Міністерство освіти і науки України

Харківський національний університет радіоелектроніки

Факультет комп’ютерної інженерії та управління
(повна назва)

Кафедра електронних обчислювальних машин
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА

Пояснювальна записка

Рівень вищої освіти другий (магістерський)

Нейромережеві методи
прогнозування поширення захворювань
(тема)

Виконав:

студент II курсу, групи КСМм-23-1

Гуренко Д.М.
(прізвище, ініціали)

Спеціальність
123 «Комп’ютерна інженерія»
(код і повна назва спеціальності)

Тип програми освітньо-професійна
(освітньо-професійна або освітньо-наукова)

Освітня програма
Комп’ютерні системи та мережі
(повна назва освітньої програми)

Керівник: доц. Іващенко Г.С.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри ЕОМ _____ Коваленко А.А.
(підпис) (прізвище, ініціали)

2025 р.

Харківський національний університет радіоелектроніки

Факультет комп'ютерної інженерії та управління

Кафедра електронних обчислювальних машин

Рівень вищої освіти другий (магістерський)

Спеціальність 123 «Комп'ютерна інженерія»
(код і повна назва)

Тип програми освітньо-професійна
(освітньо-професійна або освітньо-наукова)

Освітня програма Комп'ютерні системи та мережі
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
"____" _____ 20__ р.

ЗАВДАННЯ

НА КВАЛІФІКАЦІЙНУ РОБОТУ

студенту Гуренко Даніїлу Максимовичу
(прізвище, ім'я, по батькові)

1. Тема роботи Нейромережеві методи прогнозування поширення захворювань

затверджена наказом по університету від " 22 " листопада 2024 р. № 1237 Ст

2. Термін подання студентом роботи до екзаменаційної комісії 20 січня 2025 р.

3. Вхідні дані до роботи 1) кросплатформове інтегроване середовище розробки PyCharm; 2) документація мови програмування Python; 3) набори даних пандемії COVID-19; 4) документація TensorFlow.

4. Перелік питань, що потрібно опрацювати у роботі

1) аналіз предметної області;
2) аналіз існуючих досліджень;
3) підготовка даних для дослідження;
4) огляд методів прогнозування часових рядів;
5) програмна реалізація;
6) аналіз результатів експериментальних досліджень;
7) висновки.

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (слайдів)

Слайд-презентація – 14 слайдів

6. Консультанти розділів роботи (заповнюється за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1	Огляд сучасних моделей штучних нейронних	26.11.24-30.11.24	
	мереж		
2	Вибір та обґрунтування методики дослідження	02.12.24-05.12.24	
3	Вибір інструментальних засобів	06.12.24-10.12.24	
4	Реалізація нейромережевих методів	11.12.24-21.12.24	
5	Проведення експериментів	23.12.24-03.01.25	
6	Оформлення матеріалів кваліфікаційної роботи	04.01.25-07.01.25	
7	Подання кваліфікаційної роботи керівникові та	08.01.25-11.01.25	
	її попередній захист		
8	Подання кваліфікаційної роботи на рецензу-	13.01.25-17.01.25	
	вання		

Дата видачі завдання 25 листопада 2024 р.

Студент _____
(підпис)

Керівник роботи _____ доц. Іващенко Г.С.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка кваліфікаційної роботи: 70 с., 24 рис., 3 табл., 2 дод., 23 джерел.

НЕЙРОМЕРЕЖА, ШТУЧНА НЕЙРОННА МЕРЕЖА, ЧАСОВИЙ РЯД, ЗАХВОРЮВАННЯ, ПАНДЕМІЯ, COVID-19, RNN, LSTM, CNN, PYTHON, TENSERFLOW.

Метою кваліфікаційної роботи є реалізація та дослідження нейромережевих методів прогнозування поширення захворювань. Проведення порівняльного аналізу ефективності використання моделей на базі LSTM, RNN та CNN. Основна увага приділена застосуванню нейронних мереж для аналізу часових рядів та побудови моделей, здатних забезпечувати високоточне прогнозування динаміки захворюваності.

У ході виконання кваліфікаційної роботи було проаналізовано основні проблеми, пов’язані з прогнозуванням поширення захворювань. У рамках роботи були розглянуті сучасні нейромережеві підходи, які дозволяють ефективно обробляти великі набори даних, виявляти складні взаємозв’язки та тренди. Було виконано порівняння ефективності різних моделей на прикладах реальних даних, включаючи глобальні пандемії.

ABSTRACT

Master’s thesis: 70 pages, 24 figures, 3 tables, 2 appendices, 23 sources.

NEURAL NETWORK, ARTIFICIAL NEURAL NETWORK, TIME SERIES, DISEASE, PANDEMIC, COVID-19, RNN, LSTM, CNN, PYTHON, TENSERFLOW.

The major goal of this thesis is to develop and study neural network methods for predicting the spread of diseases, as well as to evaluate their effectiveness in comparison with traditional forecasting approaches. The main focus is on the use of neural networks to analyze time series and build models that can provide highly accurate forecasting of disease dynamics.

In the course of the qualification work, the main problems associated with forecasting the spread of diseases were analyzed. As part of the work, modern neural network approaches were considered, which allow efficient processing of large data sets, identifying complex relationships and trends. The efficiency of different models was compared using real data, including global pandemics.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

ШІ – штучний інтелект

ШНМ – штучна нейронна мережа

BPTT – зворотне поширення помилки у часі (англ., Backpropagation Through Time)

CNN – згорткові нейронні мережі (англ., Convolutional Neural Networks)

FFNN – нейронні мережі прямого поширення (англ., Feed Forward Neural Networks)

GRU – закриті рекурентні одиниці (англ., Gated Recurrent Units)

LSTM – довга короткострокова пам'ять (англ., Long Short-Term Memory)

ReLU – зрізаний лінійний вузол (англ., Rectified Linear Unit)

RNN – рекурентна нейронна мережа (англ., Recurrent Neural Network)

ВСТУП

У сучасному світі стрімке поширення інфекційних захворювань та пандемій стає все більш значущою проблемою. Зростання кількості подорожей, урбанізація та глобалізація сприяють швидкому поширенню нових інфекційних загроз. Ефективне прогнозування поширення захворювань є важливим інструментом для планування медичних заходів та запобігання потенційних пандемій.

Традиційні методи прогнозування захворювань, такі як статистичні моделі або епідеміологічні підходи, мають певні обмеження у швидкості та точності. Сучасні методи машинного навчання, зокрема нейронні мережі, покращують результати прогнозування, дозволяючи ефективніше обробляти великі обсяги даних та виявляти складні нелінійні закономірності для визначення орієнтовних дат початку та завершення захворюваності, а їхня гнучкість та можливість налаштування дозволяють адаптувати алгоритми та моделі до різних даних. Наприклад, для глобальних пандемій, таких як ВІЛ-інфекція або коронавірусна інфекція COVID-19 та для локальних випадків захворюваності з меншим обсягом даних.

Отримана інформація надає можливість своєчасно реагувати на сплески захворюваності, приймати необхідні рішення, планувати та прогнозувати майбутні події на основі аналізу минулих даних. Прогнозування розвитку захворювань є актуальним завданням, від якого залежать людські життя, стан економік країн. Розвиток цієї галузі може допомогти людству у подоланні проблеми пандемій та зменшення шкоди, завданої ними.

У роботі розглядаються основні принципи використання та побудови нейромережевих методів для роботи з часовими рядами, їх адаптація до завдань прогнозування поширення захворювань та проводиться аналіз ефективності запропонованих підходів на прикладі реальних даних.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Актуальність прогнозування поширення захворювань

Прогнозування поширення інфекційних захворювань є надзвичайно важливим аспектом у сучасній охороні здоров’я, зокрема в умовах глобалізації, зміни клімату та зростаючої мобільності населення. Ефективне прогнозування допомагає не тільки контролювати поширення захворювань, а й вчасно вживати профілактичні заходи для зниження ризику виникнення епідемій та пандемій. У цьому контексті роль прогнозування важлива при аналізі та прогнозуванні складних і динамічних систем.

Коронавірусна інфекція COVID-19 продемонструвала вразливість світової спільноти до нових вірусних інфекцій. Країни зіткнулися з труднощами в управлінні ресурсами, розподілі медичних засобів і прогнозуванні майбутніх хвиль захворюваності. Традиційні методи прогнозування виявилися недостатньо ефективними через обмеженість моделей, які використовувалися до цього часу, що потребувало активного розвитку нових підходів [1]. Використання сучасних технологій, таких як глибокі нейронні мережі, дозволяє забезпечити більш точні прогнози, враховуючи численні змінні, які впливають на поширення захворювань.

Крім того, змінювані екологічні та соціальні фактори сприяють появі нових інфекційних загроз. Зміна клімату та глобальне потепління стимулюють поширення нових патогенів у регіонах, де раніше вони не зустрічалися. Наприклад, захворювання, що передаються комахами, такі як малярія або вірус Зіка, можуть з'являтися у нових географічних зонах, що раніше були позбавлені таких загроз.

З іншого боку, урбанізація та підвищення густоти населення створюють нові виклики для епідеміологічного моніторингу та прогнозування.

У поточних обставинах прогнозування захворювань стає важливим інструментом для управління кризовими ситуаціями. Це дозволяє урядам та організаціям охорони здоров’я оперативно реагувати на потенційні спалахи та запобігати масовим зараженням, створюючи системи раннього попередження. Мета системи прогнозування поширення захворювань полягає в тому, щоб дати медичним працівникам достатнє попередження до несподівано високої кількості випадків, або ж завчасно запровадити заходи, щоб запобігти спалахам захворювань.

Ранні попередження про ймовірні спалахи можуть суттєво знизити економічні втрати та зменшити кількість жертв завдяки своєчасному введенню карантинних заходів та вакцинації населення. Довгострокове прогнозування потрібне, коли метою є стратегічний контроль захворювань, наприклад, у програмі боротьби з онхоцеркозом ВООЗ для зменшення річкової сліпоти в деяких частинах Західної Африки, що можливо лише за умови розуміння динаміки передачі хвороби.

Отже, сучасні виклики, пов'язані з поширенням інфекційних захворювань, вимагають нових підходів до їх прогнозування. Застосування методів машинного навчання та нейромережевих технологій забезпечують можливості для аналізу і прогнозування розвитку епідемій, що є важливим інструментом для захисту суспільного здоров'я та зменшення наслідків глобальних пандемій. Одним з напрямків використання методів машинного навчання є прогнозування часових рядів, що дозволяє враховувати динаміку змін та виявляти тренди в розвитку захворювань.

1.2 Характеристики часових рядів

Часовий ряд – це послідовність спостережень, структура даних, де кожен елемент складається з двох частин: число та часова відмітка, асоційована з цим числом. Часові ряди зазвичай упорядковані в часі, і дані в такому ряді можуть залежати як від часу, так і від попередніх спостережень.

Аналіз часових рядів дозволяє виявляти тренди, цикли, а також інші закономірності, що змінюються з плином часу. Прикладом може бути температура протягом дня, кількість хворих щодня. Основною метою аналізу часових рядів є виявлення моделей або закономірностей у даних і подальше використання цих закономірностей для прогнозування майбутніх значень.

Існують різні типи часових рядів, які використовуються в залежності від їх характеристик. За характером зміни одним із найпростіших є стаціонарний часовий ряд (рисунок 1.1), де статистичні властивості, такі як середнє та дисперсія, залишаються постійними протягом часу [2].

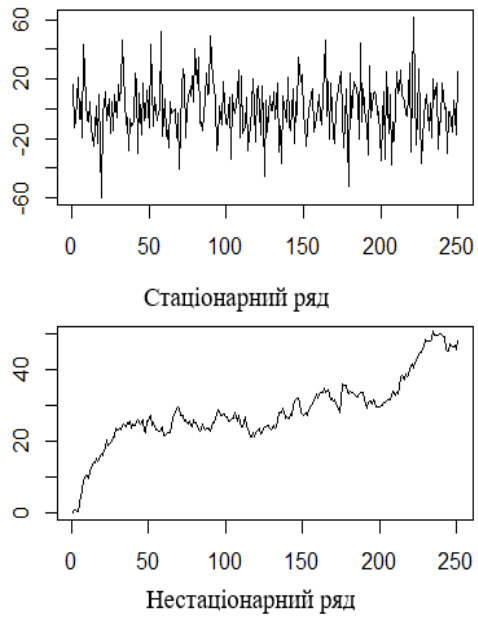

Рисунок 1.1 – Стаціонарний та нестаціонарний часові ряди

Наприклад, якщо аналізувати споживання електроенергії вночі в певному регіоні, можна вважати цей процес стаціонарним, оскільки в такі періоди споживання є стабільним і незначно змінюється. Однак, у реальному світі більшість часових рядів є нестаціонарними (рисунок 1.1), оскільки їхні характеристики змінюються з часом. У випадку захворювань це може бути пов’язано з сезонністю, змінами в поведінці населення або іншими зовнішніми факторами.

Детермінований часовий ряд – це часовий ряд [3], у якому майбутні значення можна точно спрогнозувати на основі математичних правил або рівнянь, що описують поведінку цього ряду (рисунок 1.2). У детермінованих часових рядах немає випадкових або непрогнозованих компонентів, і кожне значення повністю визначається попередніми значеннями або зовнішніми факторами. Це означає, що при повторному проведенні експерименту або вимірюванні, часовий ряд буде виглядати однаково кожного разу.

Недетермінований або стохастичний часовий ряд – це часовий ряд [4], де майбутні значення містять елемент випадковості, і їх неможливо точно спрогнозувати лише на основі минулих значень (рисунок 1.2). Прикладом недетермінованого ряду може бути коливання ринкових цін, погоди або кількості захворювань.

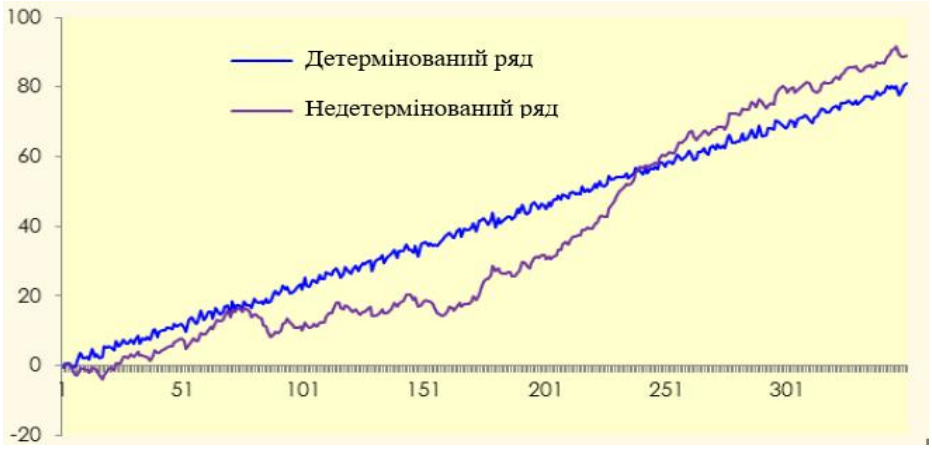

Рисунок 1.2 – Детермінований та недетермінований часові ряди

Часові ряди бувають одновимірні та багатовимірні. Одновимірні містять спостереження за зміною лише одного параметра досліджуваного процесу, а багатовимірні за двома або більше параметрами.

У дискретному часовому ряді значення змінної вимірюються окремими, дискретними моментами часу. Між цими моментами значення невідомі чи не мають значення. У свою чергу в безперервному часовому ряді змінна може набувати будь-якого значення в заданому інтервалі часу.

Тренд – це довгострокова тенденція до зростання або зменшення, яка спостерігається протягом тривалого періоду. Наприклад, кількість випадків діабету у світі може збільшуватися з кожним роком через зміни в харчуванні та способі життя.

Сезонність – це повторювані коливання, які відбуваються в певні періоди часу. Сезонні коливання можуть спостерігатися в захворюваннях, пов'язаних з погодними умовами, наприклад, при грипі, який має схильність поширюватися взимку.

1.3 Аналіз актуальних досліджень

1.3.1 COVID-19 Forecast Hub

Підходи до машинного навчання, які використовує центр прогнозування COVID-19 [5], включають мережі довгострокової короткочасної пам’яті (LSTM) і Gated Recurrent Unit (GRU), які відповідають даним часових рядів. Ці нейронні мережі можуть запам’ятовувати залежності протягом довгих послідовностей, що забезпечує їх ефективність в моделюванні тенденцій зараження, рівня госпіталізації та летальних випадків.

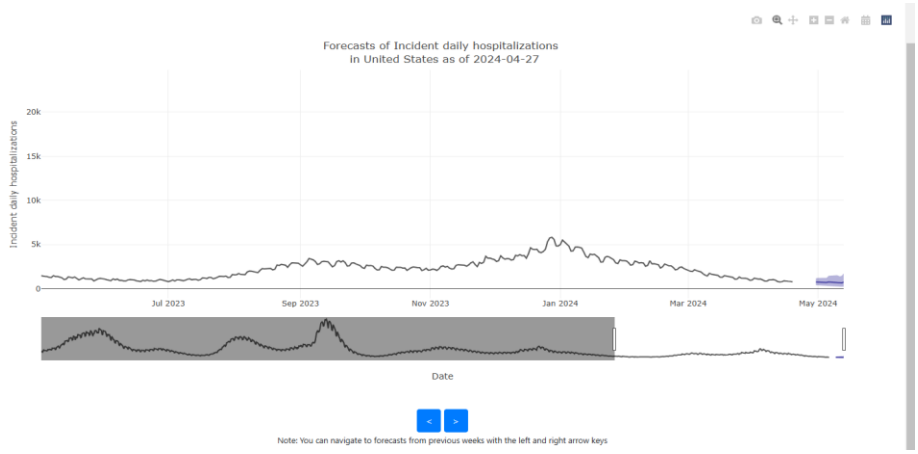

Рисунок 1.3 – Прогнозування випадків захворювання в США

Вхідними даними в статті є безперервний нестаціонарний одновимірний часовий ряд, який може відображати нові випадки захворюваності, смертей та одужання. Результатом є прогнозовані дані (рисунок 1.3). Сервіс рекомендує проводити прогнозування з горизонтом від 1 до 28 днів. Найточнішим є прогноз на один і два тижні. Прогнозування більш ніж 28 днів втрачає свою точність та показує ненадійні результати на більш довгих горизонтах.

Для точного прогнозування сервіс використовує зважену медіану всіх прогнозів, включаючи нейромережеві та статистичні методи, що забезпечує більш точний прогноз.

1.3.2 Прогнозування COVID-19 з використанням рекурентних нейронних мереж

Основною метою дослідження [6] було створення сучасної моделі рекурентних нейронних мереж (RNN) глибокого навчання для прогнозування сукупних підтверджених випадків, сукупних випадків одужання та сукупних смертельних випадків.

Для прогнозування майбутніх тенденцій COVID-19 розроблені методи на основі (GRU) і довготривалої короткочасної пам’яті (LSTM). Для прогнозування використовуються загальнодоступні дані з бази даних COVID-19 університету Джона Гопкінса. Вихідними даними є безперервний одновимірний часовий ряд, який є нестаціонарним. Дослідження підкреслює важливість різних факторів, таких як вік, профілактичні заходи та аспекти роботи заклади охорони здоров’я, щільність населення, які відіграють важливу роль у швидкому поширенні пандемії COVID-19.

Підкреслюється більша ефективність рекурентних нейронних мереж порівняно з FFNN, оскільки вони не здатні враховувати тенденції у даних часових рядів.

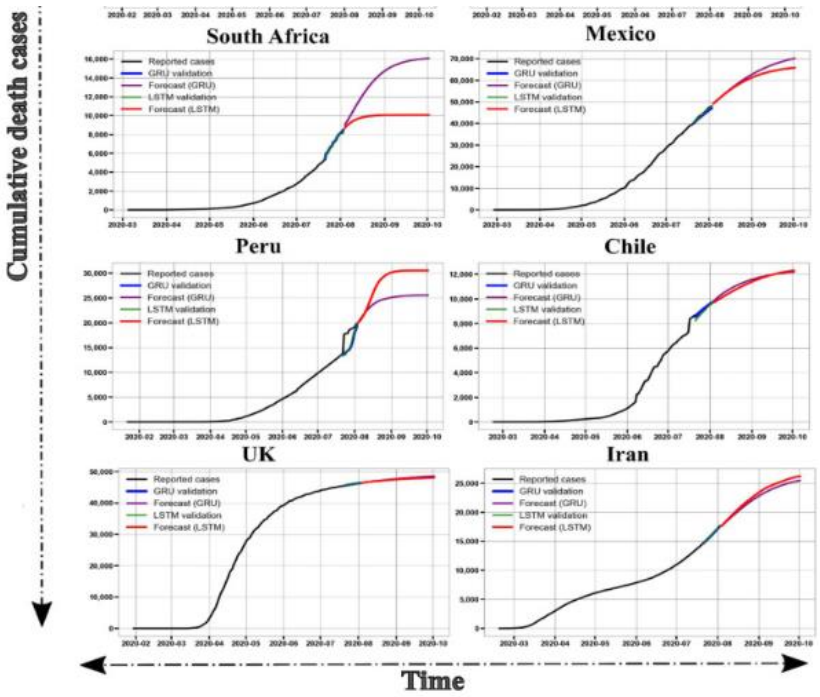

Рисунок 1.4 – Прогнозування випадків захворювань досліджуваними методами в країнах світу

Процес навчання RNN на довгих вхідних послідовностях ускладений через проблему зникнення градієнтів. Отже, основним недоліком архітектури RNN є її коротша пам'ять для запам'ятовування функцій, зникаючих та вибухаючих градієнтів. Для подолання цієї проблеми в дослідженні використовується LSTM, вказується про перевагу у довгостроковому прогнозуванні. Результатом дослідження є 60-денний прогноз (рисунок 1.4) у ряді країн пандемії COVID-19 за допомогою моделей RNN, LSTM, GRU, в якому модель LSTM показала найкращі результати для даних більшості країн.

1.3.3 Прогнозування захворюваності на ВІЛ на основі рекурентних нейронних мереж

Дослідження [7] ставить за мету визначити найефективнішу модель відстеження епідемії СНІДу, яка забезпечить методологічну основу для

тестування часових характеристик захворювання. Порівнюються моделі BPNN, RNN, LSTM та MHPSO-GRU (рисунок 1.5). Результат характеризується середньою квадратичною помилкою (RMSE), середньою абсолютною помилкою (MAE), середньою частотою помилок (MER) та середньою абсолютною відсотковою помилкою (MAPE).

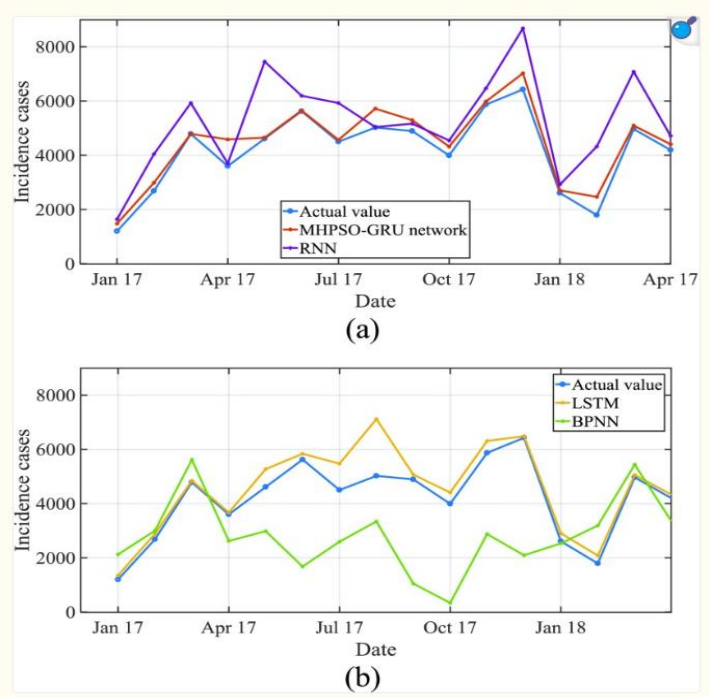

Рисунок 1.5 – Результати прогнозування захворювання на ВІЛ досліджуваних методів

Часовий ряд, необхідний для навчання моделей, був зібраний у діапазоні з січня 2004 року до квітня 2018 року, 458078 випадків СНІДу, в середньому 2664 випадки на місяць. У 2017 році в Китаї 57194 випадків ВІЛ на місяць, що в 18 разів більше, ніж у 2004 році, що є висхідною тенденцією з сезонністю і повторенням тенденцій зростання.

Результатом дослідження став порівняльний аналіз ефективності чотирьох обраних найкращих моделей. Виявлено, що порівняно з MAE, MAPE, MER та RMSE результати BPNN були найнижчими. Моделі GRU та LSTM виявилися найбільш прийнятними моделями.

1.4 Використовувані вихідні дані

Для побудови моделей прогнозування захворювань важливо використовувати набори даних, які відображають динаміку розвитку епідемії або пандемії. Основними типами даних є часові ряди, що включають інформацію про кількість випадків захворювання, летальних випадків, одужавших, але для прогнозування є сенс використовувати також дані про рівень госпіталізацій, рівень вакцинації, демографічні характеристики. Наприклад, дані про кількість нових випадків щодня дають змогу відстежувати поширення інфекції, а дані про госпіталізацію – оцінити навантаження на медичну систему. Дані про рівень вакцинації допомагають спрогнозувати рівень імунітету серед населення та можливість сповільнення поширення інфекції.

Основні джерела даних – це національні та міжнародні організації охорони здоров'я, такі як Всесвітня організація охорони здоров'я (ВООЗ), Центри контролю та профілактики захворювань (CDC), а також дослідницькі та аналітичні платформи Kaggle, COVID-19 Forecast Hub і Our World in Data [8]. Такі ресурси надають систематизовані та відкриті набори даних, що регулярно оновлюються, що важливо для забезпечення точності прогнозів.

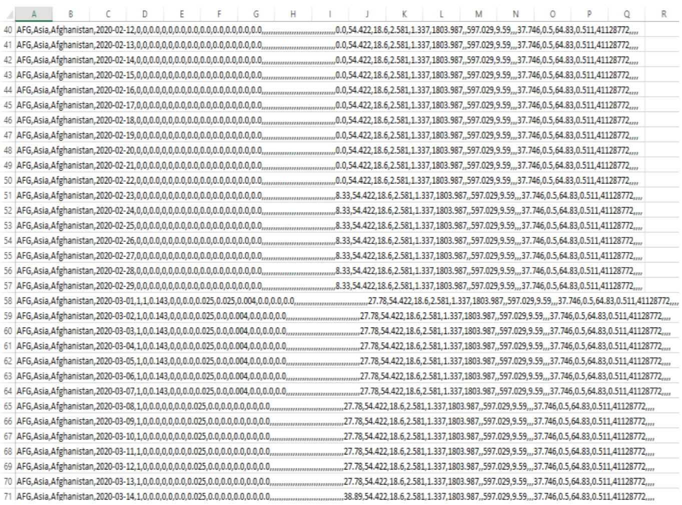

Рисунок 1.6 – Приклад датасету пандемії COVID-19

Our World in Data [8] дає можливість завантажувати датасети, наприклад, пандемії COVID-19, у форматі .csv (рисунок 1.6), в яких міститься інформація про кількість нових захворювань, смертей, одужавших пацієнтів, загальної кількості хворих. Таким чином, сучасні сервіси надають безкоштовний доступ до датасетів, які дають можливість навчати моделі на великих обсягах даних.

1.5 Постановка задачі

Метою роботи є дослідження нейромережевих методів прогнозування поширення захворювань.

Для досягнення мети необхідно виконати наступні завдання:

- оцінити та підготувати дані, необхідні для прогнозування, а саме, часові ряди захворювань, кількісті летальних випадків, кількісті одужавших;
- розглянути існуючі нейромережеві методи для створення надійних моделей прогнозування;
- налаштувати параметри обраних моделей, щоб забезпечити високу точність прогнозів для різних часових інтервалів;
- провести тестування моделей на реальних даних, порівняти прогнози з фактичними показниками та оцінити їхню ефективність.

Необхідно оцінити переваги та недоліки кожного реалізованого методу для рекомендацій щодо використання конкретних підходів у прогнозуванні захворювань.

2 ВИКОРИСТАННЯ ШТУЧНИХ НЕЙРОННИХ МЕРЕЖ У ЗАВДАННІ ПРОГНОЗУВАННЯ

2.1 Штучний нейрон

Штучний нейрон – це точка з’єднання в штучній нейронній мережі, базовий компонент, що імітує функціонування біологічного нейрону. Основна мета штучного нейрону – обробка, зберігання та передача інформації [9].

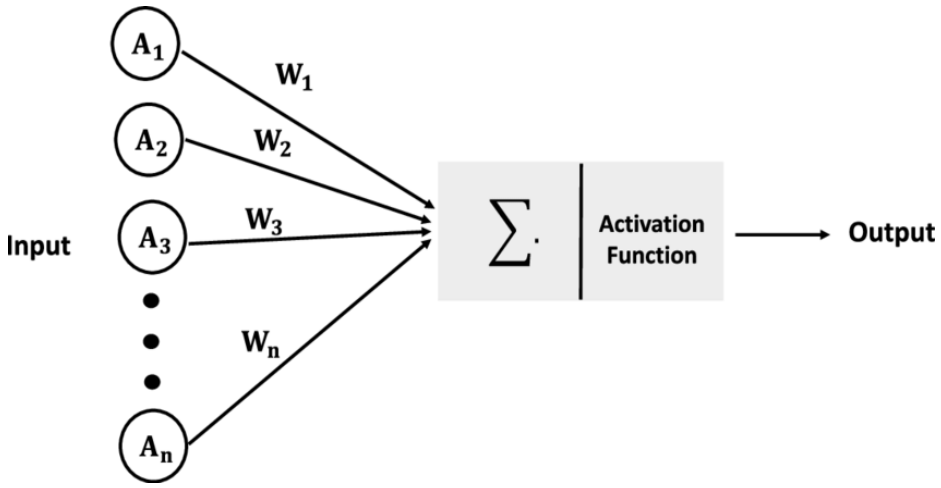

Рисунок 2.1 – Штучний нейрон

І в штучній, і в біологічній архітектурі вузли називаються нейронами [10], а з’єднання характеризуються синаптичними вагами, які представляють значимість з’єднання (рисунок 2.1).

Під час обробки вхідних даних на нейронах розраховується значення функції активації, яка визначає можливість подальшої передачі вихідної інформації на наступний рівень (шар) нейронної мережі. Оскільки функції активації можуть бути лінійними або нелінійними, нейрони часто мають широкий діапазон конвергенції та дивергенції. Конвергенція – це здатність

одного нейрона отримувати дані від багатьох інших нейронів у мережі, а дивергенція – здатність одного нейрона обмінюватися інформацією з багатьма іншими нейронами в мережі.

2.2 Штучна нейронна мережа

Нейронна мережа визначається як система обробки сигналів [11], що складається з взаємопов’язаних вузлів, через які проходять вхідні сигнали для генерування виходу. Нейронні мережі є абстрактними, але вони імітують поведінку нейронів, синапсів і аксонів у мозку людини. Керуючий алгоритм налаштовує ці пороги активації в мережі, навчаючи систему покращувати її продуктивність.

Нейронна мережа складається з шарів взаємопов’язаних вузлів (рисунок 2.2), організованих у три основні типи шарів [12], що визначають здатність мережі до навчання, узагальнення та аналізу даних.

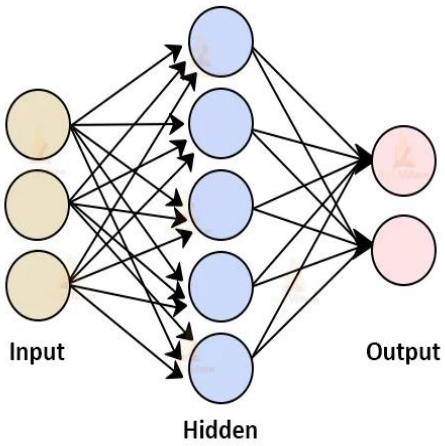

Рисунок 2.2 – Штучна нейронна мережа

Вхідний шар приймає початкові дані. Кожен нейрон у цьому шарі представляє окрему характеристику або параметр вхідних даних. Вхідний шар не виконує обчислень – він передає інформацію прихованим шарам. Між вхідним і вихідним шарами може бути один або кілька прихованих шарів. Ці

рівні виконують складні обчислення вхідних даних. Кожен зв'язок між нейронами має вагу, яка показує силу зв'язку. Під час навчання ваги коригуються, щоб мінімізувати помилку між прогнозом і реальним значенням.

Нейрони вихідного шару так само, як і прихованих шарів, обробляють дані. Число нейронів визначається кількістю залежних змінних моделі. Вихідний шар генерує кінцевий результат обробки, а саме прогнозоване значення.

2.3 Навчання нейронної мережі

Навчання нейронної мережі – це процес налаштування її параметрів, що дозволяє моделі узагальнювати вхідні дані та прогнозувати результати з високою точністю [13].

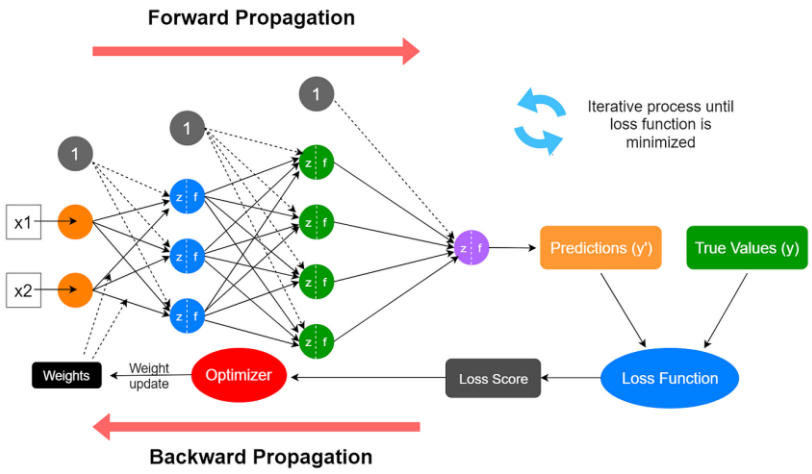

Рисунок 2.3 – Схематичне зображення тренування нейромережі

Процес тренування (рисунок 2.3) складається з кількох етапів, серед яких ініціалізація ваг, передача даних через шари (forward propagation), обчислення похибки, зворотнє поширення помилки (backpropagation) та оновлення ваг. У рамках тренування використовуються функції активації

(ReLU, Sigmoid) і методи оптимізації, такі як градієнтний спуск, що допомагає досягти бажаної точності мережі.

Перенавчання є поширеною проблемою, що виникає, коли нейронна мережа демонструє високу ефективність на тренувальних даних, але має низьку точність на реальних. Проблема виникає через надмірну адаптацію моделі до патернів у навчальних даних, включаючи випадкові шуми. Для боротьби з проблемою перенавчання [14] існують такі способи, як:

- спрощення моделі, що досягається зменшенням кількості нейронів;
- використання ранньої зупинки, що визначається вказівкою щодо того, скільки ітерацій можна виконати, перш ніж модель почне перенавчатись;
- використання методу Dropouts, який випадковим чином видаляє нейрони з нейронної мережі під час навчання на кожній ітерації. Для цього необхідно навчити кілька нейронних мереж і при їх навчанні відключати нейрони, випадкове відключання нейрону змінює саму мережу, що еквівалентно тренуванню різних нейронних мереж, які можуть навчитись з різною ефективністю.

2.4 Рекурентні нейронні мережі (RNN)

Рекурентні нейронні мережі (RNN) – це тип штучних нейронних мереж [15], які дозволяють використовувати попередні виходи як вхідні, маючи приховані стани. Підходять для роботи з послідовними даними, можуть зберігати інформацію про попередні входи та використовувати їх для прогнозувань, що забезпечує ефективність для задач аналізу часових рядів та прогнозування захворювань.

RNN відрізняються від звичайних нейронних мереж тим, що вони мають зворотні зв’язки між нейронами (рисунок 2.4). Це дозволяє їм обробляти не тільки поточний вхід, але й враховувати інформацію про попередні стани мережі.

Вихід останнього прихованого шару використовується для прогнозування поточного кроку часу (рисунок 2.4). Мережа навчається шляхом регулювання вагових коефіцієнтів зв’язків між шарами.

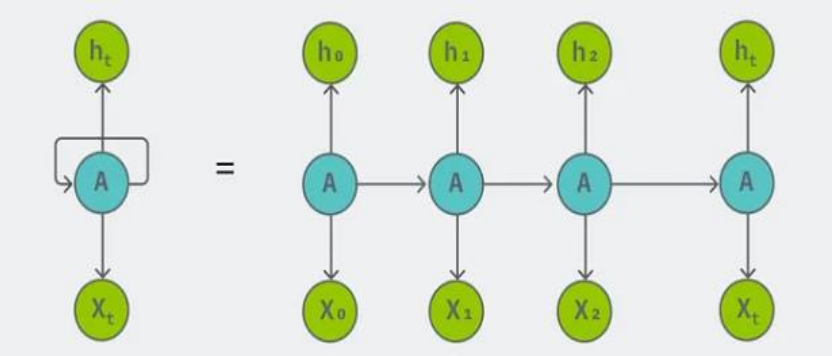

Рисунок 2.4 – Розгортка архітектури нейронної мережі

Навчання RNN здійснюється за допомогою зворотнього поширення у часі (BPTT) [16]. BPTT – це варіант зворотнього поширення помилки, який використовується для навчання рекурентних нейронних мереж. BPTT працює, поширюючи помилку з виходу мережі назад через часові кроки, оновлюючи ваги з’єднань у міру проходження.

Одним з недоліків RNN [17] є проблема зникаючого градієнта. Це відбувається, коли градієнт під час навчання стає надзвичайно малим у міру зворотнього поширення в часі. Це обмежує спроможність мережі до моделювання довгострокових залежностей. Ця проблема вирішується модифікацією LSTM. Також RNN, на відміну від CNN, обробляють дані послідовно, що ускладнює розпаралелювання і призводить до уповільнення навчання.

2.5 Long Short-Term Memory (LSTM)

LSTM (Long Short-Term Memory) – це тип рекурентної нейронної мережі (RNN), створений для обробки довгих послідовностей даних та

збереження інформації на тривалий час [18]. Основна перевага LSTM, порівняно з RNN, полягає в можливості уникати проблеми зникання градієнта, яка часто виникає під час тренування моделей з великими часовими рядами.

Архітектура LSTM включає комірку пам’яті, яка керується трьома воротами: вхідним, забуття і вихідним. Вони вирішують, яку інформацію до комірки пам’яті додати, видалити та вивести (рисунок 2.5). Це дозволяє мережі LSTM вибірково зберігати або забувати інформацію.

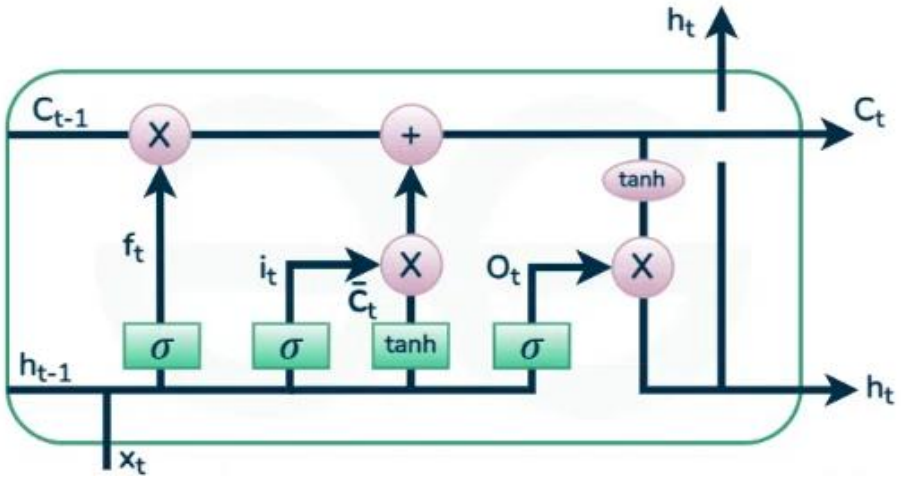

Рисунок 2.5 – Структура LSTM мережі

Вхідний вентиль [19] контролює, яка нова інформація буде додана до блоку пам’яті. Вентиль приймає поточний вхід і попередній прихований стан, після чого обробляє через сигмоїдну функцію та фільтрує значення, які потрібно запам’ятати. Функція tanh генерує вектор, який дає вихідні дані від -1 до +1, що містить усі можливі значення (рисунок 2.6).

Активація вхідних воріт визначає, які значення треба оновити. Сигмоїдна функція повертає число від 0 (повністю забути) до 1 (повністю зберегти). Наступним кроком вагова матриця та член зміщення для вхідного вентиля коригують вхідні дані та генеруються нові значення за допомогою функції tanh, яка повертає значення від -1 до 1.

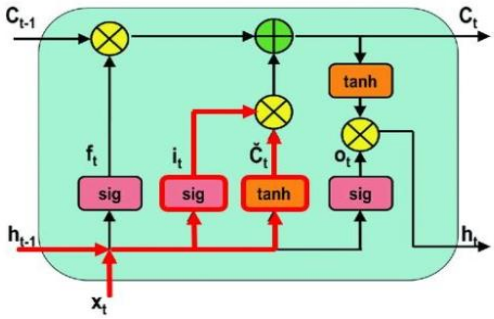

Рисунок 2.6 – Схематичне зображення вхідного вентилю

Оновлений стан комірки поєднується зі старим, масштабованим виходом вентилю забуття, з новими значеннями, масштабованими активацією вхідного вентилю (рисунок 2.7).

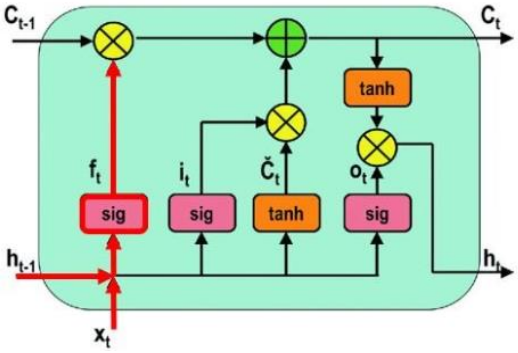

Рисунок 2.7 – Схематичне зображення вентилю забуття

Інформація, яка більше не є корисною в стані комірки, видаляється за допомогою шлюзу забуття. Він використовує сигмоїдну функцію для отримання числа від 0 до 1 для кожного фрагмента інформації. Цей шлюз допомагає запобігти накопиченню нерелевантних даних. Шлюз забуття визначає, інформацію залишити або забути. Сигмоїдальна функція використовує вихідні дані з попередньої ітерації та повертає значення від 0 до 1. Вагова матриця та зміщення для шлюзу забуття коригують вхідні дані.

Вихідний вентиль вирішує, яка частина стану комірки повинна бути виведена (рисунок 2.8).

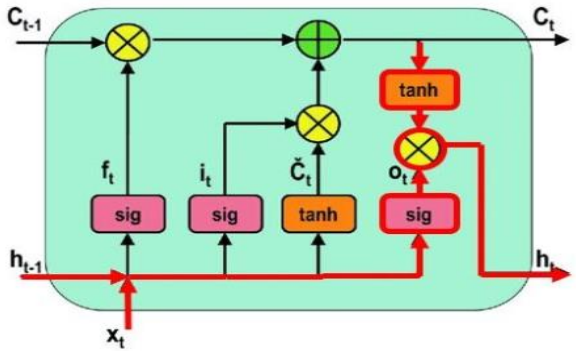

Рисунок 2.8 – Схематичне зображення вихідного вентилю

Сигмоїдна функція повертає значення від 0 до 1. Вагова матриця та зміщення для вихідного вентиля коригують вхідні дані. Вихід прихованого стану генерується шляхом застосування функції tanh до оновленого стану комірки та множення на активацію вихідного вентиля.

2.6 Згорткова нейронна мережа (CNN)

Convolutional Neural Networks (CNNs) – це особливий вид нейронних мереж [20], що використовують механізм згортки (рисунок 2.9), дозволяючи виділяти суттєві ознаки на різних рівнях, полегшуючи розпізнавання візуальних або просторових закономірностей. Незважаючи на те, що CNN спочатку розроблялися для обробки зображень, вони можуть ефективно використовуватись для обробки послідовних даних.

Можливо розглядати часовий ряд як послідовність подій, схожу на піксельні значення в зображенні, де сусідні точки даних містять інформацію, що допомагає розпізнавати закономірності.

Однією з переваг згорткових нейронних мереж (CNN) у прогнозуванні часових рядів є наявність локальних зв’язків, які забезпечують ефективне виявлення короткострокових закономірностей. CNN складаються з кількох згорткових шарів, що дозволяє вивчати ієрархічні представлення даних. Це забезпечує їх здатність виявляти короткострокові залежності.

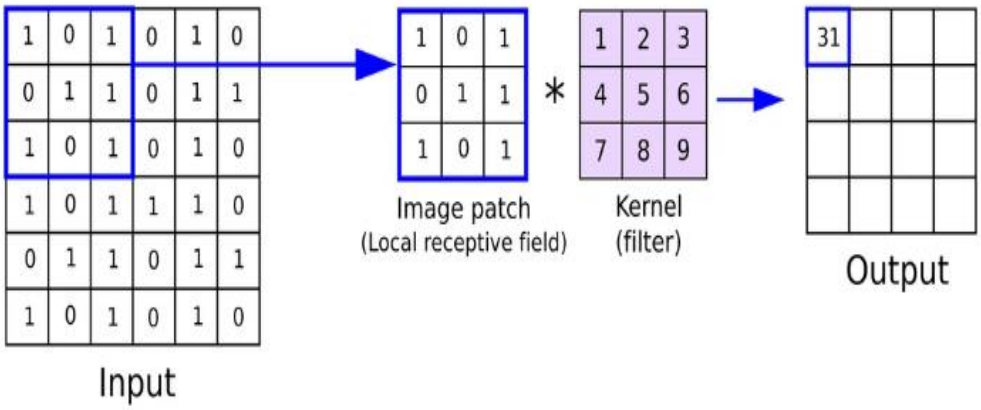

Рисунок 2.9 – Приклад операції згортки

Основною перевагою CNN над рекурентними нейронними мережами є можливість паралельної обробки, що дозволяє обробляти кілька послідовностей одночасно, що робить їх ефективними для обробки великих наборів даних і багатовимірних часових рядів.

Хоча згорткові нейронні мережі можливо використовувати для прогнозування часових рядів, існують обмеження [22].

Фіксована вхідна довжина є одним з недоліків, оскільки CNN вимагають вхідних послідовностей фіксованої довжини. Якщо дані часового ряду мають різну довжину, потрібно буде попередньо обробити їх, щоб усі послідовності мали однакову довжину. Ця попередня обробка може включати скорочення, доповнення або повторну вибірку, що потенційно може призвести до втрати інформації.

Згорткові нейронні мережі (CNN) спроєктовані для виявлення локальних шаблонів у даних, що знижує їх ефективність у прогнозуванні довгострокових залежностей. У таких випадках рекурентні нейронні мережі (RNN) є більш придатними.

3 ПРОГРАМНА РЕАЛІЗАЦІЯ

3.1 Опис набору вихідних даних

Для навчання, тестування та оцінювання нейромережевих моделей використовувався структурований набір даних, що відображає кількість зареєстрованих випадків захворювань (рисунок 3.1). Період охоплення та кількість записів визначалися вимогами до швидкості та точності навчання моделі. Наприклад, можна використовувати дані за період із 2022 до 2023 року або за весь доступний період, що дозволяє врахувати більше контекстної інформації, включно з довгостроковими залежностями та рідкісними подіями. Цей набір даних представлений у вигляді часового ряду, що дає змогу аналізувати динаміку розвитку захворювань.

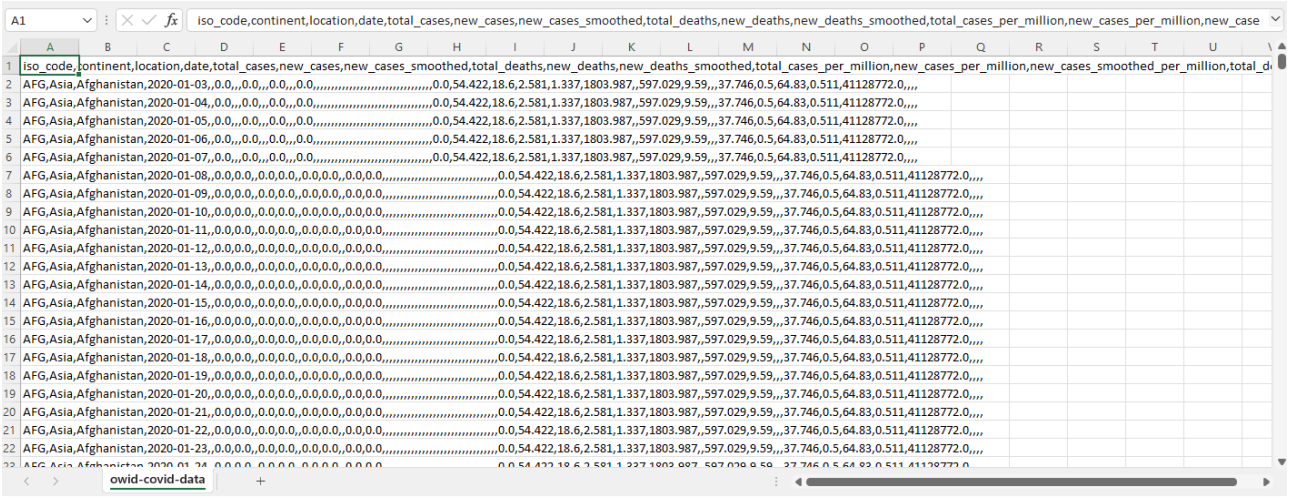

Рисунок 3.1 – Фрагмент початкових даних

У рамках дослідження було розглянуто дані, що стосуються захворюваності. Набір даних представлений у табличному форматі, що включає два основні стовпці. Перший стовпець date містить інформацію про дату вимірювань у форматі datetime, що дозволяє виконувати операції з

датами, такі як сортування, обчислення часових інтервалів та виявлення трендів. Дати впорядковані у хронологічному порядку. Другий стовпець містить дані, що описують прогнозовані параметри, зокрема кількість нових підтверджених випадків захворювань, летальних випадків і вакцинацій. Значення у цьому стовпці змінюються залежно від динаміки поширення захворювання, включаючи періоди спалахів, стабілізації та спаду.

Ця структура є базовою для підготовки даних, що використовуються під час навчання, тестування та оцінювання нейромережевих моделей. Набір даних (рисунок 3.2) є вихідним для аналізу часових рядів та розробки прогнозної моделі, здатної прогнозувати динаміку поширення захворювань. Дані завантажуються з файлу формату .xlsx, який містить хронологічно впорядковану інформацію про кількість зареєстрованих випадків.

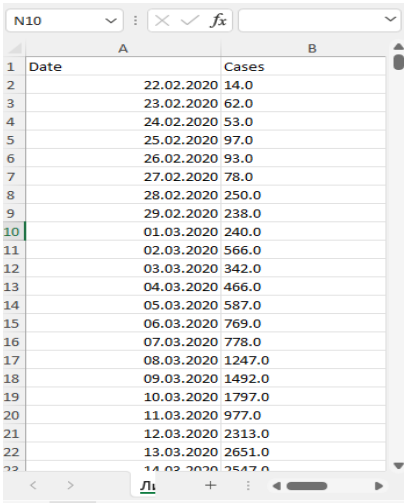

Рисунок 3.2 – Обрані дані для тренування моделі

Для підготовки даних до використання в моделі виконуються кілька етапів обробки. Спочатку дати перетворюються у формат datetime та використовуються як індекс таблиці. Після цього значення кількості випадків нормалізуються за допомогою методу MinMaxScaler, що приводить дані до діапазону [0, 1]. Це забезпечує стабільність і ефективність процесу навчання нейронної мережі.

Для використання рекурентної моделі дані розбиваються на підпослідовності фіксованої довжини, наприклад, 30 днів. Кожна підпослідовність містить історичні дані, що слугують входом для моделі, та одне наступне значення, яке модель має прогнозувати. Набір даних характеризується постійною частотою спостережень і відсутністю пропущених значень, що знижує потребу у складній обробці та корекції. Перед запуском моделі дані візуалізуються у вигляді графіків, що дозволяє проаналізувати динаміку захворювань, виявити тренди та сезонні коливання.

Після виконання прогнозування результати порівнюються з фактичними значеннями, що дає змогу оцінити точність моделі та її здатність відображати реальну динаміку процесу.

3.2 Очищення даних

Очищення даних є важливим етапом у підготовці інформації для подальшого аналізу чи моделювання. У цьому дослідженні використовувався набір даних, що містить інформацію про кількість випадків захворювань у різні дати. Вхідний набір даних зберігається у Excel-файлу (.xlsx) і включає стовпці Date та Cases. Оскільки якість початкових даних безпосередньо впливає на точність аналізу та прогнозування, було виконано ретельну обробку, зокрема усунення пропущених значень.

Перший етап включав виявлення пропущених або некоректних значень. Для цього використовувалися функції бібліотеки pandas мови програмування Python. Функція isnull() застосовувалася для ідентифікації записів із відсутніми значеннями. Пропуски в стовпцях Date та Cases могли бути спричинені помилками збору даних або відсутністю інформації за певні дати.

На другому етапі усувалися пропуски в стовпці Date. Рядки з некоректними або відсутніми значеннями були видалені за допомогою методу dropna().

Пропущені значення в стовпці Cases оброблялися кількома способами, зокрема шляхом заповнення середнім значенням або інтерполяцією. Після цього очищені дані сортувалися за стовпцем Date у порядку зростання, щоб забезпечити коректність часових рядів.

У результаті очищення (лістинг 3.1) було усунене всі пропущені та некоректні значення в стовпцях Date та Cases. Видалення або заповнення пропущених значень забезпечувало цілісність і надійність даних, що є важливим для моделювання та проведення аналізу.

Лістинг 3.1 – Фрагмент коду очищення даних

```
def clean_data(input_file, output_file):
    data = pd.read_excel(input_file)
    if 'Date' not in data.columns or 'Cases' not in
data.columns:
        raise ValueError("Файл має містити стовпці 'Date' і
'Cases'")
    data['Date'] = pd.to_datetime(data['Date'], errors='coerce')
    data['Cases'] = data['Cases'].interpolate(method='linear')
```

Для визначення характеру пропусків у наборі даних було створено теплову карту (рисунок 3.3), що відображає відсутні значення у стовпцях.

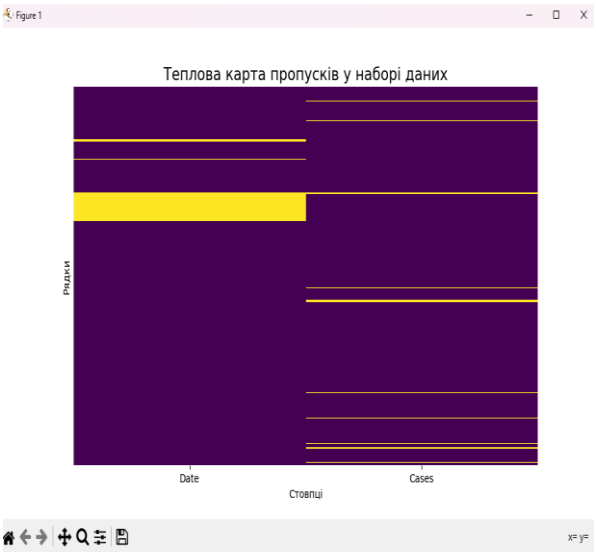

Рисунок 3.3 – Теплова карта пропущених даних

Теплова карта є ефективним інструментом для швидкої оцінки розподілу пропущених значень у наборі даних. Вона дозволяє виявити стовпці або рядки з найбільшою кількістю пропусків і ухвалити обґрунтоване рішення щодо їх подальшого усунення.

На тепловій карті пропущені значення позначаються окремим кольором, що забезпечує візуальну інтерпретацію масштабу проблеми та локалізацію ділянок, де вона виникає.

Методи очищення даних та аналізу пропусків були застосовані до початкових наборів даних, пов’язаних із прогнозуванням кількості вакцинацій, підтверджених летальних випадків, загальної кількості хворих та госпіталізованих.

3.2.1 Виявлення викидів

Викиди являють собою спостереження, значення яких суттєво відрізняються від інших у вибірці. Їх поява може бути зумовлена помилками у зборі даних, технічними проблемами або специфічними подіями, що відображають рідкісні явища. Наявність викидів може суттєво вплинути на результати аналізу, тому їх виявлення та належна обробка мають велике значення. У числових даних викиди визначаються як значення, що виходять за межі певного діапазону.

Одним із методів ідентифікації викидів є статистичний підхід, наприклад, метод міжквартильного розмаху (лістинг 3.2). Цей метод базується на аналізі розподілу даних за квартилями: значення, що виходять за межі діапазону, розрахованого на основі першого і третього квартилів із врахуванням міжквартильного розмаху, класифікуються як викиди.

Квартилі – це статистичні показники, які ділять набір даних на рівні частини з однаковою кількістю елементів у кожній. Вони використовуються для оцінки розподілу даних, виявлення аномалій і аналізу розкиду значень. Перший квартиль (Q1) розділяє найнижчі 25% даних від решти 75%. Медіана

(Q2) другий квартиль, або 50-й перцентиль. Третій квартиль (Q3) розділяє найвищі 25% даних від решти 75%.

Лістинг 3.2 – Фрагмент коду методу міжквартильного розмаху (IQR)

```
Q1 = data['Cases'].quantile(0.25)
Q3 = data['Cases'].quantile(0.75)
IQR = Q3 - Q1
lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR
outliers = data[(data['Cases'] < lower_bound) | (data['Cases'] >
upper_bound)]
cleaned_data = data[(data['Cases'] >= lower_bound) &
(data['Cases'] <= upper_bound)]
```

Метод Z-оцінки (лістинг 3.3) використовує середнє значення та стандартне відхилення. Значення, Z-оцінка яких перевищує 3 або -3, є викидами.

Лістинг 3.3 – Фрагмент коду Z-оцінки

```
mean = data['Cases'].mean()
std = data['Cases'].std()
data['Z-score'] = (data['Cases'] - mean) / std
outliers = data[abs(data['Z-score']) > 3]
cleaned_data = data[abs(data['Z-score']) <= 3]
```

Аналіз даних прогнозує не лише числові методи, а й використання графічних інструментів, які дозволяють візуалізувати характер розподілу даних, спрощуючи виявлення викидів та інших аномалій.

Scatterplot – це графік (рисунок 3.4), на якому кожна точка відповідає парі значень: одне значення по осі X (наприклад, дата), а інше – по осі Y (наприклад, кількість випадків захворювання). Scatterplot дозволяє відстежувати зміни даних у часі, демонструє тренди, сезонність та інші закономірності. У випадку дослідження поширення захворювань, дозволяє не лише побачити динаміку захворюваності, а й оцінити характер викидів.

Наприклад, окремі ізольовані точки, які значно відрізняються від решти даних, можуть вказувати на епідемічні спалахи або помилки у введенні даних.

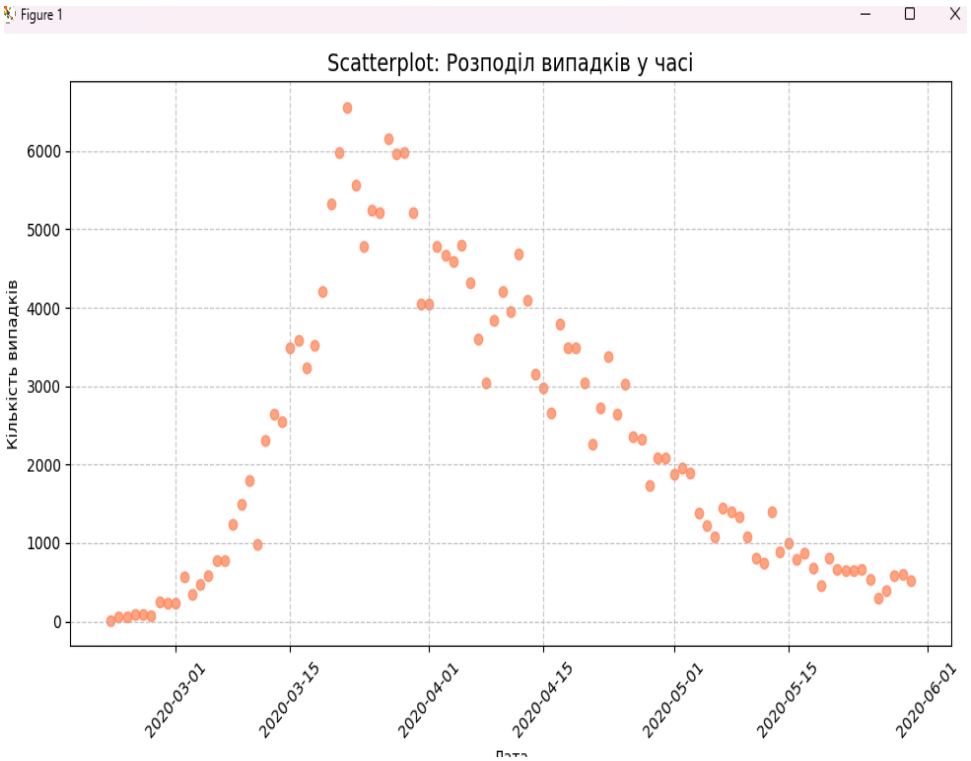

Рисунок 3.4 – Графік Scatterplot

Алгоритмічні методи є ефективним інструментом для виявлення викидів у даних, оскільки вони використовують математичні та статистичні алгоритми для аналізу структури даних і виявлення точок, що суттєво відрізняються від основного розподілу.

Описані підходи обробки викидів набувають важливості в ситуаціях, коли графічні або базові статистичні методи виявляються недостатньо ефективними для ідентифікації викидів. Одним із прикладів є методи кластеризації, такі як алгоритм k-means, який дозволяє групувати дані та ідентифікувати точки, що не належать жодному кластеру або значно віддалені від центрів кластерів (рисунок 3.5).

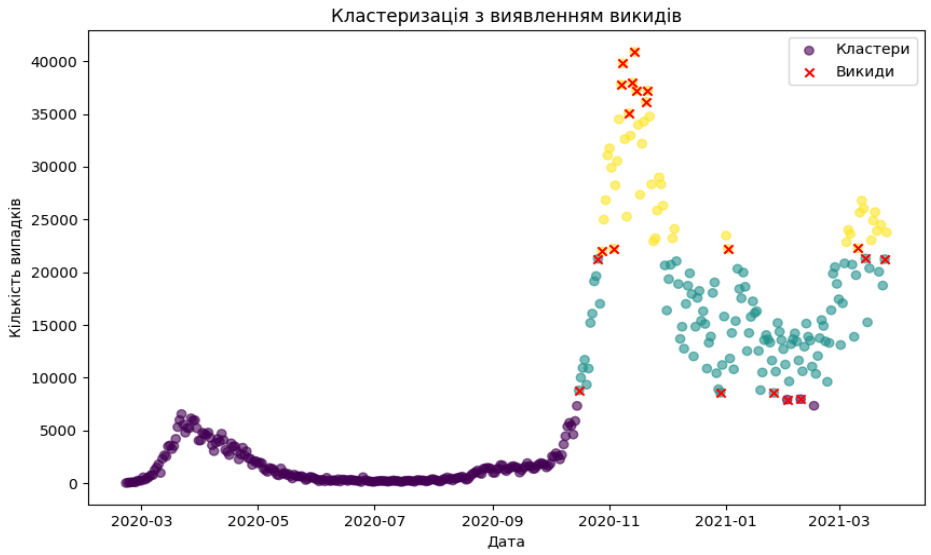

Рисунок 3.5 – Графічне зображення кластеризації

Дані поділяються на групи на основі їх подібності. Точки, які не належать жодному кластеру або знаходяться далеко від центрів кластерів, розглядаються як викиди.

Лістинг 3.4 – Фрагмент коду методу кластеризації

```
data.dropna(inplace=True)
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
scaled_data = scaler.fit_transform(data[['Cases']])
kmeans = KMeans(n_clusters=3, random_state=42)
clusters = kmeans.fit_predict(scaled_data)
data['Cluster'] = clusters
data['Distance_to_Center'] = np.linalg.norm(scaled_data -
kmeans.cluster_centers_[clusters], axis=1)
далеко від центру кластерів)
threshold = data['Distance_to_Center'].quantile(0.95)
```

Викиди у даних можуть суттєво вплинути на результати моделювання та аналізу. Для їх виявлення доцільно використовувати комбіновані підходи: статистичні, графічні та алгоритмічні методи.

3.2.2 Згладжування даних

Згладжування даних є процесом попередньої обробки, спрямованим на зменшення коливань і шуму в даних. Це один із важливих етапів підготовки інформації, який покращує її якість для навчання нейронних мереж, сприяє виділенню основних тенденцій і закономірностей, а також поліпшує візуальне сприйняття даних.

Рисунок 3.6 – Графічне зображення результату згладжування

Після очищення даних у часових рядах часто залишаються випадкові коливання, які можуть бути шкідливими у випадку перенавчання моделі. У таких випадках модель буде адаптуватися до цих випадкових варіацій замість того, щоб виявляти загальні закономірності, що знижує її здатність узагальнювати результати на тестових даних.

Одним із найпростіших і найпоширеніших методів згладжування є метод рухомого середнього (рисунок 3.6). Він прогнозує заміну кожного значення у часовому ряді середнім значенням певної кількості сусідніх точок.

Такий підхід дозволяє усунути випадкові коливання, зберігаючи основні характеристики даних. Рухоме середнє дозволяє ефективно згладжувати короткострокові коливання, зберігаючи при цьому загальний тренд даних. Таким чином, кожна точка у ряді стає менш залежною від випадкових коливань.

Після очищення та згладжування, що зменшує ризик перенавчання та підвищує точність моделі, дані готові для навчання.

3.3 Нейромережеві моделі

Для прогнозування поширення захворювань за допомогою нейромережевих методів використано мову програмування Python та інтегроване середовище розробки PyCharm Community. Для реалізації було використано бібліотеки TensorFlow і Keras. Keras – це високорівнева бібліотека для створення нейронних мереж, яка використовується у задачах машинного навчання. Моделі в Keras будуються шляхом об’єднання різних компонентів, таких як шари нейронної мережі, функції активації, оптимізатори та функції втрат.

Для навчання нейронної мережі необхідно підготувати вхідні дані. Дані завантажуються із зовнішнього файлу формату Excel за допомогою бібліотеки pandas, яка використовується для обробки табличних і структурованих даних. Її основні структури даних, такі як DataFrame і Series, дозволяють працювати з інформацією у вигляді таблиць. Завантажені дані включають хронологічну послідовність кількості випадків захворювання. Для зручності обробки дата встановлюється як індекс. Для підвищення ефективності роботи нейронної мережі числові дані масштабуються в діапазон [0, 1] за допомогою MinMaxScaler із бібліотеки sklearn.

Для навчання мережі формується набір вхідних послідовностей та відповідей. Послідовність охоплює кількість випадків за останні 7 днів, а вихід – кількість випадків на наступний день. Після цього дані розбиваються

на навчальну та тестову вибірки у співвідношенні 80:20. Дані можна розбивати на різні співвідношення залежно від потреби.

Для прогнозування обрано рекурентні нейронні мережи RNN та LSTM, а також згорткову нейронну мережу CNN. Дані розбиваються на послідовності – набори з 7 значень, які використовуються як вхід до мережі. Рекурентний шар RNN із, наприклад, 32 нейронами, який отримує на вхід послідовність тривалістю 7 днів. Вихідний шар має один нейрон і відповідає за прогнозування одного числа – кількості випадків захворювання на наступний день. Оптимізатором було обрано Adam, оскільки він є універсальним і ефективно працює з більшістю задач. Функція втрат – середньоквадратична помилка.

З метою реалізації підготовки даних для роботи з часовими рядами розроблена функція load_training_data (лістинг 3.5), яка виконує завантаження даних з файлу .xlsx.

Лістинг 3.5 – Приклад завантаження даних для навчання

```
def load_training_data():
    global scaler, X_train, y_train, X_test, y_test
    file_path = filedialog.askopenfilename(title="Виберіть файл
для навчання", filetypes=(("Excel Files", "*.xlsx"),))
    if not file_path:
        return
```

Після завантаження дані нормалізуються в діапазон [0, 1] за допомогою MinMaxScaler (лістинг 3.6). Масштабування є важливим етапом при роботі з нейронними мережами.

Лістинг 3.6 – Приклад підготовки даних для навчання

```
data['Date'] = pd.to_datetime(data['Date'])
    data.set_index('Date', inplace=True)
    scaler = MinMaxScaler(feature_range=(0, 1))
    data['Cases'] = scaler.fit_transform(data[['Cases']])
```

```
    X, y = [], []
    for i in range(len(data) - sequence_length):
        X.append(data['Cases'].values[i:i+sequence_length])
        y.append(data['Cases'].values[i+sequence_length])
    X, y = np.array(X), np.array(y)
    train_size = int(len(X) * 0.8)
    global X_train, y_train, X_test, y_test
    X_train, X_test = X[:train_size], X[train_size:]
    y_train, y_test = y[:train_size], y[train_size:]
```

Дані перетворюються у послідовності довжиною sequence_length для підготовки до використання в моделі. Вхідні послідовності X містять кілька значень, а вихідні Y – наступне значення після послідовності. Набір даних поділяється на навчальну та тестову вибірки. Тестова вибірка використовується для перевірки, щоб уникнути запам'ятовування даних.

Функція create_model (лістинг 3.7) реалізує рекурентну нейронну мережу для прогнозування часових рядів.

Лістинг 3.7 – Приклад створення моделі

```
def create_model():
    global model
    model = Sequential([
    SimpleRNN(50,activation='relu',
input_shape=(sequence_length, 1)), Dense(1)])
    model.compile(optimizer='adam', loss='mean_squared_error')
def train_model():
    if model is None:
        create_model()
    if X_train is None or y_train is None:
        load_training_data()
    history = model.fit(X_train, y_train, epochs=20,
batch_size=16, validation_data=(X_test, y_test))
    plt.plot(history.history['loss'], label='Втрати на
навчанні')
    plt.plot(history.history['val_loss'], label='Втрати на
валідації')
    plt.legend()
    plt.title('Процес навчання')
    plt.show()
    messagebox.showinfo("Навчання", "Модель успішно навчена!")
def make_prediction():
    if model is None:
        messagebox.showerror("Помилка", "Спочатку навчіть
модель!") return
```

Мережа містить один RNN шар. Активаційна функція ReLU забезпечує врахування лише позитивних значень, що дозволяє уникнути негативного впливу градієнтного затухання. Вихідний шар Dense складається з одного нейрона і призначений для генерації прогнозу. Оптимізатор Adam використовується для швидкого та стабільного навчання. У разі відсутності навчальних даних викликається функція load_training_data. Якщо модель ще не створена, викликається функція create_model. Модель навчається протягом 20 епох із розміром пакета 16. Для перевірки якості роботи моделі використовується 20% даних із тестової вибірки. Хоча RNN підходить для аналізу коротких залежностей, для більш складних часових рядів доцільно використовувати LSTM або CNN.

LSTM використовується з активаційною функцією ReLU. Шари LSTM забезпечують збереження довготривалих залежностей у часових рядах завдяки своїй унікальній структурі, що включає механізм гейтів. Вихідний шар Dense містить один нейрон, який відповідає за генерацію прогнозу.

Для моделі CNN використовується згортковий шар із 64 фільтрами та ядром розміром 3. Шар Flatten перетворює вихідні дані із згорткового шару у вектор для подальшої передачі у повнозв'язні шари та забезпечує сумісність між вихідним форматом згорткових шарів і повнозв'язними шарами.

Таким чином було реалізовано 3 типи нейронних мереж готових до навчання. Реалізація прогнозування захворюваності виконується за допомогою нейронної мережі, а також створення графічного інтерфейсу для спрощення взаємодії з користувачем (лістинг 3.8). У процесі прогнозування модель послідовно генерує значення на основі попередніх даних.

Лістинг 3.8 – Приклад прогнозування

```
predictions = []
for _ in range(n_days):
    next_prediction = model.predict(input_sequence)[0, 0]
    predictions.append(next_prediction)
    input_sequence = np.append(input_sequence[:, 1:, :],
```

```
[[[next_prediction]]], axis=1)
    predictions_rescaled =
scaler.inverse_transform(np.array(predictions).reshape(-1, 1))
    last_date = data.index[-1]
    forecast_dates = pd.date_range(last_date +
pd.Timedelta(days=1), periods=n_days)
```

Вхідна послідовність (input_sequence) оновлюється кожного разу, додаючи новий прогноз і видаляючи найстаріше значення. Це дозволяє моделі динамічно адаптуватися до нових даних. Отримані прогнози масштабуються зворотньо у початковий діапазон, що дозволяє зберегти відповідність реальним даним. Графік результатів включає історичні дані та прогнозовані значення, що візуалізується для зручного аналізу.

Лістинг 3.9 – Реалізація візуалізації прогнозованих даних

```
plt.figure(figsize=(12, 6))
    plt.plot(data.index,
scaler.inverse_transform(data['Cases'].values.reshape(-1, 1)),
label='Історичні даніе',
             color='blue')
    plt.plot(forecast_dates, predictions_rescaled,
label='Прогноз', color='red', linestyle='dashed')
    plt.title('Прогноз захворюваності')
    plt.xlabel('Дата')
    plt.ylabel('Кількість випадків')
    plt.legend()
    plt.show()
root = tk.Tk()
root.title("Прогнозирование заболеваемости")
```

Інтерфейс Tkinter (лістинг 3.9) дозволяє виконувати три основні дії: завантаження даних для навчання, запуск процесу навчання та генерація прогнозів. Кожна дія реалізована у вигляді кнопки, розташованої в головному вікні програми. Це забезпечує інтуїтивно зрозумілий підхід до роботи з моделлю навіть для користувачів, які не мають глибоких технічних знань. За допомого plt.plot виконується побудова графіка історичних даних та прогнозованих значень.

4 АНАЛІЗ РЕЗУЛЬТАТІВ ДОСЛІДЖЕННЯ

4.1 Параметри нейромережевих моделей

У рамках дослідження були використані три типи архітектур нейронних мереж: RNN, CNN та LSTM. Кожна з цих моделей має свої унікальні параметри, які впливають на процес навчання, точність прогнозування та здатність до узагальнення даних.

Для рекурентної нейронної мережі (RNN) основними параметрами є кількість нейронів у прихованому шарі, функція активації, розмір вхідної послідовності та оптимізатор.

Параметр batch_size визначає кількість зразків даних, що обробляються моделлю за один раз під час навчання. Під час кожної ітерації навчання модель оновлює ваги, використовуючи лише один пакет даних. Для невеликих датасетів можна використовувати великий batch_size, щоб за одну епоху проходити всі дані, але для прогнозування захворювань, де даних багато, треба експериментувати, щоб знайти оптимальний розмір, враховуючи доступну пам’ять і швидкість навчання. Один повний прохід через всі дані є epoch, складається з багатьох ітерацій, кожна з яких відповідає одному batch_size.

Величину горизонта прогнозування визначає параметр prediction_horizon – це період часу, на який модель або алгоритм робить прогнози.

Параметр units (для RNN, LSTM) визначає кількість нейронів у прихованому шарі. Прихований стан зберігає інформацію про попередні обчислення. Кількість units визначає розмір цього стану.

Параметр activation – математична функція, яка застосовується до виходу кожного нейрона в шарі нейронної мережі. Вона визначає, як нейрон активується або яку частину інформації передає на наступний шар.

Використовувалась функція втрат (loss), у якості якої обрана метрика mean squared error, яка обчислює середнє квадратичне відхилення між прогнозованим та фактичними значеннями.

Алгоритм, який використовується для налаштування ваг нейронної мережі з метою мінімізації функції втрат під час навчання обирається через параметр optimizer. Оптимізатор визначає, як і з яким кроком змінюються ваги моделі, щоб модель навчилась ефективно вирішувати завдання.

Параметр sequence_length визначає довжину вхідної послідовності, кількість часових точок, які використовуються для визначення наступного значення.

Нейромережеві моделі прогнозування часових рядів використовуються фільтри (параметр filters), це матриці, які застосовуються до вхідних даних для виділення специфічних ознак, наприклад, патерни. Виконують операцію згортки, переміщуючись по типу даних, і обчислюючи вихід, який потім передається на наступний шар. Розмір фільтра (kernel_size), який застосовується до вхідних даних під час операції згортки, визначає, скільки елементів з вхідного набору даних буде оброблятися одночасно.

4.2 Вплив гіперпараметрів моделі на результати прогнозування

Для дослідження впливу гіперпараметрів моделі на результати прогнозування було обрано: units, activation, optimizer для RNN, LSTM та filters, kernel_size для CNN. Щоб досягнути оптимальних результатів у прогнозуванні захворювань важливо правильно налаштувати гіперпараметри моделі. Неправильно налаштовані гіперпараметри можуть призвести до перенавчання або недонавчання.

Для аналізу прогнозів (таблиця 4.1) моделі було налаштовано такі значення параметрів роботи нейромережевої моделі: history – 60, prediction_horizon – 20, epoch – 30, dropout – 0,2, кількість прихованих шарів нейронної мережі – 1.

У якості метрики була обрана MSE (Mean Squared Error), що вимірює середнє значення квадратів різниць між прогнозними та фактичними значеннями в наборі даних. Якщо MSE близька до нуля, це вказує на те, що модель є адекватною, оскільки прогноз та фактичні значення близькі.

Таблиця 4.1 – Результати дослідження зміни гіперпараметрів RNN

№	units	optimizer	activation	MSE	Час навчання, с
1	2	adam	relu	0.6648	4.34
2	2	sgd	tanh	0.5346	4.10
3	4	adam	relu	0.4081	4.28
4	4	sgd	tanh	0.4232	4.15
5	8	adam	relu	0.3587	4.52
6	8	sgd	tanh	0.3701	4.15
7	16	adam	relu	0.3527	4.62
8	16	sgd	tanh	0.3561	4.33
9	32	adam	relu	0.3483	4.93
10	32	sgd	tanh	0.3448	4.71
11	64	adam	relu	0.3483	5.43
12	64	sgd	tanh	0.3512	4.99
13	128	adam	relu	0.3598	6.01
14	128	adam	tanh	0.3687	5.87
15	128	sgd	relu	0.3701	5.99

Результати дослідження демонструють, що збільшення кількості нейронів у шарі моделі сприяє підвищенню точності прогнозування. При збільшенні кількості нейронів (units) від 2 до 64 спостерігається поступове зменшення значення середньоквадратичної помилки (MSE). Найкращі результати досягнуті при 64 нейронах у шарі, де MSE досягає мінімального значення 0.3365. Однак подальше збільшення кількості нейронів до 128 призводить до зростання MSE, що може вказувати на перенавчання моделі.

Крім того, збільшення кількості нейронів супроводжується пропорційним збільшенням часу навчання моделі.

При порівнянні оптимізаторів встановлено, що Adam забезпечує кращу якість навчання в порівнянні з SGD. У більшості експериментів значення MSE при використанні Adam є меншим. Хоча SGD демонструє дещо швидший час навчання, різниця в швидкості є несуттєвою, що робить Adam більш ефективним вибором завдяки його вищій точності.

Аналіз використання різних функцій активації показав, що Tanh забезпечує кращі результати, ніж ReLU, значення MSE при використанні Tanh є нижчим у більшості експериментів. Водночас ReLU обробляє дані трохи швидше, що може бути перевагою в окремих випадках, наприклад, при необхідності прогнозування у режимі реального часу.

Оптимальною комбінацією гіперпараметрів виявлено конфігурацію з 64 нейронами у шарі, оптимізатором Adam та функцією активації Tanh. Ця конфігурація забезпечує найнижче значення MSE і прийнятний час навчання. Як альтернативу можна використовувати конфігурацію з 32 нейронами, яка також демонструє близьке до оптимального значення MSE, але має менший час навчання, що може бути доцільним у випадках, коли швидкість є пріоритетною.

Результати дослідження (таблиця 4.2) підтверджують, що збільшення кількості нейронів у шарі LSTM сприяє підвищенню якості моделі. Зокрема, зі зростанням параметра units спостерігається зменшення середньоквадратичної помилки (MSE).

Найбільше покращення якості прогнозування досягається при збільшенні кількості нейронів до 64. Наприклад, для 2 нейронів значення MSE залишається відносно високим, тоді як при 64 нейронах воно досягає мінімальних значень незалежно від використаної функції активації (Tanh або ReLU). Однак збільшення кількості нейронів до 128 не забезпечує суттєвого покращення MSE, водночас значно збільшуючи час навчання, що може свідчити про ризик перенавчання моделі.

Таблиця 4.2 – Вплив гіперпараметрів LSTM

№	units	optimizer	activation	MSE	Час навчання, с
1	2	adam	relu	0.5667	4.69
2	2	sgd	tanh	0.5495	5.29
3	4	adam	relu	0.4784	5.30
4	4	sgd	tanh	0.4232	5.69
5	8	adam	relu	0.3587	5.89
6	8	sgd	tanh	0.3701	6.10
7	16	adam	relu	0.2485	5.78
8	16	sgd	tanh	0.2411	6.35
9	32	adam	relu	0.2047	5.91
10	32	sgd	tanh	0.1845	6.58
11	64	adam	relu	0.1864	6.58
12	64	sgd	tanh	0.1767	6.58
13	128	adam	relu	0.1825	7.21
14	128	adam	tanh	0.1902	8.55
15	128	sgd	relu	0.1809	8.23

Аналіз оптимізаторів показав, що Adam демонструє стабільно високі результати для всіх конфігурацій, хоча SGD у деяких випадках забезпечує трохи нижчі значення MSE, особливо при більшій кількості нейронів. Проте Adam має перевагу в часі навчання, оскільки в більшості експериментів працює швидше за SGD.

Щодо функцій активації, Tanh майже у всіх випадках забезпечує кращу якість моделі порівняно з ReLU, оскільки значення MSE із Tanh є нижчим. Це особливо помітно при великій кількості нейронів, де Tanh демонструє стабільно вищу ефективність.

Результати дослідження роботи CNN (таблиця 4.3) підтверджують, що якість моделі значною мірою залежить від розміру ядра згортки (kernel size) та кількості фільтрів.

Таблиця 4.3 – Результати дослідження зміни гіперпараметрів CNN

№	kernel_size	filters	MSE	Час навчання, с
1	2	32	0.7012	3.01
2	2	64	0.6567	3.22
3	2	128	0.4977	3.24
4	3	32	0.4875	3.02
5	3	64	0.4826	3.19
6	3	128	0.4103	3.24
7	4	32	0.4567	3.01
8	4	64	0.4475	3.25
9	4	128	0.4002	3.27
10	5	32	0.4476	3.03
11	5	64	0.4965	3.19
12	5	128	0.4254	3.44

Зі збільшенням кількості фільтрів значення середньоквадратичної помилки (MSE) поступово зменшується, що особливо помітно для великих фільтрів, таких як 128. Наприклад, для ядра розміром 2 значення MSE суттєво знижується зі збільшенням кількості фільтрів з 32 до 128, демонструючи покращення якості моделі. Подібна закономірність спостерігається і для інших розмірів ядра, де більша кількість фільтрів сприяє зниженню похибки.

Розмір ядра згортки також має значний вплив на якість моделі. Більші ядра, такі як 3 або 4, демонструють менші значення MSE у порівнянні з ядром розміром 2. Найнижчі значення MSE досягаються при використанні ядра розміром 4 і 128 фільтрів, що свідчить про ефективність цієї конфігурації. Однак при збільшенні розміру ядра до 5 якість прогнозування дещо погіршується, особливо за меншої кількості фільтрів (32 або 64).

Час навчання моделей CNN залишається стабільним і майже не залежить від розміру ядра чи кількості фільтрів, коливаючись у межах 3–3,5

секунд навіть за зростання складності моделі. Оптимальною конфігурацією можна вважати розмір ядра 4 та 128 фільтрів, оскільки ця комбінація забезпечує мінімальне значення MSE за помірний час навчання.

Найкращі результати прогнозування були досягнуті шляхом оптимізації гіперпараметрів для різних моделей. Серед них LSTM із 64 нейронами, функцією активації Tanh та оптимізатором Adam забезпечила найнижчі значення MSE. Модель CNN показала високу точність і стабільний час навчання, особливо при використанні 128 фільтрів та ядра розміром 4. Хоча модель RNN може забезпечувати прийнятну якість прогнозування, для більш складних задач доцільніше застосовувати LSTM або CNN.

4.3 Візуалізація прогнозів

Для наочної оцінки ефективності моделей було виконано візуальний аналіз прогнозування, результати якого представлені на графіках. Кожна модель продемонструвала унікальні особливості прогнозування, які доцільно враховувати при виборі підходу прогнозування.

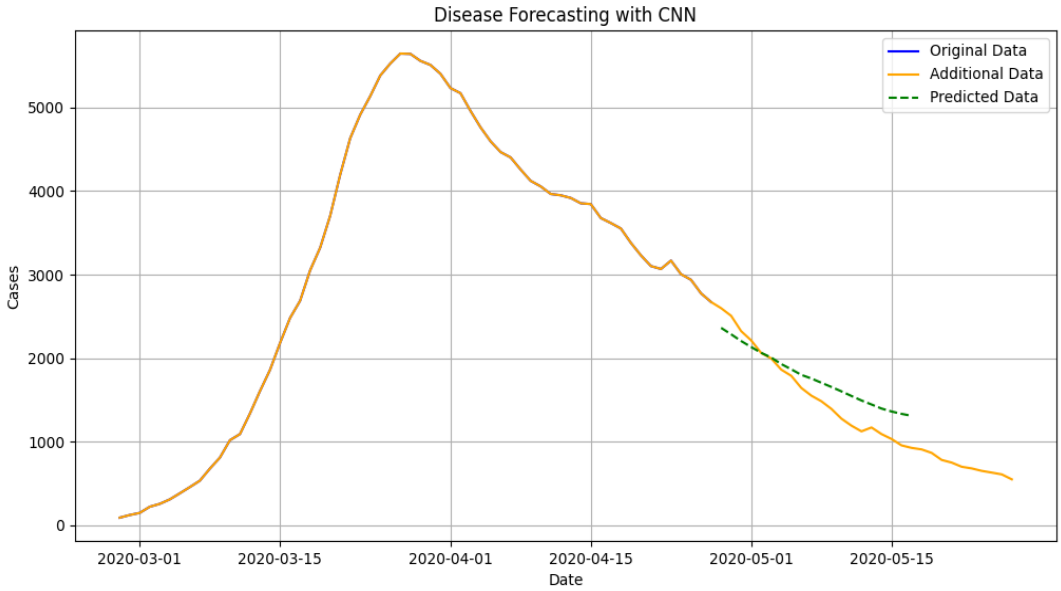

Рисунок 4.1 – Результат прогнозування за допомогою CNN

Модель CNN (рисунок 4.1) забезпечує адекватне прогнозування трендів даних. Проте, у деяких випадках спостерігається невелике відхилення від реальних значень, що може бути спричинено недостатньою кількістю навчальних даних або обмеженнями архітектури моделі.

Проведено порівняння прогнозів трьох моделей (рисунок 4.2) – RNN, LSTM та CNN. Оригінальні дані представлені жовтою лінією, результати прогнозування кожної з моделей позначені пунктирними лініями різних кольорів. Можна помітити, що модель LSTM демонструє більш стабільний і точний прогноз у порівнянні з іншими моделями, зокрема RNN, яка має тенденцію до більших коливань. CNN також демонструє конкурентний результат, однак її прогнози менш точні у деяких ділянках, порівняно з нейромережевою моделлю LSTM.

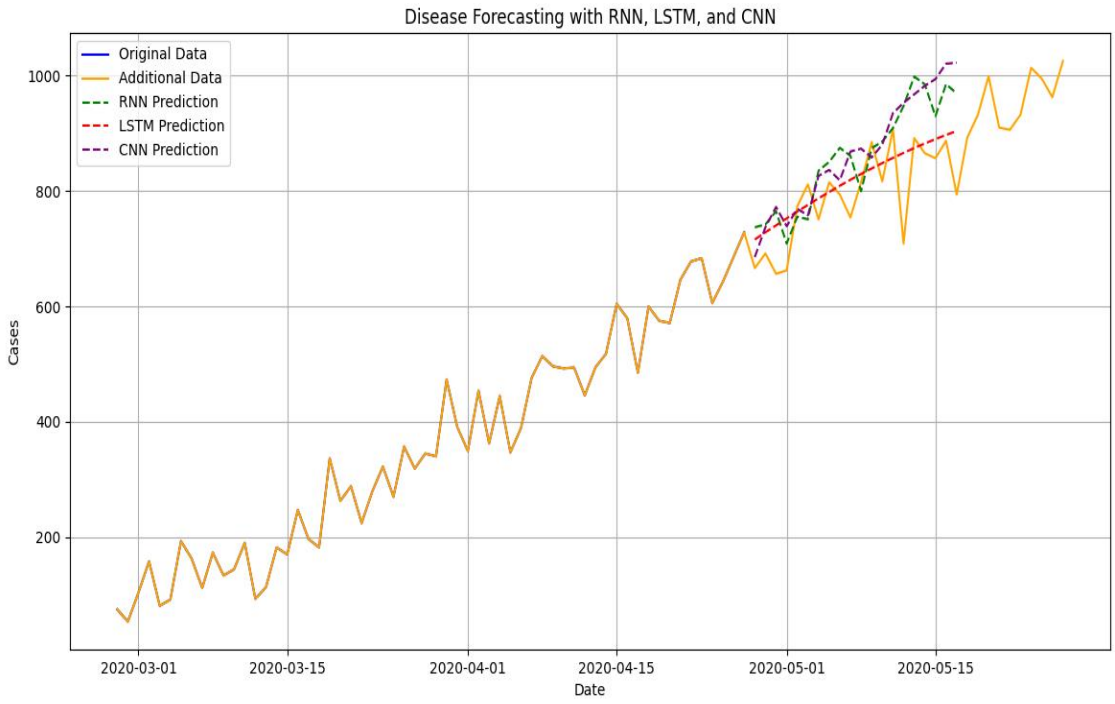

Рисунок 4.2 – Прогнозування на висхідному тренді захворюваності

Аналіз результатів (рисунок 4.3) демонструє, що RNN має обмеження у забезпеченні високої точності прогнозування. Натомість LSTM підтверджує свою ефективність у забезпеченні стабільності результатів прогнозів

(значення метрик помилок без суттєвих викидів). У свою чергу, CNN показує прийнятний рівень точності, однак у певних інтервалах спостерігаються відхилення прогнозованих значень від реальних даних.

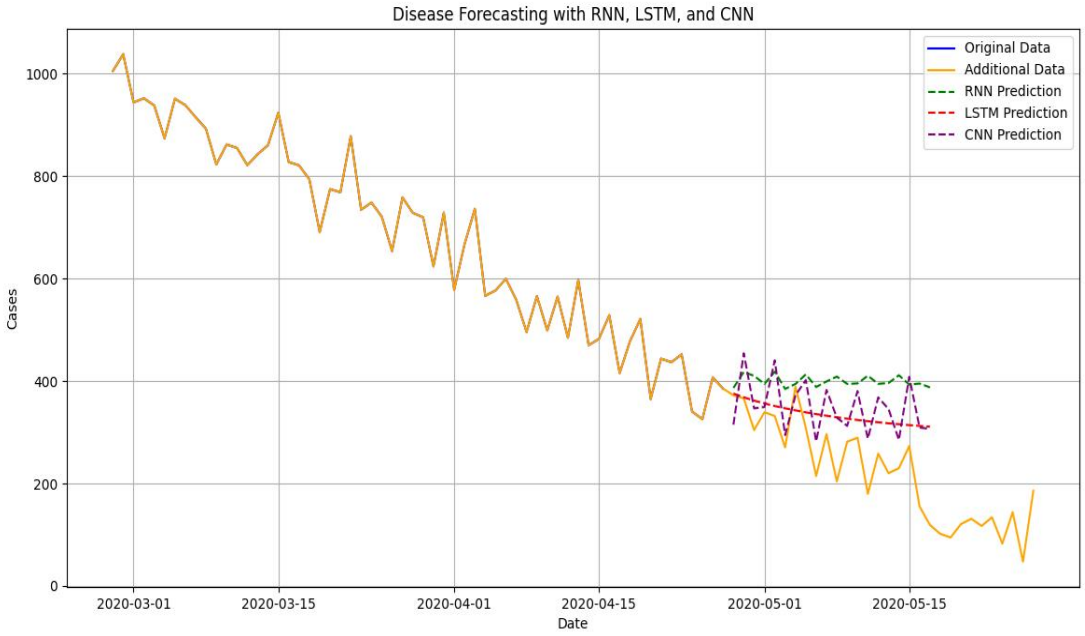

Рисунок 4.3 – Прогнозування на низхідному тренді захворюваності

На основі проведеного аналізу графічних даних можна зробити висновок, що модель LSTM є найбільш ефективною для задач прогнозування в рамках даного дослідження. Це підтверджується її здатністю забезпечувати мінімальні відхилення від фактичних даних і демонструвати стабільність результатів на різних ділянках досліджуваного інтервалу.

Модель CNN також може бути корисною, особливо в умовах, коли важливим є швидкість обчислень. Однак, її точність прогнозування дещо поступається LSTM, що може обмежувати її використання у задачах, де точність є більш важливою. У свою чергу, модель RNN показала нижчу якість прогнозування, що значно знижує її ефективність і доцільність застосування в складних задачах прогнозування.

ВИСНОВКИ

У роботі досліджено застосування нейромережевих методів для рішення задачі прогнозування поширення захворювань. Для навчання моделей використовувалися датасети з відкритих джерел, таких як ВООЗ, дослідницькі та аналітичні платформи. Створено та проаналізовано ефективність нейромережевих моделей [23], таких як CNN, RNN та LSTM, в задачах прогнозування поширення захворювань, де вихідні дані описані у вигляді часових рядів.

Нейромережеві методи показали свою ефективність у якості інструментів для прогнозування поширення захворювань. Здатність моделей ШНМ знаходити складні закономірності та взаємозв'язки у часових рядах дозволяє підвищити точність прогнозів порівняно із класичними статистичними підходами.

Наукова новизна полягає у використанні різних нейромережевих моделей для прогнозування поширення захворювань продовж пандемій захворювань, таких як ВІЛ-інфекція або коронавірусна інфекція COVID-19.

Для реалізації нейромережевих методів і аналізу результатів використовувалось кросплатформове інтегроване середовище розробки PyCharm Community та мова програмування Python.

Подальші дослідження можуть бути спрямовані на комбінуванні різних моделей для більш точного прогнозування у режимі реального часу, що має вирішальне значення для покращення процесу управління ризиками та контролю за поширенням захворювань.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Forecasting Disease Risk for Increased Epidemic Preparedness in Public Health [Електронний ресурс] – Режим доступу : www/ URL: https://pmc.ncbi.nlm.nih.gov/articles/PMC3196833/#:~:text=The%20objective%20of%20a%20forecasting,happening%20in%20the%20first%20place. – 20.10.2024 р. – Загол. з екрану.

2. Аналіз та прогнозування часових рядів [Електронний ресурс] – Режим доступу : www/ URL: https://drukarnia.com.ua/articles/analiz-ta-prognozuvannya-chasovikh-ryadiv-H6gg0. – 20.10.2024 р. – Загол. з екрану.

3. Approaches for time series analysis [Електронний ресурс] – Режим доступу : www/ URL: https://medium.com/analytics-vidhya/approaches-for-time-series-analysis-db27fe252c70. – 21.10.2024 р. – Загол. з екрану.

4. Stochastic Time Series Model [Електронний ресурс] – Режим доступу : www/ URL: https://www.math.hkbu.edu.hk/~hpeng/Math4826/Chapter4.pdf. – 21.10.2024 р. – Загол. з екрану.

5. The COVID-19 Forecast Hub [Електронний ресурс] – Режим доступу : www/ URL: https://covid19forecasthub.org. – 22.10.2024 р. – Загол. з екрану.

6. Forecasting of COVID-19 using deep layer Recurrent Neural Networks (RNNs) with Gated Recurrent Units (GRUs) and Long Short-Term Memory (LSTM) cells [Електронний ресурс] – Режим доступу : www/ URL: https://www.sciencedirect.com/science/article/pii/S0960077921002149#sec0002. – 20.10.2024 р. – Загол. з екрану.

7. Study on Prediction Model of HIV Incidence Based on GRU Neural Network Optimized by MHPSO [Електронний ресурс] – Режим доступу : www/ URL: https://pmc.ncbi.nlm.nih.gov/articles/PMC7176027/. – 25.10.2024 р. – Загол. з екрану.

8. Our World in Data [Електронний ресурс] – Режим доступу : www/ URL: https://ourworldindata.org/coronavirus. – 25.10.2024 р. – Загол. з екрану.

9. Artificial neuron [Електронний ресурс] – Режим доступу : www/ URL: https://www.techtarget.com/searchcio/definition/artificial-neuron. – 25.10.2024 р. – Загол. з екрану.

10. Explain like I’m five: Artificial neurons [Електронний ресурс] – Режим доступу : www/ URL: https://towardsdatascience.com/explain-like-im-five-artificial-neurons-b7c475b56189. – 26.10.2024 р. – Загол. з екрану.

11. What Is an Artificial Neural Network, and Why Does It Matter for AI [Електронний ресурс] – Режим доступу : www/ URL: https://www.coursera.org/articles/artificial-neural-network. – 26.10.2024 р. – Загол. з екрану.

12. Basic Understanding of Neural Network Structure [Електронний ресурс] – Режим доступу : www/ URL: https://medium.com/@sarita_68521/basic-understanding-of-neural-network-structure-eecc8f149a23. – 27.10.2024 р. – Загол. з екрану.

13. How neural networks are trained [Електронний ресурс] – Режим доступу : www/ URL: https://ml4a.github.io/ml4a/how_neural_networks_are_trained. – 20.10.2024 р. – Загол. з екрану.

14. 5 Techniques to Prevent Overfitting in Neural Networks [Електронний ресурс] – Режим доступу : www/ URL: https://www.kdnuggets.com/2019/12/5-techniques-prevent-overfitting-neural-networks.html. – 20.10.2024 р. – Загол. з екрану.

15. Recurrent Neural Networks cheatsheet [Електронний ресурс] – Режим доступу : www/ URL: https://stanford.edu/~shervine/teaching/cs-230/cheatsheet-recurrent-neural-networks. – 01.11.2024 р. – Загол. з екрану.

16. Goodfellow, I. Deep Learning (Adaptive Computation and Machine Learning series) [Текст] / I. Goodfellow, Y. Bengio, A. Courville. – The MIT Press, 2016. – 800 p.

17. What are the advantages and disadvantages of a Recurrent Neural Network (RNN) [Електронний ресурс] – Режим доступу : www/ URL:

https://aiml.com/what-are-the-advantages-and-disadvantages-of-a-recurrent-neural-network-rnn. – 02.11.2024 р. – Загол. з екрану.

18. What is LSTM – Long Short Term Memory [Електронний ресурс] – Режим доступу : www/ URL: https://www.geeksforgeeks.org/deep-learning-introduction-to-long-short-term-memory. – 02.11.2024 р. – Загол. з екрану.

19. LSTM Gates: Input, Forget, and Output [Електронний ресурс] – Режим доступу : www/ URL: https://www.linkedin.com/pulse/lstm-gates-input-forget-output-morteza-heidari-mzbdf. – 03.11.2024 р. – Загол. з екрану.

20. A Comprehensive Guide to Convolutional Neural Networks – the ELI5 way [Електронний ресурс] – Режим доступу : www/ URL: https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53. – 03.11.2024 р. – Загол. з екрану.

21. Unlocking the Potential of Convolutional Neural Networks (CNNs) in Time Series Forecasting [Електронний ресурс] – Режим доступу : www/ URL: https://thejaskiran99.medium.com/unlocking-the-potential-of-convolutional-neural-networks-cnns-in-time-series-forecasting-b2fac329e184. – 03.11.2024 р. – Загол. з екрану.

22. Disadvantages of CNN models [Електронний ресурс] – Режим доступу : www/ URL: https://sandeep-bhuiya01.medium.com/disadvantages-of-cnn-models-95395fe9ae40. – 05.11.2024 р. – Загол. з екрану.

23. Гуренко, Д. М. Модель на основі штучних нейронних мереж для прогнозування часових рядів епідеміологічних даних / Д. М. Гуренко, Г. С. Іващенко // Проблеми інформатизації: Тези доповідей дванадцятої міжнародної науково-технічної конференції. Баку – Харків – Бельсько-Бяла. – 21-22 листопада 2024. – Том 2, С. 82.