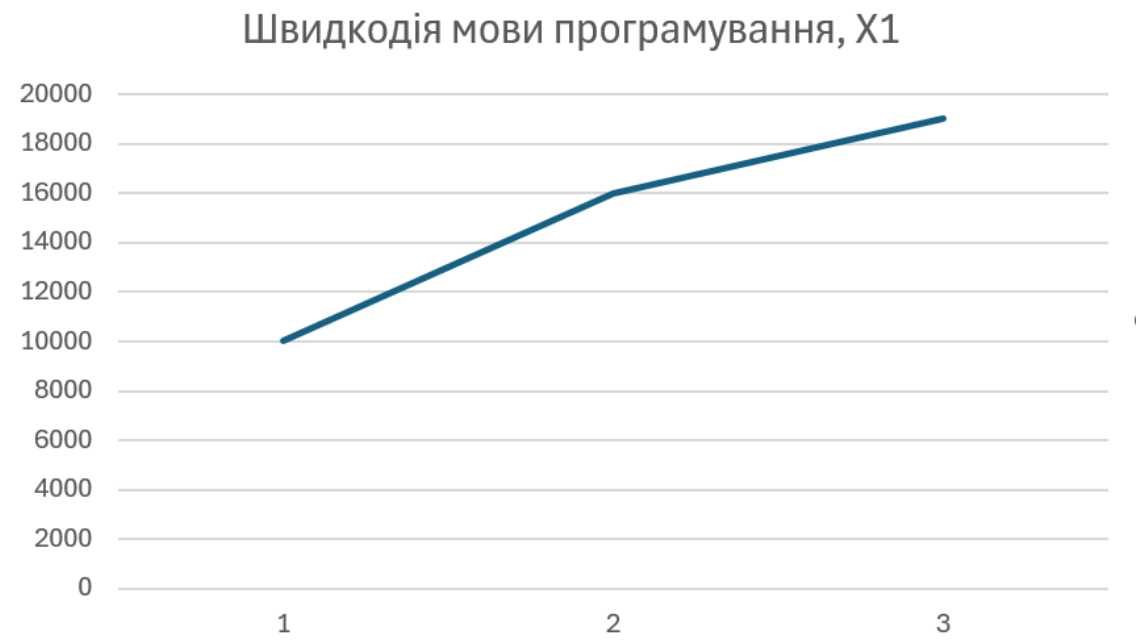

Рисунок 4.2 – X1, швидкодія мови програмування

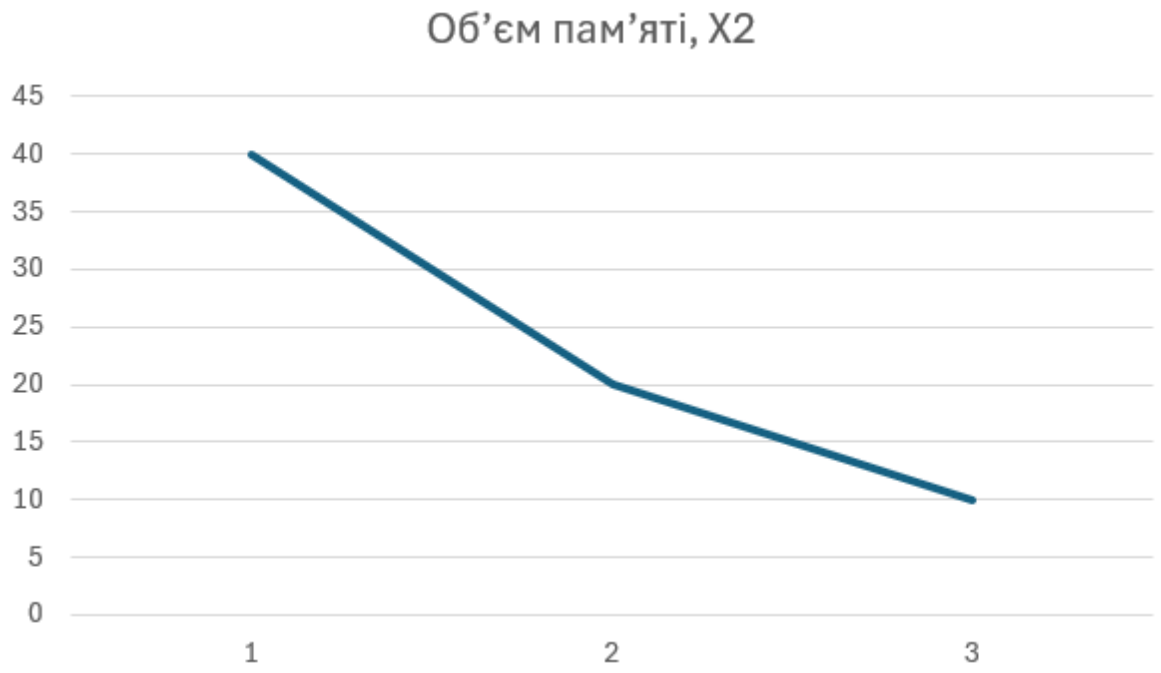

Рисунок 4.3 – X2, об’єм пам’яті

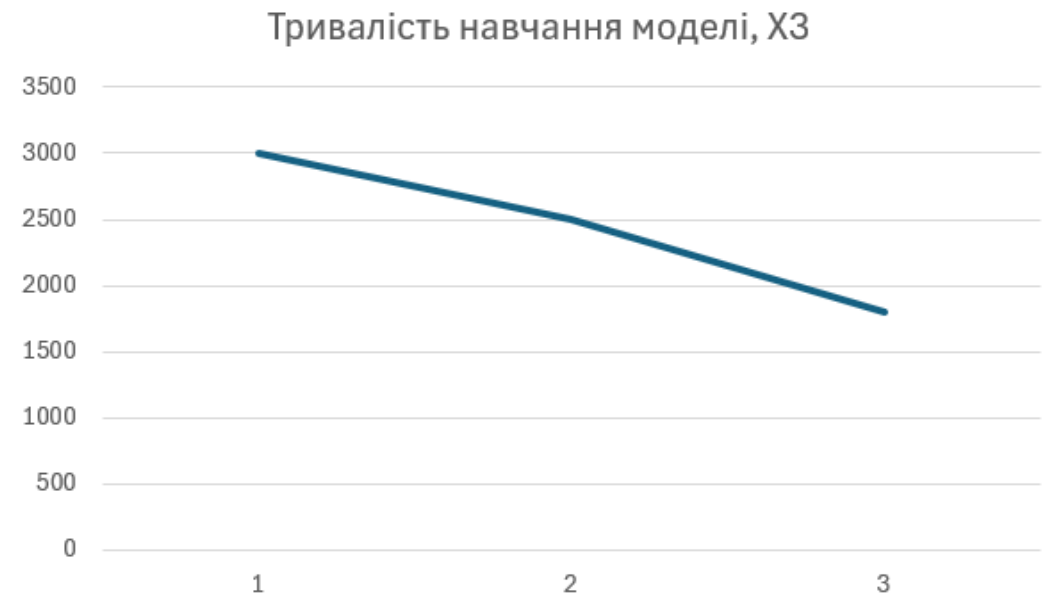

Рисунок 4.4 – X3, тривалість навчання моделі

Рисунок. 4.5 – X4, об’єм програмного коду

4.4 Аналіз експертного оцінювання результатів

Після детального обговорення й аналізу кожний експерт оцінює ступінь важливості кожного параметру для конкретно поставленої цілі – розробка програмного продукту, який дає найбільш точні результати при знаходженні параметрів моделей адаптивного прогнозування і обчислення прогнозних значень.

Значимість кожного параметра визначається методом попарного порівняння. Оцінку проводить експертна комісія із 7 людей. Визначення коефіцієнтів значимості передбачає:

- визначення рівня значимості параметра шляхом присвоєння різних рангів;
- перевірку придатності експертних оцінок для подальшого використання;
- визначення оцінки попарного пріоритету параметрів;
- обробку результатів та визначення коефіцієнту значимості.

Результати експертного ранжування наведені у табл. 4.3.

Для перевірки степені достовірності експертних оцінок, визначимо наступні параметри:

а) сума рангів кожного з параметрів і загальна сума рангів:

$$R_i = \sum_{j=1}^{N} r_{ij} R_{ij} = \frac{Nn(n+1)}{2} = 70,$$

де N – число експертів,

n – кількість параметрів;

б) середня сума рангів:

$$T = \frac{1}{n} R_{ij} = 17{,}5.$$

в) відхилення суми рангів кожного параметра від середньої суми рангів:

$$\Delta_i = R_i - T.$$

Таблиця 4.3 – Результати ранжування параметрів

Позначення параметра	*Назва параметра*	*Одиниці виміру*	*Ранг параметра за оцінкою експерта*							*Сума рангів* R_i	*Відхилення* Δ_i	Δ_i^2
			1	2	3	4	5	6	7			
X1	Швидкодія мови програмування	оп/мс	2	3	4	3	3	3	2	20	2,5	6,25
X2	Об’єм пам’яті	Кб	3	1	2	1	1	2	3	13	-4,5	20,25
X3	Тривалість навчання моделі	с	4	4	3	4	4	4	4	27	9,5	90,25
X4	Приблизний об’єм програмного коду	Кількість рядків коду	1	2	1	2	2	1	1	10	-7,5	56,25
	Разом		10	10	10	10	10	10	10	70	0	173

Сума відхилень по всім параметрам повинна дорівнювати 0;

г)загальна сума квадратів відхилення:

$$S = \sum_{i=1}^{N} \Delta_i^2 = 173.$$

Порахуємо коефіцієнт узгодженості:

$$W = \frac{12S}{N^2(n^3 - n)} = \frac{12 \cdot 173}{7^2(4^3 - 4)} = 0{,}706 > W_k = 0{,}67.$$

Ранжування можна вважати достовірним, тому що знайдений коефіцієнт узгодженості перевищує нормативний, котрий дорівнює 0,67.

Скориставшись результатами ранжирування, проведемо попарне порівняння всіх параметрів і результати занесемо у табл. 4.4.

Таблиця 4.4 – Попарне порівняння параметрів

Параметри	*Експерти*							*Кінцева оцінка*	*Числове значення*
	1	2	3	4	5	6	7		
*X*1 і *X*2	<	>	>	>	>	>	<	>	1,5
*X*1 і *X*3	<	<	>	<	<	<	<	<	0,5
*X*1 і *X*4	>	>	>	>	>	>	>	>	1,5
*X*2 і *X*3	<	<	<	<	<	<	<	<	0,5
*X*2 і *X*4	>	<	>	<	<	>	>	>	1,5
*X*3 і *X*4	>	>	>	>	>	>	>	>	1,5

Числове значення, що визначає ступінь переваги *i*–го параметра над *j*–тим, a_{ij} визначається по формулі:

$$a_{ij} = \begin{cases} 1.5 \text{ при } X_i > X_j \\ 1.0 \text{ при } X_i = X_j. \\ 0.5 \text{ при } X_i < X_j \end{cases}$$

З отриманих числових оцінок переваги складемо матрицю $A = \| a_{ij} \|$.

Для кожного параметра зробимо розрахунок вагомості $K_{вi}$ за наступними формулами:

$$K_{\text{вi}} = \frac{b_i}{\sum_{i=1}^{n} b_i},$$

$$b_i = \sum_{i=1}^{N} a_{ij}.$$

Відносні оцінки розраховуються декілька разів доти, поки наступні значення не будуть незначно відрізнятися від попередніх (менше 2%). На другому і наступних кроках відносні оцінки розраховуються за наступними формулами:

$$K_{\text{ві}} = \frac{b_i'}{\sum_{i=1}^{n} b_i'},$$

$$b_i' = \sum_{i=1}^{N} a_{ij} b_j.$$

Як видно з табл. 4.5, різниця значень коефіцієнтів вагомості не перевищує 2%, тому більшої кількості ітерацій не потрібно.

Таблиця 4.5 – Розрахунок вагомості параметрів

Параметри x_i	*Параметри* x_j				*Перша ітер.*		*Друга ітер.*		*Третя ітер.*	
	X1	X2	X3	X4	b_i	$K_{\text{ві}}$	b_i^1	$K_{\text{ві}}^1$	b_i^2	$K_{\text{ві}}^2$
*X*1	1	1,5	0,5	1,5	4,5	0,28	16,25	0,28	59,125	0,27
*X*2	0,5	1	0,5	1,5	3,5	0,22	12,25	0,21	44,875	0,21
*X*3	1,5	1,5	1	1,5	5,5	0,34	21,25	0,36	77,875	0,36
*X*4	0,5	0,5	0,5	1	2,5	0,16	9,25	0,15	34,125	0,16
Всього:					16	1	59	1	216	1

4.5 Аналіз рівня якості варіантів реалізації функцій

Визначаємо рівень якості кожного варіанту виконання основних функцій окремо.

Абсолютні значення параметрів *X*2 (об'єм пам'яті), *X*3 (тривалість тренування моделі) та *X*4 (приблизний об'єм програмного коду) відповідають технічним вимогам умов функціонування даного ПП.

Абсолютне значення параметра *X*1 (швидкість роботи мови програмування) обрано не найгіршим.

Коефіцієнт технічного рівня для кожного варіанта реалізації ПП розраховується так (табл. 4.6):

$$K_K(j) = \sum_{i=1}^{n} K_{вi,j} B_{i,j} ,$$

де n – кількість параметрів,

$K_{вi}$ – коефіцієнт вагомості i–го параметра,

B_i – оцінка i–го параметра в балах.

Таблиця 4.6 – Розрахунок показників рівня якості варіантів реалізації основних функцій ПП

Основні функції	*Варіант реалізації функції*	*Параме-три*	*Абсолютне значення параметра*	*Бальна оцінка параметра*	*Коефіцієнт вагомості параметра*	*Коефіцієнт рівня якості*
*F*1	А	*X*1	10000	2	0,27	0,54
*F*2	А	*X*4	260	6	0,16	0,96
	Б	*X*4	300	3	0,16	0,48
*F*3	Б	*X*3	1300	10	0,36	3,6

За даними з табл. 4.6 за формулою:

$$K_K = K_{\text{ТУ}}[F_{1k}] + K_{\text{ТУ}}[F_{2k}] + \ldots + K_{\text{ТУ}}[F_{zk}],$$

визначаємо рівень якості кожного з варіантів:

$$K_{K1} = 0{,}54 + 0{,}96 + 3{,}6 = 5{,}1.$$

$$K_{K2} = 0{,}54 + 0{,}48 + 3{,}6 = 4{,}62.$$

Як видно з розрахунків, кращим є 1 варіант, для якого коефіцієнт технічного рівня має найбільше значення.

4.6 Економічний аналіз варіантів розробки ПП

Для визначення вартості розробки ПП спочатку проведемо розрахунок трудомісткості.

Усі варіанти охоплюють два окремих завдання:

1. Розробка проєкту програмного продукту.
2. Розробка програмної оболонки.

Завдання 1 за ступенем новизни відноситься до групи А, завдання 2 – до групи Б. За складністю алгоритми, які використовуються в завданні 1 належать до групи 1; а в завданні 2 – до групи 3.

Для реалізації завдання 1 використовується довідкова інформація, а завдання 2 використовує інформацію у вигляді даних.

Проведемо розрахунок норм часу на розробку та програмування для кожного з завдань.

Загальна трудомісткість обчислюється як:

$$T_{\text{О}} = T_{\text{Р}} \cdot K_{\text{П}} \cdot K_{\text{СК}} \cdot K_{\text{М}} \cdot K_{\text{СТ}} \cdot K_{\text{СТ.М}},$$

де $T_{\text{Р}}$ – трудомісткість розробки ПП,

$K_{\text{П}}$ – поправочний коефіцієнт,

$К_{СК}$ – коефіцієнт на складність вхідної інформації,

$К_{М}$ – коефіцієнт рівня мови програмування,

$К_{СТ}$ – коефіцієнт використання стандартних модулів і прикладних програм,

$К_{СТ.М}$ – коефіцієнт стандартного математичного забезпечення.

Для першого завдання, виходячи із норм часу для завдань розрахункового характеру степеню новизни А та групи складності алгоритму 1, трудомісткість дорівнює: $Т_{Р}$ = 70 людино-днів. Поправочний коефіцієнт, який враховує вид нормативно-довідкової інформації для першого завдання: $К_{П}$ = 1,7. Поправочний коефіцієнт, який враховує складність контролю вхідної та вихідної інформації для всіх семи завдань рівний 1: $К_{СК}$ = 1. Оскільки при розробці першого завдання використовуються стандартні модулі, врахуємо це за допомогою коефіцієнта $К_{СТ}$ = 0,9. Тоді загальна трудомісткість програмування першого завдання дорівнює:

$$Т_1 = 70 \cdot 1,7 \cdot 0,9 = 107,1 \text{ людино-днів.}$$

Проведемо аналогічні розрахунки для подальших завдань.

Для другого завдання (використовується алгоритм третьої групи складності, степінь новизни Б), тобто $Т_{Р}$ = 28 людино-днів, $К_{П}$ = 0.9, $К_{СК}$ = 1, $К_{СТ}$ = 0.8:

$$Т_2 = 28 \cdot 0.9 \cdot 0,8 = 20,2 \text{ людино-днів.}$$

Складаємо трудомісткість відповідних завдань для кожного з обраних варіантів реалізації програми, щоб отримати їх трудомісткість:

$$Т_{I} = (107,1 + 20,2 + 4,8 + 20,2) \cdot 8 = 1218,4 \text{ людино-годин.}$$

$$Т_{II} = (107,1 + 20,2 + 6,91 + 20,2) \cdot 8 = 1235,3 \text{ людино-годин.}$$

Найбільш високу трудомісткість має варіант II.

У розробці беруть участь два програмісти з окладом 37000 грн, один аналітик в області даних з окладом 32000 грн. Визначимо середню зарплату за годину за формулою:

$$C_{\text{ч}} = \frac{M}{T_m \cdot t} \text{ грн},$$

де M – місячний оклад працівників,

T_m – кількість робочих днів в місяць,

t – кількість робочих годин в день.

$$C_{\text{ч}} = \frac{37000 + 37000 + 32000}{3 \cdot 21 \cdot 8} = 210{,}32 \text{ грн}$$

Тоді, розрахуємо заробітну плату за формулою:

$$C_{\text{ЗП}} = C_{\text{ч}} \cdot T_i \cdot K_{\text{Д}},$$

де $C_{\text{ч}}$ – величина погодинної оплати праці програміста,

T_i – трудомісткість відповідного завдання,

$K_{\text{Д}}$ – норматив, який враховує додаткову заробітну плату.

Зарплата розробників за варіантами становить:

I. $C_{\text{ЗП}} = 210{,}32 \cdot 1218{,}4 \cdot 1{,}2 = 307504{,}67$ грн,

II. $C_{\text{ЗП}} = 210{,}32 \cdot 1235{,}3 \cdot 1{,}2 = 311769{,}96$ грн.

Відрахування на єдиний соціальний внесок становить 20%:

I. $C_{\text{ВІД}} = C_{\text{ЗП}} \cdot 0{,}2 = 307504{,}67 \cdot 0{,}2 = 61500{,}93$ грн,

II. $C_{\text{ВІД}} = C_{\text{ЗП}} \cdot 0{,}2 = 311769{,}96 \cdot 0{,}2 = 62353{,}99$ грн.

Тепер визначимо витрати на оплату однієї машино-години. ($С_М$)

Так як одна ЕОМ обслуговує одного програміста з окладом 37000 грн., з коефіцієнтом зайнятості 0,2 то для однієї машини отримаємо:

$$С_Г = 12 \cdot М \cdot К_З = 12 \cdot 37000 \cdot 0,2 = 88800 \text{ грн.}$$

З урахуванням додаткової заробітної плати:

$$С_{ЗП} = С_Г \cdot (1 + К_З) = 88800 \cdot (1 + 0,2) = 106560 \text{ грн.}$$

Відрахування на соціальний внесок:

$$С_{ВІД} = С_{ЗП} \cdot 0,2 = 106560 \cdot 0,2 = 21312 \text{ грн.}$$

Амортизаційні відрахування розраховуємо при амортизації 25% та вартості ЕОМ – 37000 грн:

$$С_А = К_{ТМ} \cdot К_А \cdot Ц_{ПР} = 1,1 \cdot 0,25 \cdot 37000 = 10175 \text{ грн,}$$

де $К_{ТМ}$ – коефіцієнт, який враховує витрати на транспортування та монтаж приладу у користувача,

$К_А$ – річна норма амортизації,

$Ц_{ПР}$ – договірна ціна приладу.

Витрати на ремонт та профілактику розраховуємо як:

$$С_Р = К_{ТМ} \cdot Ц_{ПР} \cdot К_Р = 1,1 \cdot 37000 \cdot 0,08 = 3256 \text{ грн,}$$

де $К_Р$ – відсоток витрат на поточні ремонти.

Ефективний годинний фонд часу ПК за рік розраховуємо за формулою:

$$T_{ЕФ} = (Д_К – Д_В – Д_С – Д_Р) \cdot t_З \cdot К_В = (365 – 104 – 12 – 16) \cdot 8 \cdot 0,85 = 1584,4 \text{ години},$$

де $Д_К$ – календарна кількість днів у році,

$Д_В$, $Д_С$ – відповідно кількість вихідних та святкових днів,

$Д_Р$ – кількість днів планових ремонтів устаткування,

t – кількість робочих годин в день,

$К_В$ – коефіцієнт використання приладу у часі протягом зміни.

Витрати на оплату електроенергії розраховуємо за формулою:

$$С_{ЕЛ} = T_{ЕФ} \cdot N_С \cdot К_З \cdot Ц_{ЕН} = 1584,4 \cdot 0,2 \cdot 0,2 \cdot 6,32 = 400,54 \text{ грн},$$

де $N_С$ – середньо-споживча потужність приладу,

$К_З$– коефіцієнтом зайнятості приладу,

$Ц_{ЕН}$ – тариф за 1 КВт-годин електроенергії.

Накладні витрати розраховуємо за формулою:

$$С_Н = Ц_{ПР} \cdot 0,67 = 37000 \cdot 0,67 = 24790 \text{ грн}.$$

Тоді, річні експлуатаційні витрати будуть:

$$С_{ЕКС} = С_{ЗП} + С_{ВІД} + С_А + С_Р + С_{ЕЛ} + С_Н,$$

$$С_{ЕКС} = 106560 + 21312 + 10175 + 3256 + 400,54 + 24790 = 166493,54 \text{ грн}.$$

Собівартість однієї машино-години ЕОМ дорівнюватиме:

$$C_{\text{М-Г}} = C_{\text{ЕКС}} / T_{\text{ЕФ}} = 166493{,}54 \ / \ 1584{,}4 = 105{,}08 \text{ грн/год.}$$

Оскільки в даному випадку всі роботи, які пов‘язані з розробкою програмного продукту ведуться на ЕОМ, витрати на оплату машинного часу, в залежності від обраного варіанта реалізації, складає:

$$C_{\text{М}} = C_{\text{М}-\text{Г}} \cdot T,$$

I. $C_{\text{М}} = 105{,}08 \cdot 1218{,}4 = 128029{,}47$ грн,

II. $C_{\text{М}} = 105{,}08 \cdot 1235{,}3 = 129805{,}32$ грн.

Накладні витрати складають 67% від заробітної плати:

$$C_{\text{Н}} = C_{\text{ЗП}} \cdot 0{,}67,$$

I. $C_{\text{Н}} = 307504{,}67 \cdot 0{,}67 = 206028{,}13$ грн,

II. $C_{\text{Н}} = 311769{,}96 \cdot 0{,}67 = 208885{,}87$ грн.

Отже, вартість розробки ПП за варіантами становить:

$$C_{\text{ПП}} = C_{\text{ЗП}} + C_{\text{ВІД}} + C_{\text{М}} + C_{\text{Н}},$$

I. $C_{\text{ПП}} = 307504{,}67 + 61500{,}93 + 128029{,}47 + 206028{,}13 = 703063{,}2$ грн,

II. $C_{\text{ПП}} = 311769{,}96 + 62353{,}99 + 129805{,}32 + 208885{,}87 = 712815{,}14$ грн.

4.7 Вибір кращого варіанту ПП техніко-економічного рівня

Розрахуємо коефіцієнт техніко-економічного рівня за формулою:

$$K_{\text{ТЕР}j} = K_{\text{Кj}} / C_{\text{Фj}},$$

$$K_{\text{ТЕР1}} = 5{,}1 / 377574{,}8 = 1{,}3731 \cdot 10^{-5},$$
$$K_{\text{ТЕР2}} = 4{,}62 / 380825{,}78 = 1{,}2253 \cdot 10^{-5}.$$

Як бачимо, найбільш ефективним є перший варіант реалізації програми з коефіцієнтом техніко-економічного рівня $K_{\text{ТЕР1}} = 1{,}3731 \cdot 10^{-5}$.

Після виконання функціонально-вартісного аналізу програмного комплексу що розроблюється, можна зробити висновок, що з альтернатив, що залишились після першого відбору двох варіантів виконання програмного комплексу оптимальним є перший варіант реалізації програмного продукту. У нього виявився найкращий показник техніко-економічного рівня якості $K_{\text{ТЕР}} = 1{,}3731 \cdot 10^{-5}$.

Цей варіант реалізації програмного продукту має такі параметри:

– мова програмування – Python;
– використаний фреймворк – TensorFlow;
– платформа для виконання коду – Jupyter Notebook.

Даний варіант виконання програмного комплексу дає користувачу зручний інтерфейс, швидку реалізацію програми та доступний функціонал для роботи. Python має легкий синтаксис та великий вибір бібліотек. Tensorflow зручна для роботи бібліотека, яка швидко обробляє великий об’єм даних. Jupyter Notebook дозволяє використовувати необхідні обчислювальні ресурси.

Висновки до розділу 4

Проведено повний функціонально-вартісний аналіз програмного продукту. Визначено та проведено оцінку основних функцій програмного продукту. Визначено параметри, які характеризують програмний продукт. Проведено експертне оцінювання параметрів та аналіз якості варіантів реалізації функцій. Проведено економічний аналіз варіантів розробки – трудомісткість, витрати на заробітну плату та інші витрати. На основі аналізу вибрано варіант реалізації програмного продукту.

ВИСНОВКИ

У цій роботі досліджено потенціал використання нейронних мереж для прогнозування туристичних потоків. Використання глибоких рекурентних нейронних мереж з довгою короткочасною пам'яттю (LSTM) продемонструвало високу ефективність у передбаченні тенденцій туристичних потоків.

Здійснено огляд стану туристичної галузі Великої Британії, методів прогнозування туристичних потоків, а також проведено аналіз методів оптимізації нейронних мереж. Окреслено постановку задачі та вибір методів дослідження.

Розроблено програмний продукт, у якому нейронна мережа прогнозує стан туристичних потоків на основі як стандартного набору даних, так і даних, отриманих за допомогою фрактальної інтерполяції. Отримані результати підтверджують, що фрактальна інтерполяція покращує продуктивність нейронної мережі завдяки створенню більш деталізованих даних.

Виконано функціонально-вартісний аналіз та обрано оптимальні параметри, включаючи мову програмування, бібліотеки для розробки нейронних мереж та середовище виконання. Також проведено економічний аналіз можливих варіантів розробки.

Таким чином, ця робота підтверджує перспективність використання сучасних алгоритмів машинного навчання для покращення прогнозування туристичних потоків. Подальші дослідження та вдосконалення в цій сфері можуть значно розширити можливості застосування цих технологій і підвищити їхню ефективність у туристичній галузі.

ПЕРЕЛІК ПОСИЛАНЬ

1. Рівненський державний гуманітарний університет. Розвиток туризму. Класифікація туризму. URL: https://kegt.rshu.edu.ua/images/dustan/2020/l_st_l_01.pdf (дата звернення: 10.05.24)
2. П. Р. Пуцентейло. Економіка і організація туристично-готельного підприємства. Розділ 3.4. Класифікація туризму. Центр учбової літератури, 2007. 344 с.
3. ONS. Travel trends estimates: overseas residents in the UK. URL: https://www.ons.gov.uk/peoplepopulationandcommunity/leisureandtourism/datasets/overseasresidentsvisitstotheuk (дата звернення: 15.05.24)
4. Department for Digital, Culture, Media & Sport, The de Bois review: an independent review of Destination Management Organisations in England (pdf), p57, August 2021. URL: https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/1011664/2585-C_The_de_Bois_Review_ACCESSIBLE__for_publication_.pdf#page=57 (дата звернення: 15.05.24)
5. Reuters, “China, UK agree to resume direct passenger flights − British embassy”, 10 August 2022. URL: https://www.reuters.com/world/china-uk-agree-resume-direct-passenger-flights-british-embassy-2022-08-10/ (дата звернення: 16.05.24)
6. ONS, Travel Trends: 2022, 26 May 2023. URL: https://www.ons.gov.uk/peoplepopulationandcommunity/leisureandtourism/articles/traveltrends/latest (дата звернення: 16.05.24)
7. ONS, International Passenger Survey methodological information, 24 March 2017. URL:

https://www.ons.gov.uk/peoplepopulationandcommunity/leisureandtourism/methodologies/internationalpassengersurveybackgroundnotes#:%7E:text=Earnings%20and%20expenditure&text=They%20exclude%20payments%20for%20air,the%20UK%20by%20overseas%20residents (дата звернення: 16.05.24)

8. VisitBritain, Great Britain Tourism Survey 2021 and 2022 Detailed Overview. URL: https://www.visitbritain.org/research-insights/great-britain-domestic-overnight-trips-archive (дата звернення: 16.05.24)
9. Welsh Government, Domestic GB tourism statistics (overnight trips in Wales): 2022, 12 September 2023. URL: https://www.gov.wales/domestic-gb-tourism-statistics-overnight-trips-wales-2022-html#130308 (дата звернення: 17.05.24)
10. Great Britain Tourism Survey results for England, Scotland and Wales. URL: https://www.visitbritain.org/research-insights/great-britain-domestic-overnight-trips-archive (дата звернення: 17.05.24)
11. VisitScotland, Great Britain Tourism Survey 2022. URL: https://www.visitscotland.org/research-insights/about-our-visitors/uk/overnight-tourism-survey#summary22 (дата звернення: 17.05.24)
12. Northern Ireland Statistics and Research Agency, External Overnight. Trips to Northern Ireland 2019, 22 October 2020. URL: https://www.nisra.gov.uk/publications/external-overnight-trips-northern-ireland-publications (дата звернення: 17.05.24)
13. Northern Ireland Statistics and Research Agency, Northern Ireland domestic overnight trips 2019, 22 October 2020. URL: https://www.nisra.gov.uk/publications/domestic-tourism-publications-2019 (дата звернення: 17.05.24)
14. S. Raubitzek, T. Neubauer. A fractal interpolation approach to improve neural network predictions for difficult time series data. Vienna University of Technology, Institute of Information Systems Engineering, Information & Software Engineering Research Division (194-01), Favoritenstrasse 9-11/188, 1040 Vienna, Austria.

15. ONS, OS visits to UK: All visits Thousands – NSA. URL: https://www.ons.gov.uk/peoplepopulationandcommunity/leisureandtourism/timeseries/gmaa/ott (дата звернення: 18.05.24)
16. Statistics on Ukrainians in UK. URL: https://www.gov.uk/government/statistics/immigration-system-statistics-year-ending-march-2023/statistics-on-ukrainians-in-the-uk#data-tables (дата звернення: 18.05.24)
17. P. Manousopoulos. Curve Fitting by Fractal Interpolation. Department of Informatics and Telecommunications, University of Athens, Panepistimioupolis, 157 84, Athens, Greece.
18. Detecting trends and mean reversion with the hurst exponent. URL: https://macrosynergy.com/research/detecting-trends-and-mean-reversion-with-the-hurst-exponent/ (дата звернення: 19.05.24)
19. Efficient calculation of fractal properties via the Higuchi method. URL: https://link.springer.com/article/10.1007/s11071-022-07353-2 (дата звернення: 19.05.24)
20. What is LSTM – Long Short Term Memory? URL: https://www.geeksforgeeks.org/deep-learning-introduction-to-long-short-term-memory/ (дата звернення: 19.05.24)
21. Python Introduction. URL: https://www.w3schools.com/python/python_intro.asp (дата звернення: 21.05.24)
22. Introduction to TensorFlow. URL: https://developers.google.com/machine-learning/crash-course/first-steps-with-tensorflow/toolkit (дата звернення: 21.05.24)
23. The Good and the Bad of Pandas Data Analysis Library. URL: https://www.altexsoft.com/blog/pandas-library/ (дата звернення: 21.05.24)
24. Why Should We Use NumPy? URL: https://medium.com/fintechexplained/why-should-we-use-numpy-c14a4fb03ee9 (дата звернення: 21.05.24)

25. Visualization with Matplotlib. URL: https://jakevdp.github.io/PythonDataScienceHandbook/04.00-introduction-to-matplotlib.html (дата звернення: 21.05.24)
26. The Significance of Jupyter Notebook in Data Science. URL: https://www.linkedin.com/pulse/significance-jupyter-notebooks-data-science-tharaneetharan-m-kclfc/ (дата звернення: 21.05.24)
27. Office for National Statistics. URL: https://www.gov.uk/government/organisations/office-for-national-statistics/about (дата звернення: 21.05.24)

ДОДАТОК А ЛІСТИНГ ПРОГРАМИ

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
import tensorflow as tf
from keras.models import Model
from keras.layers import Input, LSTM, Dense
from tensorflow.keras.callbacks import EarlyStopping
import matplotlib.pyplot as plt
from matplotlib.ticker import MaxNLocator
import random
import hfda

def higuchi_fd(time_series, k_max=12):
    N = len(time_series)
    L = []
    x = np.array(time_series)

    for k in range(1, k_max+1):
        Lk = []
        for m in range(k):
            Lmk = 0
            for i in range(1, int(np.floor((N-m)/k))):
                Lmk += abs(x[m + i * k] - x[m + (i - 1) * k])
            Lk.append((N - 1) / (int(np.floor((N - m) / k)) * k ** (-1)) * Lmk)
        L.append(np.mean(Lk))

    L = np.log(L)
```

```
    k = np.log(np.arange(1, k_max+1))

    H = np.polyfit(k, L, 1)[0]
    return H

def getValuesNorm(file_path='Visits-General.xls', covid=True):
    df = pd.read_excel(file_path)
    if covid:
        df = df[df['Year'] < 2019]
    column_values = df['Visitors'].values
    column_values, data_min, data_max = normalize_values(column_values)
    return column_values, data_min, data_max

def getValuesStand(file_path='Visits-General.xls', covid=True):
    df = pd.read_excel(file_path)
    if covid:
        df = df[df['Year'] < 2019]
    column_values = df['Visitors'].values
    column_values, mean, std_dev = standardize_values(column_values)
    return column_values, mean, std_dev

def normalize_values(data):
    data_normalized = (data - data.min()) / (data.max() - data.min())
    data_normalized = data_normalized.tolist()
    return data_normalized, data.min(), data.max()

def standardize_values(values):
    values_array = np.array(values)
    mean = np.mean(values_array)
    std_dev = np.std(values_array)
```

```
    standardized_values = (values_array - mean) / std_dev
    return standardized_values, mean, std_dev

def revert_values(data, data_min, data_max):
    data_reverted = data * (data_max - data_min) + data_min
    data_reverted_list = data_reverted.tolist()
    return data_reverted_list

def add_hd(data, x_len):
    number_to_add = higuchi_fd(data)

    added_array = np.full((x_len, 1), number_to_add)

    result_array = np.concatenate((data, added_array), axis=1)
    return result_array

def create_samples(column_values, n_samples, x_len):
    seq_len = len(column_values)

    X, y = [], []
    for i in range(seq_len - x_len - 1):
        sample_x, sample_y = np.array(column_values[i:i + x_len]),
np.array(column_values[i + x_len:i + x_len + 1])
#       sample_x = add_hd(sample_x.reshape(x_len, 1), x_len)
        X.append(sample_x)
        y.append(sample_y)
    if n_samples > len(X):
        n_samples = len(X)
    random_indices = np.random.choice(len(X), n_samples, replace=False)
    X = [X[i] for i in random_indices]
```

```
    y = [y[i] for i in random_indices]
    return X, y

def fractal_interpolation(x, y, num_points, D):

    # Ensure x and y are numpy arrays
    x = np.array(x)
    y = np.array(y)

    # Number of segments
    num_segments = len(x) - 1

    # Calculate step size for interpolation
    step = (x[-1] - x[0]) / (num_points - 1)

    # Create interpolated x values
    xi = np.linspace(x[0], x[-1], num_points)

    # Initialize interpolated y values
    yi = np.zeros_like(xi)

    # Initialize with the first y value
    yi[0] = y[0]

    # Fractal interpolation
    for i in range(1, num_points):
        xi_i = xi[i]
        # Find the segment for the current x value
        for j in range(num_segments):
            if x[j] <= xi_i <= x[j+1]:
```

```
            # Linear interpolation within the segment
            t = (xi_i - x[j]) / (x[j+1] - x[j])
            yi[i] = (1 - t) * y[j] + t * y[j+1]

            # Add fractal component
            delta = (y[j+1] - y[j]) * np.abs(t - 0.5) ** (2 - D)
            if np.random.rand() > 0.5:
                yi[i] += delta
            else:
                yi[i] -= delta
            break

    return xi, yi

column_values, data_min, data_max = getValuesNorm()
column_values = column_values[:100]

x = np.array(range(len(column_values)))
y = np.array(column_values)
D = hfda.measure(y, k_max=3)
num_points = len(y)*5
xi, yi = fractal_interpolation(x, y, num_points, D)

# Plot the original and interpolated series
plt.figure(figsize=(10, 6))
plt.plot(x, y, label='Original Series')
plt.plot(xi, yi, 'r-', label='Fractal Interpolated Series')
plt.xlabel('Time')
plt.ylabel('Value')
plt.title('Fractal Interpolation of Time Series')
```

```
plt.legend()
plt.grid(True)
plt.show()

n_samples = 10000
x_len = 60

column_values, data_min, data_max = getValuesNorm()
#column_values = column_values[:100]

x = np.array(range(len(column_values)))
y = np.array(column_values)
D = hfda.measure(y, k_max=3)
num_points = len(y)*5
xi, yi = fractal_interpolation(x, y, num_points, D)

X, y = create_samples(column_values, n_samples, x_len)

X_inter, y_inter = create_samples(yi, n_samples, x_len)

X_train, X_val, y_train, y_val = train_test_split(X, y, test_size=0.2)
X_train = np.array(X_train).reshape(len(X_train),len(X_train[0]),1)
X_val = np.array(X_val).reshape(len(X_val),len(X_val[0]),1)
y_val = np.array(y_val).reshape(len(y_val),len(y_val[0]),1)
y_train = np.array(y_train).reshape(len(y_train),len(y_train[0]),1)

X_inter_train = np.array(X_inter).reshape(len(X_inter),len(X_inter[0]),1)
y_inter_train = np.array(y_inter).reshape(len(y_inter),len(y_inter[0]),1)

def rnn_model(Tx, n_a, n_values=1, rnn_layers=2, drop_prob=0.1,
```

```
activation='linear'):
    inputs = Input(shape=(Tx, n_values))
    x = inputs
    for i in range(rnn_layers - 1):
        lstm = LSTM(n_a, return_sequences=True, dropout=drop_prob)
        x = lstm(x)
    lstm = LSTM(n_a, return_sequences=False, dropout=drop_prob)
    x = lstm(x)
    x = Dense(25, activation='relu')(x)
    x = Dense(1, activation=activation)(x)
    model = Model(inputs=inputs, outputs=x)
    return model

def create_model(n_a, n_epochs, n_layers, drop_prob, activation):
  Tx = len(X_train[0])
  model = rnn_model(Tx=Tx, n_a=n_a, rnn_layers=n_layers, drop_prob=drop_prob,
activation=activation)
  model.compile(optimizer='adam', loss='mse', metrics=['mean_squared_error'])
  history = model.fit(x=X_train, y=y_train, epochs=n_epochs,
validation_data=(X_val, y_val), verbose=0)
  return model, history

Tx = len(X_train[0])
print(Tx)
n_a = 200
n_epochs = 1000
batch_size = len(X_train)

n_values = X_train.shape[2]
```

```
loss = 'mae'

model = rnn_model(Tx=Tx, n_a=n_a, n_values=n_values, rnn_layers=3,
drop_prob=0.1, activation='linear')
model.compile(optimizer='adam', loss=loss, metrics=['mse'])
model.summary()
early_stopping = EarlyStopping(
    monitor='val_loss',
    patience=100,
    verbose=1,
    restore_best_weights=True
)
history = model.fit(x=X_train, y=y_train, epochs=n_epochs, batch_size=batch_size,
validation_data=(X_val, y_val), callbacks=[early_stopping])

Tx = len(X_inter_train[0])
print(Tx)
n_a = 200
n_epochs = 1000
batch_size = len(X_inter_train)

n_values = X_inter_train.shape[2]

loss = 'mae'

model2 = rnn_model(Tx=Tx, n_a=n_a, n_values=n_values, rnn_layers=3,
drop_prob=0.1, activation='linear')
model2.compile(optimizer='adam', loss=loss, metrics=['mse'])
model2.summary()
early_stopping = EarlyStopping(
```

```
    monitor='val_loss',
    patience=100,
    verbose=1,
    restore_best_weights=True
)
history2 = model2.fit(x=X_inter_train, y=y_inter_train, epochs=n_epochs,
batch_size=batch_size, validation_data=(X_val, y_val), callbacks=[early_stopping])

def draw_histories(histories, label, parameters):
  plt.figure(figsize=(10, 6))
  for i in range(len(histories)):
    val_mse = histories[i].history['val_mean_squared_error']
    epochs = range(1, len(val_mse) + 1)
    plt.plot(epochs, val_mse, label='Val MSE'+label+' '+str(parameters[i]))
  plt.title(label)
  plt.xlabel('Epochs')
  ax = plt.gca()
  ax.xaxis.set_major_locator(MaxNLocator(integer=True))
  plt.ylabel('MSE')
  plt.legend()
  plt.grid(True)
  plt.show()

drop_probs = [0, 0.1, 0.2, 0.3, 0.4, 0.5]
activations = ['relu', 'linear', 'selu']
n_layers = [1, 2, 3, 4, 5]
n_a = [10, 25, 50, 100, 200]

histories = []
for i in drop_probs:
```

```
  _, history = create_model(25, 50, 3, i, 'linear')
  histories.append(history)
draw_histories(histories, 'drop_prob', drop_probs)

histories = []
for i in activations:
  _, history = create_model(25, 100, 3, 0.1, i)
  histories.append(history)
draw_histories(histories, 'activations', activations)

histories = []
for i in n_layers:
  _, history = create_model(25, 50, n_layers=i, drop_prob=0.1, activation='linear')
  histories.append(history)
draw_histories(histories, 'n_layers', n_layers)

histories = []
for i in n_a:
  _, history = create_model(i, 50, 3, 0.1, 'linear')
  histories.append(history)
draw_histories(histories, 'n_a', n_a)

hist = history.history['loss']
val_hist = history.history['val_loss']
epochs = range(1, len(hist) + 1)

plt.figure(figsize=(10, 6))
plt.plot(epochs, hist, label='Training '+loss)
plt.plot(epochs, val_hist, label='Validation '+loss)
plt.title(loss+' Over Epochs')
```

```
plt.xlabel('Epochs')
ax = plt.gca()
ax.xaxis.set_major_locator(MaxNLocator(integer=True))
plt.ylabel(loss)
plt.legend()
plt.grid(True)
plt.show()

hist = history2.history['loss']
val_hist = history2.history['val_loss']
epochs = range(1, len(hist) + 1)

plt.figure(figsize=(10, 6))
plt.plot(epochs, hist, label='Training '+loss)
plt.plot(epochs, val_hist, label='Validation '+loss)
plt.title(loss+' Over Epochs')
plt.xlabel('Epochs')
ax = plt.gca()
ax.xaxis.set_major_locator(MaxNLocator(integer=True))
plt.ylabel(loss)
plt.legend()
plt.grid(True)
plt.show()

X_test = column_values[:x_len]
X_test = np.array(X_test).reshape((len(X_test), 1))
for i in range(0, len(column_values)-x_len):
    input_data = X_test[-x_len:].reshape((1, x_len, n_values))
    prediction = model.predict(input_data, batch_size=batch_size, verbose=0)
    X_test = np.append(X_test, prediction, axis=0)
```

```
epochs = range(1, len(column_values) + 1)

plt.figure(figsize=(10, 6))
plt.plot(epochs, column_values, label='True Data')
plt.plot(epochs, X_test, label='Predicted Data')
plt.title('Tourists')
plt.xlabel('Time series')
ax = plt.gca()
ax.xaxis.set_major_locator(MaxNLocator(integer=True))
plt.ylabel('Norm Tourists')
plt.legend()
plt.grid(True)
plt.show()

X_test = column_values[:x_len]
X_test = np.array(X_test).reshape((len(X_test), 1))
for i in range(0, len(column_values)-x_len):
    input_data = X_test[-x_len:].reshape((1, x_len, n_values))
    prediction = model2.predict(input_data, batch_size=batch_size, verbose=0)
    X_test = np.append(X_test, prediction, axis=0)

epochs = range(1, len(column_values) + 1)

plt.figure(figsize=(10, 6))
plt.plot(epochs, column_values, label='True Data')
plt.plot(epochs, X_test, label='Predicted Data')
plt.title('Tourists')
plt.xlabel('Time series')
ax = plt.gca()
```

```
ax.xaxis.set_major_locator(MaxNLocator(integer=True))
plt.ylabel('Norm Tourists')
plt.legend()
plt.grid(True)
plt.show()
```