ПОЛТАВСЬКИЙ ДЕРЖАВНИЙ АГРАРНИЙ УНІВЕРСИТЕТ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ЕКОНОМІКИ, УПРАВЛІННЯ, ПРАВА ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНФОРМАЦІЙНИХ СИСТЕМ ТА ТЕХНОЛОГІЙ

Пояснювальна записка

до кваліфікаційної роботи на здобуття ступеня вищої освіти магістр

на тему: **«Прогнозування трафіку сайту Інтернет за допомогою нейронної мережі»**

Виконав: здобувач вищої освіти
за освітньо-професійною програмою
Інформаційні управляючі системи та технології спеціальності
126 Інформаційні системи та технології ступеня вищої освіти магістр
групи 126ІСТмд_21
Колєснік В. В.
Керівник: Слюсар В. І.
Рецензент: Муравльов В. В.

Полтава – 2023 року

ВСТУП

Актуальність теми кваліфікаційної роботи підтверджується необхідністю прогнозування трафіку сайту, який визначає ефективність онлайн-бізнесу по залученню аудиторії, вибору функціоналу хостингу та ін. Через те, що трафік сильно варіюється в залежності від часу доби, дня, сезону та інших чинників, прогноз доцільно виконувати за допомогою штучного інтелекту. Однак, питання реалізації прогнозу трафіку сайту на основі нейронної мережі недостатньо опрацьовано в теоретичному плані. Все це свідчить про актуальність теми роботи.

Зв'язок роботи з науковими програмами, темами. Робота відповідає дослідженням в рамках науково-дослідної роботи «Управління стратегією інноваційного розвитку підприємств в контексті підвищення їх конкурентоспроможності на аграрному ринку, сталого розвитку та забезпечення продовольчої безпеки держави» (2021 р.), що фінансувалась господарськими договорами із замовниками, Концепції розвитку штучного інтелекту в Україні (розпорядження Кабінету Міністрів України № 1787-р від 29.12.2021), тематиці досліджень навчально-дослідної лабораторії інтелектуальних систем, комп'ютерних мереж та інтернет речей кафедри інформаційних систем та технологій Полтавського державного аграрного університету.

Метою кваліфікаційної роботи є підвищення ефективності сайту Інтернет за рахунок прогнозування трафіку.

Завданнями кваліфікаційної роботи є:

- визначення особливостей трафіку сайту Інтернет;
- вибір архітектури нейронної мережі для обробки часових рядів;
- оцінка синтезованої моделі глибокого навчання нейронної мережі прогнозування трафіку сайту Інтернет;
- розробка рекомендації щодо використання нейронної мережі для прогнозування трафіку сайту Інтернет.

Об'єктом дослідження є процес прогнозування трафіку сайту Інтернет.

Предметом дослідження є точність нейронної мережі, що застосовуються для прогнозування трафіку сайту Інтернет.

Методами дослідження є аналітичний, інформаційно-пошуковий, методи синтезу та навчання нейронних мереж обробки часових рядів, робота з фреймворком Keras.

Інформаційна база кваліфікаційної роботи базується на ресурсах з даними про згорткові та рекурентні нейронні мережі, інструментарій для виконання глибокого машинного навчання та Transfer Learning.

Елементи наукової новизни роботи полягають у розробці архітектури нейронної мережі для прогнозування трафіку сайту Інтернет.

Практична значущість роботи полягає у розробці рекомендацій щодо використання нейронної мережі для прогнозування трафіку сайту Інтернет, які можуть бути використані для подальших досліджень за даною тематикою та при проектуванні сайту Інтернет.

Апробація результатів відбувалася в рамках V Міжнародної студентської конференції «Теоретичне та практичне застосування результатів сучасної науки» (жовтень 2023 р., м. Рівне), III-ої Міжнародної студентської конференції «Міждисциплінарні наукові дослідження та перспективи їх розвитку» (листопад 2023 р., м. Дніпро).

За результатами досліджень здійснено 2 публікації тез доповідей.

Структура кваліфікаційної роботи логічно пов'язана з завданнями досліджень і містить вступ, три розділи основної частини, висновки, список використаних джерел, додатки. Загальний обсяг пояснювальної записки кваліфікаційної роботи складає 68 сторінок формату А4. Вона містить 31 рисунок і 2 таблиці.

РОЗДІЛ 1
АНАЛІЗ ОСОБЛИВОСТЕЙ ПРОГНОЗУВАННЯ ТРАФІКУ САЙТУ

1.1 Особливості обробки часових рядів

Часові ряди оточують нас усюди, від прогнозування продажів до прогнозування трафіку та багато іншого. Найбільш поширеними їх прикладами є: фінансові дані; біометричні дані; виробничі процеси; прогноз погоди; аудіо; відео. Тобто, тип представлення даних є незамінним у таких секторах, як фінанси, охорона здоров'я, метеорологія, соціальні науки та інших, де чітке визначення часових закономірностей, тенденцій і циклічних коливань має вирішальне значення для прогнозування майбутніх значень і інформування процесів прийняття рішень. З ними працюють у завданнях від вимірювання океанських припливів до відстеження щоденного значення промислового індексу Доу Джонса (DJI) на момент закриття. Простим прикладом часових рядів є обсяг пасажиропотоку у річному обчисленні.

Формально, часові ряди – це дані, що упорядковані в хронологічному порядку. Вони є основою систем, підприємств та установ. База даних часових рядів є найшвидшою категорією баз даних за останні роки, і як традиційні, так і нові технологічні галузі генерують все більше і більше даних часових рядів. Обробляючи часові ряди, можна пробувати прогнозувати поведінку параметрів у часі. Прогнозування цих рядів є важливим завданням. Воно дозволяє отримати оцінку майбутніх спостережень заздалегідь, що несе вигоду різного характеру. Зазвичай, для цього використовують послідовне моделювання. Часові ряди моделюються за допомогою стохастичного процесу $Y(t), t \in N$, тобто обчислювальної сукупності випадкових величин. При цьому, цей процес залежить від часу. Тобто, часовий ряд містить інформацію про те, як змінюється параметр з часом. Наприклад, в умовах прогнозування спостерігаємо момент часу t, і намагаємось отримати оцінку $Y(t_0 + t)$, для $t_0, t \in N$, використовуючи тільки інформацію, доступну в

інтервалі часу $[t_0;t_0+t)$. Довжиною часового ряду найчастіше позначають кількість спостережень або час від початку остаточного процесу [1]. Точність при прогнозуванні майбутніх спостережень залежить кількості відомих нам спостережень, тобто від довжини напівінтервалу $[t_0;t_0+t)$.

Одним з бізнес-варіантів використання обробки часових рядів є прогнозування трафіку сайту Інтернет. Трафік сайту стосується вебкористувачів, які відвідують сайт. Вебтрафік вимірюється у відвідуваннях, які іноді називають «сеансами» за певний період часу, і є поширеним способом вимірювання ефективності онлайн-бізнесу щодо залучення аудиторії. Це дозволяє перевірити популярність сайтів або будь-яких конкретних вебсторінок цього сайту. Трафік сильно варіюється в залежності від того, який зараз час дня, який це день тижня та ін. Крім того, різке зростання кількості відвідувачів з соціальних мереж може бути пов'язане з використанням ботів, в тому числі, і на основі штучного інтелекту (AI) [2, 3]. Щоразу, коли будь-який користувач відвідує веб-сторінку, генерується відвідування сторінки, і будь-який користувач, якщо він підключений до Інтернету, завжди відвідає принаймні одну сторінку та може відвідати багато інших сторінок, якщо залишатиметься на зв'язку. В загальному випадку, для моніторингу трафіку потрібна така інформація, як загальна кількість відвідувачів, середнє значення переглядів сторінки на відвідувача, найвідоміші сторінки, середня відвідуваність відвідувачів і тривалість сторінки тощо. Це функції, які, зазвичай, використовуються для прогнозування трафіку сайту.

Від кількості та розміру обчислювальної потужності серверів, що підтримують платформу, залежить, який вона обсяг вебтрафіку здатна витримувати. Якщо трафік перевищує те, що сервери можуть обробити, сайт може показати помилку «404», що є не бажаним. Це змусить відвідувачів піти. Одне з вирішень цієї проблеми – збільшити кількість серверів. Однак зворотний бік рішення – причина може піти вгору, що знову ж таки небажано. З іншого боку, можна динамічно використовувати багато серверів на основі

історичних даних про обсяг відвідувачів або на основі історичних даних вебтрафіку. І це підводить нас до проблеми науки про дані, яка переважно прогнозує вебтрафік або кількість сеансів на основі історичних даних.

На даний час, методи аналізу часових рядів допомагають зрозуміти поведінку та зміну часових даних та виявити приховані тенденції та закономірності. Існують основні методи аналізу часових рядів: стаціонарність, спектральний аналіз та автокореляція.

Стаціонарність – це властивість часового ряду, яка означає, що його середні та стандартні відхилення не змінюються з часом. Якщо часовий ряд є стаціонарним, його можна легко аналізувати та прогнозувати. Нестаціонарний часовий ряд може мати тренд (постійне зростання або падіння), циклічність (повторення циклів) або сезонність (повторення певних подій у різні пори року). Один із способів перевірки стаціонарності часових рядів – це використання графічних методів, таких як графік часових рядів, графік автокореляції та графік приватної автокореляції. Графік часового ряду дозволяє візуалізувати зміни значень ряду з часом, графік автокореляції показує кореляцію між значеннями ряду в різні періоди часу, а графік автокореляції враховує кореляцію тільки між двома значеннями, пропускаючи всі проміжні значення.

Спектральний аналіз дозволяє досліджувати частотну складову часового ряду. Він використовується для виявлення прихованих циклічних патернів у часових рядах. Для цього, ряд розбивається на сигнали різних частот, а потім аналізується спектр цих частот. Спектральний аналіз часто використовується для аналізу фінансових ринків, у яких ціни можуть змінюватись в залежності від часу та частоти.

Автокореляція – це міра кореляції між значеннями поряд із різницею у часі. Якщо часовий ряд має високу автокореляцію, це означає, що значення ряду в різні періоди мають сильний зв'язок між собою. Автокореляцію можна обчислити за допомогою функції кореляції Пірсона, яка обчислює кореляцію між двома змінними. Для часових рядів це обчислення кореляції між

значеннями ряду у різні періоди часу. Якщо автокореляція є значною, це може означати наявність тренду, циклічності чи сезонності у ряді.

В свою чергу, існує кілька методів прогнозування часових рядів, включаючи методи екстраполяції, SARIMA та методи машинного навчання (регресія, нейронні мережі).

Методи машинного навчання (Machine Learning, ML) – це методи прогнозування, які використовують дані минулих значень ряду як вхідні дані для моделі навчання. Регресія – це метод ML, який моделює залежність між незалежними (вхідними) та залежними (вихідними) змінними. У разі прогнозування часових рядів, незалежними змінними будуть значення ряду у минулих періодах, а залежною змінною – майбутні значення ряду. Нейронні мережі – це складніші методи ML, які використовують алгоритми обробки інформації, подібні до тих, які використовують мозок [4]. Нейронні мережі можуть бути використані для прогнозування часових рядів і, зазвичай, мають більш високу точність, ніж методи екстраполяції.

Історично, статистичні методи, такі як ARIMA, ETS, MSTL, Theta та CES, надійно використовувалися в різних областях. За останнє десятиліття такі моделі машинного навчання, як XGBoost і LightGBM, набули популярності, демонструючи багатообіцяючі результати як у публічних конкурсах, так і в практичних застосуваннях. Однак із появою глибокого навчання відбулася зміна парадигми аналізу часових рядів. Методи глибокого навчання стали популярними в академічних колах і для великомасштабного промислового прогнозування [5].

Враховуючи глобальний підхід, методи глибокого навчання пропонують значні переваги перед статистичними локальними методами з точки зору масштабованості, гнучкості та потенційної точності. Крім того, їх здатність вивчати складні залежності даних ефективно обходить потребу в розробці складних функцій, необхідних для інших глобальних методів, таких як LightGBM або XGBoost. Отже, моделі часових рядів на основі глибокого навчання спрямовані на спрощення конвеєра прогнозування і підвищення

масштабованості. Їх здатність обробляти великі обсяги даних і фіксувати довгострокові залежності робить їх вигідними для складних завдань прогнозування в епоху постійного зростання обсягів даних.

Щодо переваги підходів до глибокого навчання спільнота прогнозистів наразі розділилася. Єдиний підхід ще належить створити. Останнім часом ці різні парадигми дедалі більше кидають виклик одна одній, ставлячи під сумнів корисність, точність і складність нових розробок. Незважаючи на успіх архітектур глибокого навчання в інших сферах, деякі практики часових рядів продемонстрували, що деякі запропоновані інновації в цій галузі не відповідають їхнім претензіям чи очікуванням.

Однак, думки академічних дослідників і практиків щодо цих обіцянок розходяться. Різні дослідники та практики заперечували основне припущення про підвищену точність, наводячи докази того, що простіші моделі перевершують більш складні підходи; з меншою вартістю та складністю. Навпаки, деякі лідери галузі повідомляють, що підхід до глибокого навчання покращив їхні результати та спростив канали аналізу [6].

У поточному історичному контексті, коли незаперечні переваги моделей глибокого навчання для NLP і CV, слід зазначити, що область аналізу часових рядів залишається скептичною щодо продуктивності методів нейронного прогнозування. Скептицизм може виникати через кілька чинників.

1. Невідповідні або погано визначені параметри оцінки: на відміну від інших галузей, які отримали вигоду від впровадження достатніх тестових наборів даних (ImageNet) для CV, загальнодоступні набори даних для часових рядів не мають необхідного масштабу та об'єму, щоб методи глибокого навчання могли досягти успіху.

2. Неоптимальні моделі: враховуючи обмежені та специфічні набори даних, навіть добре продумані архітектури глибокого навчання можуть мати проблеми з узагальненням або вимагати значних зусиль для пошуку оптимальних налаштувань і параметрів.

3. Крім того, відсутність стандартизованих великомасштабних наборів даних, які відповідають вимогам методів глибокого навчання, також може перешкоджати прогресу в цій галузі. У той час як інші галузі отримали вигоду від еталонних наборів даних і чітких оціночних показників, співтовариство часових рядів все ще потребує розробки таких ресурсів для сприяння інноваціям і перевірки нових методів.

В цілому, до переваг методів аналізу та прогнозування часових рядів можна віднести:

– дозволяють оцінити та виявити тенденції та циклічність часового ряду, що може призвести до більш точного прогнозування майбутніх значень;

– дозволяють аналізувати кореляцію між даними та виявляти залежності між ними;

– дозволяють визначити вплив різних чинників на зміни часового ряду.

Однак, методи аналізу (прогнозування) часових рядів мають недоліки:

– недостатня кількість даних може суттєво обмежити точність прогнозів;

– незважаючи на складання щодо точних прогнозів на основі історичних даних, випадкові фактори та позапланові обставини можуть призвести до невірного прогнозу;

– методи аналізу часових рядів можуть бути недостатніми, якщо дані мають сезонність і цей фактор не був врахований в аналізі.

Таким чином, сучасне теоретичне та практичне розуміння часових рядів ще не досягло такого рівня консенсусу серед практиків, який би віддзеркалював широке визнання генеративних моделей в інших фундаментальних сферах життя людини, таких як мова та сприйняття. Наукова спільнота все ще розділена в оцінці ефективності глибокого навчання для завдань прогнозування. Зусилля в науці про прогнозування не відповідають обіцянкам справді універсальних попередньо навчених моделей.

1.2 Використання нейронних мереж для обробки часових рядів

Одна з переваг нейронних мереж – можливість навчання –мереж перед традиційними алгоритмами. Після навчання мережа здатна передбачити майбутнє значення певної послідовності на основі кількох попередніх значень та/або якихось існуючих на даний момент факторів. Слід зазначити, прогнозування можливе лише тоді, коли попередні зміни справді певною мірою визначають майбутнє. Вдосконалення базових моделей для часових рядів можуть потенційно відкрити нову главу в цій галузі, сприяючи більш глибокому розумінню часових даних і підвищуючи точність і ефективність прогнозів, як одно-, так і багатосерійних часових рядів (рис. 1.1).

Найбільш придатними для цього завдання вважались кілька архітектур:

– повнозв'язані (Dense, іноді застосовують назву MLP);

– одновимірна згортка (Conv1D) – CNN;

– рекурентна (в тому числі, її різновиди: LSTM і GRU) – RNN;

– складні змішані мережі (наприклад, TCN та ін);

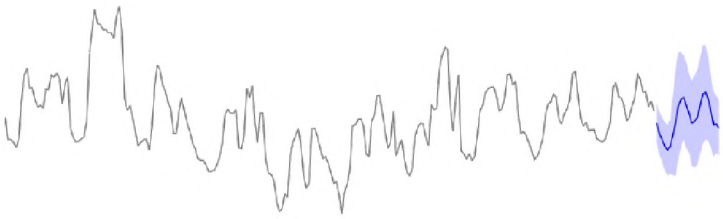
– трансформери (наприклад, TimeGPT) [7].

Моделі прогнозування глибокого навчання стали помітною областю досліджень завдяки їхньому успіху в нещодавніх відомих змаганнях, зокрема [8], і їх застосуванню до масштабних завдань у галузі. [5] представляє комплексний огляд і систематику моделей нейронного прогнозування та їх застосування.

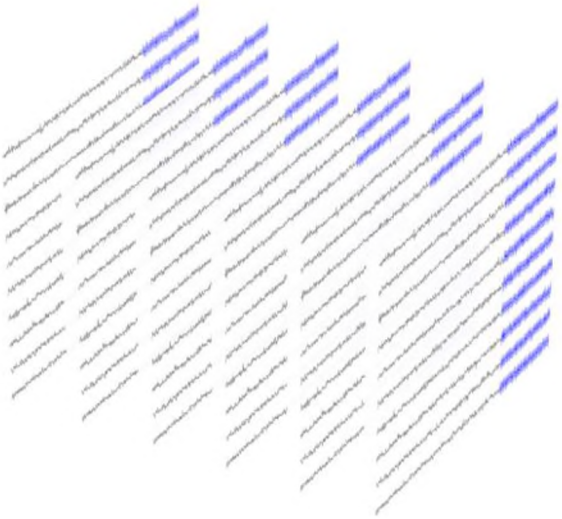
Початковий успіх у прогнозуванні часових рядів глибокого навчання став результатом адаптації апробованих архітектур, а саме RNN і CNN, спочатку розроблених для NLP і CV, відповідно. RNN слугували основою для таких популярних моделей, як DeepAR [9] для імовірнісного прогнозування та ESRNN [10, 11].

Зараз вони є популярним будівельним блоком, оскільки використовують такі моделі, як DPMN [12] і TimesNet [13]. Завдяки низьким

обчислювальним витратам і ефективності також часто використовуються мережі прямого зв'язку, зокрема N-BEATS [14] і NHITS [15].



а)



б)

умовні познаки:

а) – односерійне прогнозування;

б) – багатосерійне прогнозування.

Рисунок 1.1 – Варіанти прогнозування часових рядів

Потенціал основних моделей, а саме великомасштабних моделей, попередньо навчених на великому наборі даних і пізніше налаштованих для конкретних завдань, залишається відносно недостатньо вивченим для завдань прогнозування часових рядів. Основні моделі покладаються на свої можливості для узагальнення між доменами, зокрема в нових наборах даних, які не були доступні під час навчання. Відповідно, перенесення навчання як здатність застосовувати знання, отримані з одного завдання, для вирішення нових завдань [16].

Трансферне навчання (Transfer Learning, TL) стосується попереднього навчання моделі на, зазвичай, великому вихідному наборі даних, щоб покращити свою продуктивність у новому завданні прогнозування з цільовим набором даних (рис. 1.2). Серед переваг такого підходу в інтересах прогнозування часових рядів слід виділити кілька аспектів.

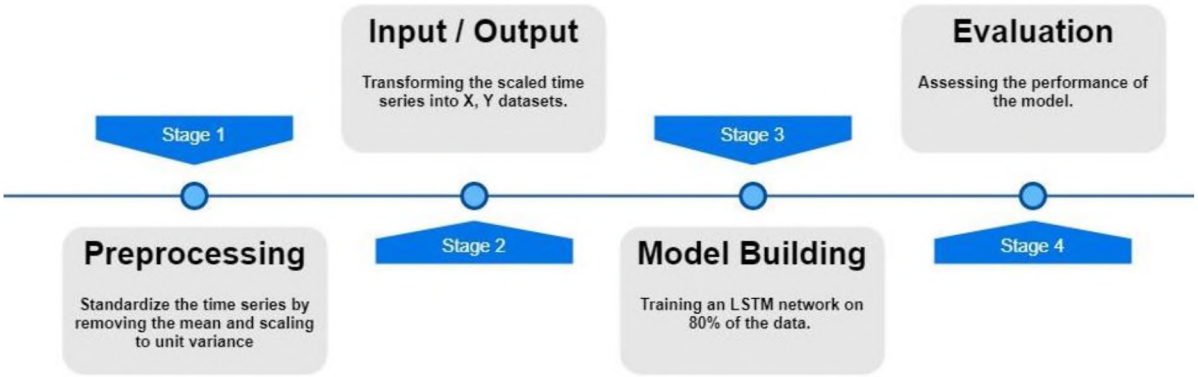


Рисунок 1.2 – Приклад застосування TL [17]

TL дозволяє використовувати заздалегідь навчальні моделі на великих наборах даних, що зменшує час і ресурси, необхідні для навчання нових моделей з нуля. Моделі, навчені на широких і різноманітних наборах даних, часто демонструють кращу продуктивність завдяки більш широкому «опиту» та покращеній узагальнюючій здатності. У сценаріях, де доступ до великих обсягів даних обмежений, трансферне навчання може допомогти, так як моделі навчаються на вже існуючих, обширних датасетах. Моделі, навчені на одних типах даних, можуть бути адаптовані до різних завдань прогнозування тимчасових рядів, що робить підхід більш універсальним. Скорочення часу, необхідного для розробки та налаштування нових моделей прогнозування, дозволяє швидше переходити до практичного застосування. Трансферне навчання дозволяє передавати знання з одного домену (наприклад, фінансовий) в інший (наприклад, здоров'я), що збільшує потенціал застосування моделей у різних сферах. Моделі можуть бути доопрацьовані на специфічних даних, що покращує їх здатність вирішувати унікальні і специфічні завдання прогнозування.

Ці переваги роблять TL потужним інструментом у сфері прогнозування часових рядів, особливо в умовах обмеженості даних і ресурсів. При цьому розглядаються два випадки трансфертного навчання:

– нульове навчання – попередньо навчена модель безпосередньо передається для вирішення нового завдання прогнозування без повторного навчання її параметрів на новому наборі даних [18];

– тонке налаштування – модель додатково донавчається на новому наборі даних (починаючи з попередньо навчених параметрів).

1.3 Створення датасету для прогнозування трафіку сайту

При створенні датасету у завданні прогнозування часового ряду, який відповідає трафіку сайту, потрібно виконати послідовність. Для збору даних про трафік можна використовувати інструменти вебаналітики (наприклад, Google Analytics [19]), щоб зібрати дані про відвідування сайту. В жданому випадку, знадобляться дані про кількість відвідувань (сесій) на сайті за певний період часу (наприклад, щодня, щотижня або щомісяця).

Надалі проводиться структурування даних, наприклад, їх подання у вигляді таблиці, де кожен рядок відповідає одному часовому інтервалу (наприклад, одному дню), а стовпці містять різні метрики, такі як кількість відвідувань, тривалість сесій, коефіцієнт відмов та інші.

Якщо даних мало, то виконується їх збагачення даних. Датасет доповнюється додатковою інформацією, яка може впливати на трафік, наприклад, маркетинговими кампаніями, змінами на сайті, сезонністю, святами тощо.

Надалі доцільно здійснити очищення даних, під час якого видаляють або коригують аномалії та викиди даних, оскільки вони можуть спотворити результати прогнозування.

Якщо дані мають великі відмінності в масштабах, то проводиться нормалізація даних (іноді, стандартизація).

Після цього датасет розділяють на навчальну та тестову вибірки. Навчальна вибірка використовується для налаштування моделі прогнозування, а тестова – для оцінки її ефективності.

Часові мітки у датасеті повинні мати правильний формат і є однорідними (наприклад, всі дати представлені в тому самому форматі).

Після того, як датасет підготовлений, його можна використовувати для навчання нейронних мереж.

Завдання отримання вхідних образів на формування навчальної множини завдання прогнозування часових рядів передбачає використання методу «вікна».

Цей метод має на увазі використання «вікна» з фіксованим розміром, здатного переміщатися за тимчасовою послідовністю історичних даних, починаючи з першого елемента, і призначені для доступу до даних тимчасового ряду, причому «вікно» розміром N, отримавши такі дані, передає на вхід нейронної мережі елементи з 1 по $N-1$, а $N-$ий елемент використовується як вихід.

Надалі більш детально розглянемо формування датасету. Даний якийсь часовий ряд, на основі якого бажаємо побудувати навчання нейронної мережі. На прикладі (рис. 1.2) показано дані зміни ціни акцій (дата, ціна відкриття угоди, максимальна ціна, мінімальна ціна, ціна закриття та розмір).

Параметр «VOLUME» на графіку часового ряду, зазвичай, відноситься до обсягу продажу, торгів або інших активностей за певний період. Він допомагає визначити тенденції, патерни та аномалії даних, що може бути корисно для прийняття рішень.

Наприклад, для сайтів або програм параметр «VOLUME» може стосуватися кількості відвідувань або активних користувачів за певний період часу, що важливо для оцінки популярності та ефективності веб-сайту або програми.

Таким чином, для навчання використовуватиметься лише ціна закриття угоди. Задаємо довжину xLen = 500 значень (довжина відрізка, який аналізуємо). З 1 до 500 значення будуть xTrain 1-го елемента, значення 501 буде yTrain 1-го елемента і т. д. по всій довжині вибірки (рис. 1.3).

DATE	TIME	OPEN	MAX	MIN	CLOSE	VOLUME
11.01.2023	10:30:00	1744.32	1749	1740	**1743**	14849
11.01.2023	10:31:00	1743.01	1750	1742.13	**1746.04**	23797
11.01.2023	10:32:00	1740.05	1750	1739.8G	**1739.8G**	17G94
11.01.2023	10:33:00	1740.01	1740.01	1736.8	**1737.83**	6207
11.01.2023	10:34:00	1737.83	1740.26	1737.11	**1739**	3894
11.01.2023	10:35:00	1740	1742	1738.06	**1739**	6901
11.01.2023	10:36:00	1738.99	1739.97	1736.8	**1738**	7271
11.01.2023	10:37:00	1738	1738.68	1736.8	**1737.6**	1203
11.01.2023	10:38:00	1737.98	1738	1736.8	**1736.8**	4303
11.01.2023	10:39:00	1736.8	1737.9	1736.6	**1737.59**	3968
11.01.2023	10:40:00	1736.69	1739	1736.62	**1739**	3932
11.01.2023	10:41:00	1738.95	1741.7	1738.24	**1741**	13044
11.01.2023	10:42:00	1740.99	1741.96	1738	**1738**	5934
11.01.2023	10:43:00	1739.7	1739.7	1737.22	**1737.9**	2719
11.01.2023	10:44:00	1737.86	1739.21	1736.83	**1739.09**	8094
11.01.2023	10:45:00	1739.09	1742	1738.9	**1742**	5160
11.01.2023	10:46:00	1741.89	1743.12	1741	**1743**	3591
11.01.2023	10:47:00	1743	1744	1741.08	**1741.08**	6803
11.01.2023	10:48:00	1742	1743.99	1741	**1741.02**	4765

Рисунок 1.2 – Формування вибірок датасету

При цьому, xTrain і xVal не повинні перетинатися, і між ними має бути відстань ≥ xLen. Якщо тренувальні дані та перевірочні будуть перетинатися,

нейронна мережа видаватиме недійсні показання точності. Надалі виконується масштабування. Наприклад, розглянемо вже знайомий часовий ряд. Зазвичай, для роботи з вартості акцій є п'ять наборів даних (закриття, відкриття, максимум, мінімум та обсяг торгівлі).

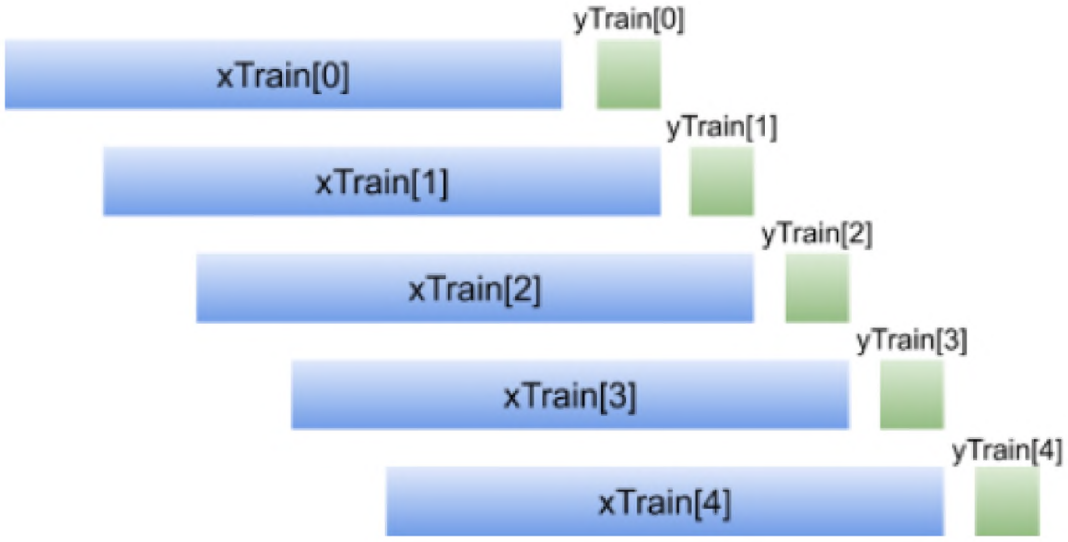


Рисунок 1.3 – Приклад формування вибірок yTrain

Якщо перші чотири можуть мати той самий порядок, то обсяг торгівлі може бути іншим, що погіршить роботу нейронної мережі. Тобто, нормування використовують для приведення даних до загального вигляду та порядку. Завдяки нормалізації даних на вхід мережі подаватимуться не самі значення, а різниця між послідовністю значень. Так мережі працюють набагато краще. У часових рядах ми можемо нормувати xTrain і yTrain. Для цього застосовують методи StandardScaler, MinMaxScaler:

– N (0, 1) – StandardScaler;
– [-1; 1] – MinMaxScaler;
– [0; 1] – MinMaxScaler.

StandardScaler наводить дані з параметрами: середнє дорівнює 0 і середньоквадратичне відхилення дорівнює 1. MinMaxScaler може мати два діапазони: $[-1; +1]$ та $[0; 1]$. Дані нормують як усі одразу, так і стовпці окремо. Для кожного нормування важливо підібрати правильну функцію

активації на вихідному шарі. Для стандартного Scaler використовують linear, оскільки він видає результат діапазоні $(-\infty; +\infty)$. Для MinMaxScaler з діапазоном $[-1; +1]$, крім Linear, використовують Tanh, а з діапазоном $[0;1]$ – Sigmoid та ReLu.

Автокореляція – це основна проблема часових рядів. Працюючи з часовими рядами ми використовуємо такі функції помилки, як Mean Square Error (MSE) і Mean Absolute Error (MAE), які визначають різницю між двома значеннями. Коли маємо справу з часовими рядами і послідовність даних має невелике відхилення, то мережа починає «хитрувати» і видає передбачене значення, що дорівнює попередньому (рис. 1.4).

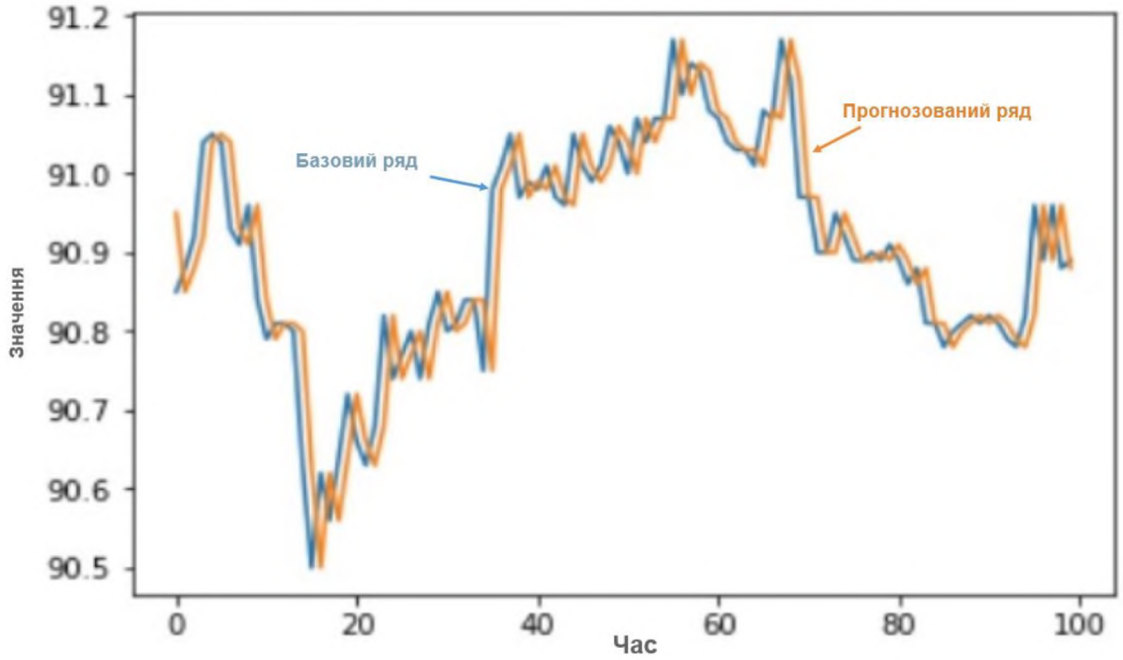


Рисунок 1.4 – Автокореляція прогнозування

Передбачення точно повторює контур базового ряду, але відстає на один крок. При цьому значення помилки мережі буде дуже низьким, а насправді прогноз буде незадовільним. Аналізуючи графік кореляції, можна визначити відхилення (рис. 1.5). Основне завдання у тому, щоб графік кореляції наближався до еталонного. Нейронні мережі можуть досягти гарних результатів у прогнозуванні. Прогноз набагато точніше відображатиме поведінку часового ряду, ніж при використанні таких методів

прогнозування, як тренди та ARIMA. Коефіцієнт кореляції можна визначити за допомогою різних статистичних методів, включаючи Пірсона, Спірмана і Кендала. Кожен з методів має особливості та застосовується залежно від типу даних та передбачуваного зв'язку між змінними.

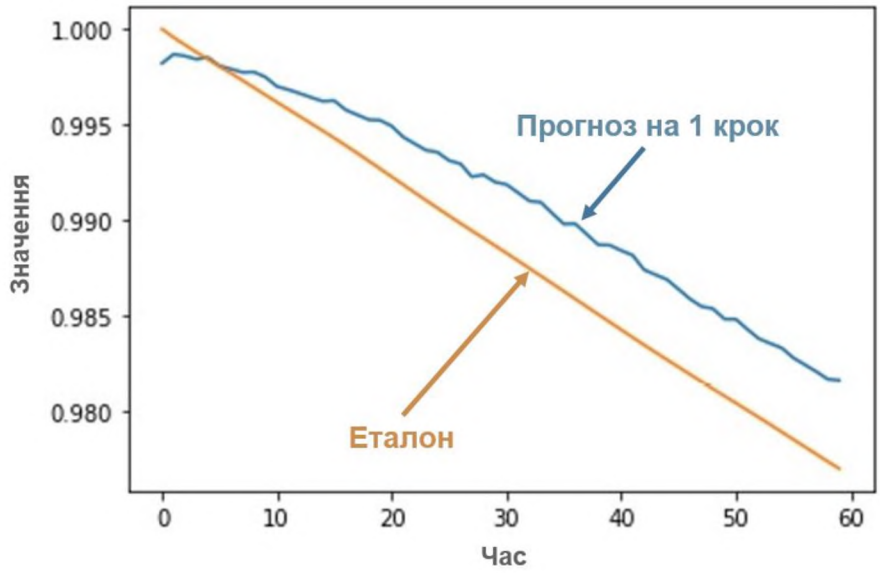


Рисунок 1.5 – Кореляція прогнозу

У часових рядах тренд є загальним напрямом руху даних у часі. Це може бути висхідний, низхідний або стаціонарний тренд. Тренд може суттєво спотворити кореляційні показники вихідних даних. Його вплив тим гірше, чим більше величина елементів тренду проти величиною елементів вихідного ряду. Визначення та усунення тренду допомагає виділити більш тонкі структури даних, такі як сезонність або циклічні коливання.

Для цього використовують метод чисельного диференціювання часових рядів, який полягає у обчисленні різниць між послідовними точками даних. Тобто, диференціювання даних – це віднімання поточного значення з наступного. Це просте диференціювання першого порядку. Після диференціювання тренд даних стає менш вираженим або повністю зникає. Це полегшує аналіз стаціонарних характеристик часового ряду, таких як сезонність чи шум. Після усунення тренду дані стають більш підходящими для навчання нейронних мереж, оскільки більшість алгоритмів краще

працюють із стаціонарними даними. Це дозволяє моделі краще вловлювати внутрішні закономірності та робити більш точні прогнози. При чисельному диференціюванні важливо вибрати правильні гіперпараметри, такі як порядок диференціювання та розмір вікна (якщо використовується ковзне середнє). Неправильний вибір може призвести до втрати важливої інформації або надмірного шуму даних.

Таким чином, при підготовці датасету візьмемо 100 елементів у xTrain, 101 елемент у yTrain і т. д. Щоб не робити цього вручну, можна скористатись інструментом TimeseriesGenerator. Це утиліта з бібліотеки Keras, призначена для генерації пакетів часових рядів для навчання або тестування моделей машинного навчання. Основні параметри для TimeseriesGenerator виступають:

– xTrain –часовий ряд, з якого збираються дані;

– yTrain – цільові значення. Повинні бути тієї ж довжини, що і xTrain;

– xLen – довжина вихідних послідовностей, вікно, яким ви пройдете за даними;

– sampling_rate – розмір кроку під час вибірки даних у xTrain.

Висновки до розділу 1

Прогнозування трафіку сайту може визначати ефективність онлайн-бізнесу щодо залучення аудиторії, формування регламентів використання серверів хостингу, хмарних ресурсів та ін. Трафік сильно варіюється в залежності від того, який зараз час дня, який це день тижня та ін. З математичної точки зору, трафік сайту можна представити еквівалентним часовим рядом.

На даний час, є кілька варіантів прогнозування часових рядів, що спираються на методи екстраполяції, SARIMA та методи машинного навчання. До останніх відноситься використання моделей глибокого

навчання нейронних мереж. В ході досліджень встановлено, що найбільш придатними для Прогнозування трафіку сайту є кілька архітектур: MLP, CNN, RNN (LSTM, GRU), складні змішані мережі, трансформери.

Для прогнозування трафіку сайту доцільно використовувати TL, що має кілька переваг: ефективність навчання; покращена продуктивність; подолання обмежень даних; гнучкість і адаптивність; прискорення розробки; передача знань між доменами; поліпшення вирішення специфічних завдань.

При створенні датасету у завданні прогнозування часового ряду виконується низка процедур: структуризація даних; за необхідності – збагачення та очищення даних; нормалізація даних; усунення впливу тренда шляхом чисельного диференціювання часових рядів; формування навчальної перевірочної та тестової вибірок.

РОЗДІЛ 2
МОДЕЛЬ ГЛИБОКОГО НАВЧАННЯ НЕЙРОННОЇ МЕРЕЖІ ПРОГНОЗУВАННЯ ТРАФІКУ САЙТУ

2.1 Базові архітектури нейронних мереж

RNN (рис. 2.1) – вид нейронних мереж, де зв'язки між елементами утворюють спрямовану послідовність. RNN підходять для прогнозування трафіку сайту завдяки їхній здатності обробляти часові послідовності та вловлювати залежності в даних. Однак, слід враховувати їх складність у налаштуванні та високі вимоги до ресурсів, а також потенційні проблеми із довгостроковим запам'ятовуванням та чутливістю до шуму у даних.

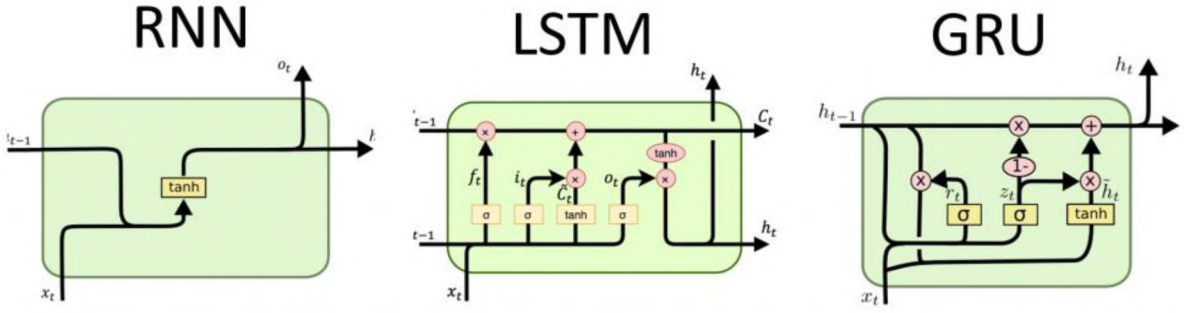


Рисунок 2.1 – Архітектури рекурентних мереж

RNN мають низку переваг та недоліків при використанні для прогнозування трафіку сайту. Перевагами слід вважати наступне. RNN ідеально підходять для роботи з тимчасовими рядами, такими як дані про трафіку сайту, оскільки вони здатні обробляти послідовність даних різної довжини. Такі мережі здатні вловлювати довгострокові залежності у даних, що важливо для розуміння тенденцій та патернів у трафіку сайту. RNN можуть бути налаштовані для врахування різних факторів, що впливають на трафік, таких як часові інтервали, сезонність, спеціальні заходи та ін. Недоліки RNN є такі положення. У класичних RNN є проблема зникаючого градієнта, яка ускладнює навчання на довгих послідовностях даних. Тобто, це

проблема втрати інформації. Чим довша послідовність, тим більша ймовірність того, що інформація про перші значення буде губитися. Навчання RNN може вимагати значних обчислювальних ресурсів, особливо великих наборів даних. Вибір відповідної архітектури RNN (наприклад, LSTM або GRU) та налаштування параметрів може бути складним завданням. RNN можуть бути чутливі до шуму та аномалій даних, що може призвести до некоректних прогнозів. 2-направлені (Bidirectional) RNN можуть використовуватися для прогнозування часових рядів, особливо в сценаріях, де важливо враховувати контекст як з минулих, так і з майбутніх точок даних (рис. 2.2).

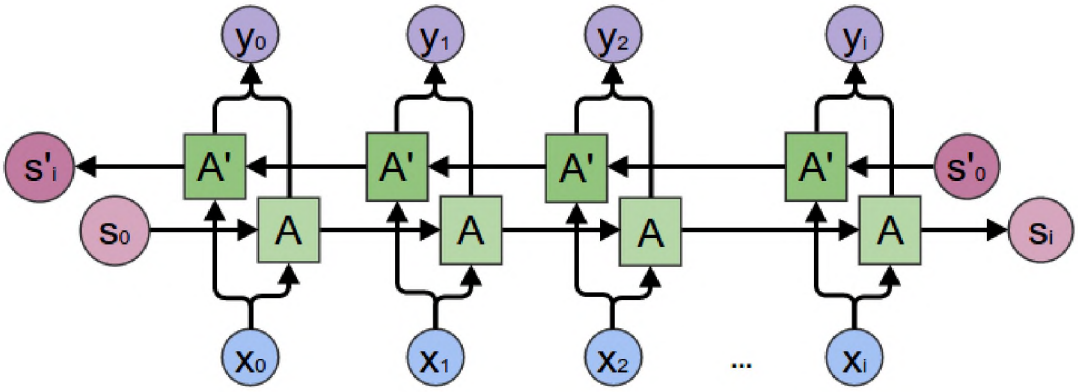


Рисунок 2.2 – Bidirectional RNN

На відміну від традиційних RNN, які обробляють дані послідовно від початку до кінця, Bidirectional RNN обробляє дані у двох напрямках: від початку до кінця і від кінця до початку (див. рис. 2.2). Це дозволяє мережі вловлювати залежності та контекст з обох напрямків часового ряду. За рахунок використання інформації з минулого та майбутнього, Bidirectional RNN може покращити якість прогнозів, особливо у завданнях, де для розуміння поточного стану важливий контекст з обох сторін. Ці моделі знаходять застосування у різних галузях, де тимчасові ряди відіграють ключову роль, наприклад, у фінансовому прогнозуванні, прогнозуванні погоди, аналізі електрокардіограм (ЕКГ) та інших медичних додатках. Bidirectional RNN, зазвичай, вимагають більше обчислювальних ресурсів і

часу для навчання в порівнянні з односпрямованими RNN через 2-направлену обробку даних. Одним з обмежень Bidirectional RNN є те, що для роботи їм потрібні дані з майбутнього, що робить їх менш придатними для прогнозування в реальному часі. У разі частіше використовуються односпрямовані RNN чи інші підходи. В цілому, Bidirectional RNN є потужним інструментом для аналізу та прогнозування часових рядів, особливо коли необхідно враховувати контекст як з минулих, так і з майбутніх даних. Однак їх ефективне використання вимагає ретельного планування та врахування специфіки даних і завдань, що застосовуються. Мережі LSTM (Long Short-Term Memory), буквально «довга короткострокова пам'ять» – це RNN, що здатна навчатися довгостроковим залежностям. Вона спеціально розроблена для усунення проблеми градієнта, що зникає. Спеціалізація LSTM – запам'ятовування інформації протягом тривалих періодів часу. Мережі GRU можна розглядати як більш просту версію мереж LSTM. Вона включає багато схожих понять, але має менше параметрів, і тому при тому самому розмірі прихованого шару GRU навчається швидше. Використання згорткової нейронної мережі (CNN) для прогнозування часових рядів – це ефективний метод, який застосовується у машинному навчанні. Conv1D добре підходить для часових рядів, тому що він ефективно обробляє одновимірні послідовні дані і може виявляти важливі часові ознаки для прогнозування (рис. 2.3).

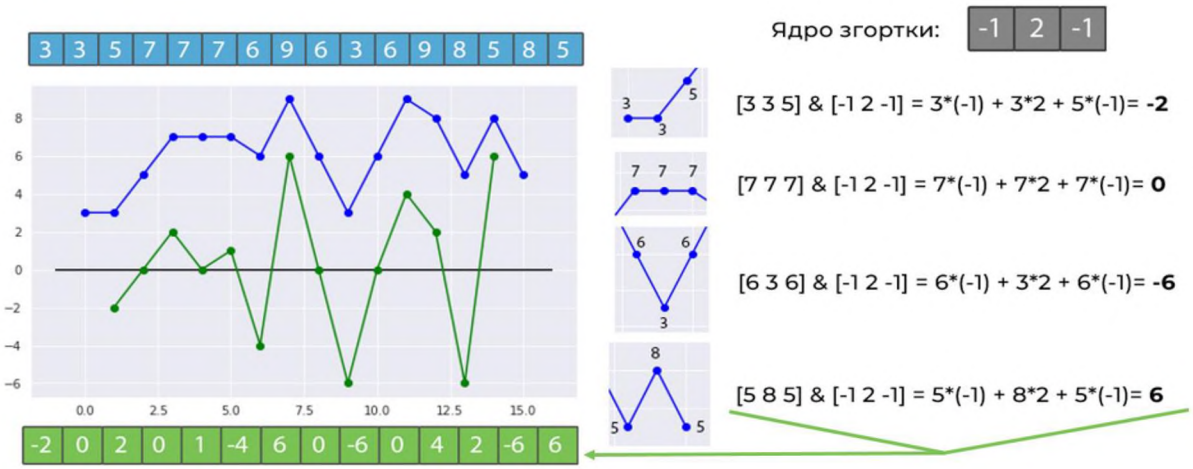


Рисунок 2.3 – Принцип реалізації Conv1D

Створюючи шар, ми вказуємо такі параметри: вікно згортки (фільтри) та ядро згортки. Щоб порахувати всі значення згортки, потрібно кожен елемент вікна помножити кожен елемент ядра і скласти їх. Після цього зміщуємось на один елемент вправо і повторюємо процедуру поки не дійдемо до останнього елемента послідовності.

Для створення моделі CNN Conv1D необхідно реалізувати кілька шарів нейронної мережі.

1. Вхідний шар, який повинен відповідати формі даних. Наприклад, якщо використовуємо часові ряди довжиною 100 з одною ознакою, форма вводу буде (100, 1).

2. Згорткові шари (Conv1D) – дані проходять через фільтри, які автоматично витягують важливі ознаки. При цьому, можна налаштувати кількість фільтрів та розмір ядра.

3. Шари Pooling. Вони використовуються для зменшення розмірності даних та виділення найбільш значимих ознак.

4. Повнозв'язні шари (Dense). Після згортання та пулінгових шарів дані подаються на один або кілька повнозв'язних шарів для подальшого аналізу.

До переваг використання CNN Conv1D при обробці часових рядів слід віднести те, що час навчання значно нижчий, ніж у RNN. До недоліків відносяться відсутність можливості «запам'ятати» потрібні дані на тривалий термін. Однак, це можна усунути за допомогою механізму уваги.

2.2 Складні змішані мережі

Використання техніки Embedding при прогнозуванні часових рядів є одним із перспективних напрямків. Вона дозволяє перетворити складні, високорозмірні дані на більш компактне та кероване представлення. Це особливо корисно в контексті часових рядів, де дані часто характеризуються високою розмірністю та складною структурою. Існує кілька ключових

моментів щодо застосування Embedding у прогнозуванні часових рядів. Вона допомагає скоротити кількість ознак часового ряду, зберігаючи у своїй важливу інформацію.

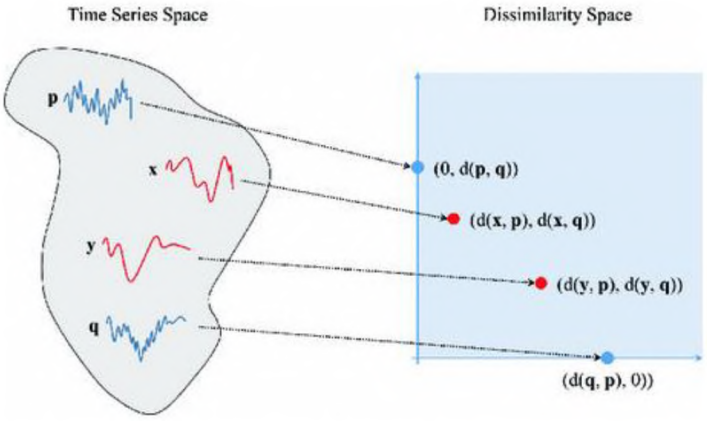


Рисунок 2.4 – Сутність роботи техніки Embedding [20]

Це спрощує моделювання та може покращити продуктивність передбачуваних моделей. За її допомогою можна виявити приховані структури та закономірності даних, які можуть бути неочевидні при прямому аналізі. Ця техніка дозволяє інтегрувати різні типи даних (наприклад, часові ряди з додатковими категоріальними або текстовими даними) у єдине уявлення, що покращує якість аналізу та прогнозування. У контексті глибокого навчання Embedding часто використовується для підготовки вхідних даних для нейронних мереж, що дозволяє більш ефективно моделювати складні залежності в часових рядах. Незважаючи на те, що Embedding може зробити дані абстрактнішими, він також може допомогти в інтерпретації моделей, виявляючи ключові фактори, що впливають на прогноз. На першому етапі вихідні дані (які можуть бути часовими рядами та ін.) перетворюються на числову форму, якщо вони ще не представлені в такому вигляді. В процесі Embedding кожен елемент відображається у просторі з меншою розмірністю. Ці вектори називаються «утворення». Вони, зазвичай, навчаються таким чином, щоб схожі елементи були ближче один до одного у просторі Embedding. Отримані вектори використовуються як вхідні дані для інших алгоритмів ML або глибокого навчання. Це дозволяє моделям

ефективно працювати з вихідними даними, оскільки embedding зменшує розмірність даних і виділяє важливі ознаки. Вектори, одержані за допомогою Embedding, можуть бути інтерпретовані та аналізовані для виявлення прихованих шаблонів та залежностей у даних.

Крім техніки Embedding в складних змішаних мережах можуть використовуватись кілька послідовно включених нейронних мереж з різними архітектурами, а також мереж на основі TL. В якості прикладу такого підходу можна розглядати мережу, що спочатку містить LSTM, а потім MobileNetV3 (що відноситься до CNN). Для початку на прикладі детектора руху розглянемо архітектуру тимчасових згорткових мереж (Temporal Convolutional Network) та їх переваги перед традиційними підходами, такими як CNN і RNN. У [21] було вперше запропоновано використовувати часові згорткові мережі (TCN) для сегментації дій. Процес розбивається на два етапи: по-перше, обчислення низькорівневих ознак з використанням (найчастіше) CNN, яка кодує просторово-часову інформацію, і по-друге, введення низькорівневих ознак до класифікатора, який отримує високорівневу часову інформацію за допомогою RNN. TCN пропонує уніфікований підхід, щоб покрити обидва рівні інформації за принципом ієрархії. На рис. 2.5 представлена структура енкодера-декодера [21, 22].

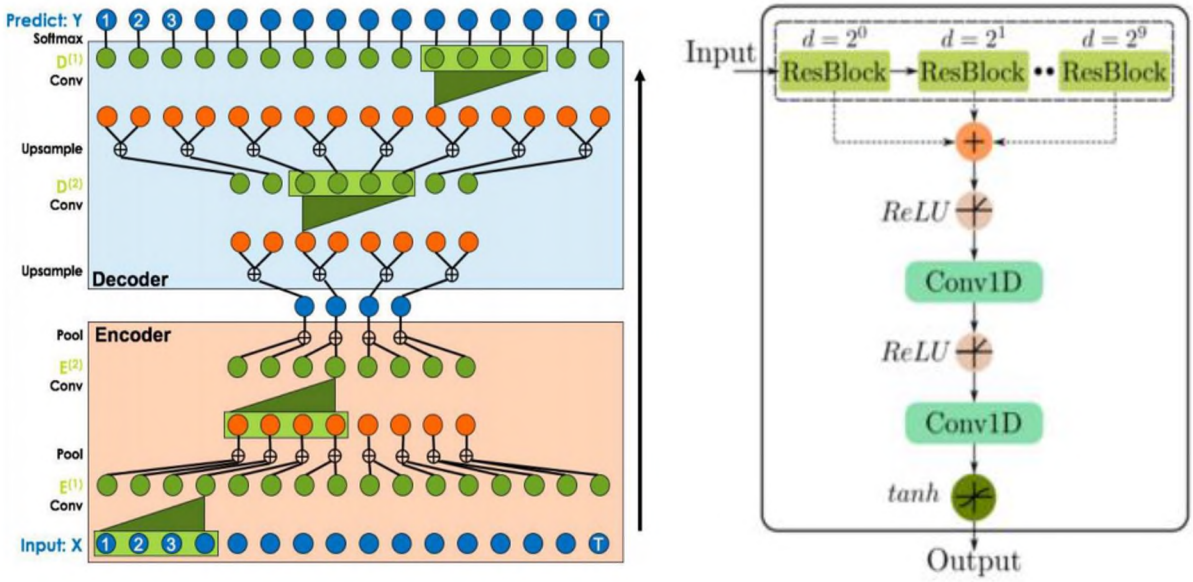


Рисунок 2.5 – Варіант TCN

Згідно [23], автори провели експеримент порівняно TCN та LSTM. Одним із результатів став висновок про те, що TCN добре справляється із завданнями прогнозування часових рядів (рис. 2.6).

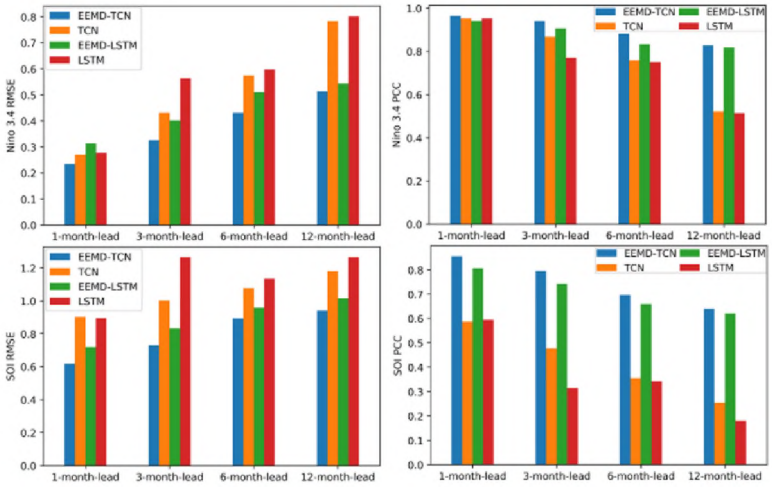


Рисунок 2.6 – Використання TCN для прогнозування часових рядів

У [24] запропонована архітектура глибокої TCN для прогнозування часових рядів (рис. 2.7).

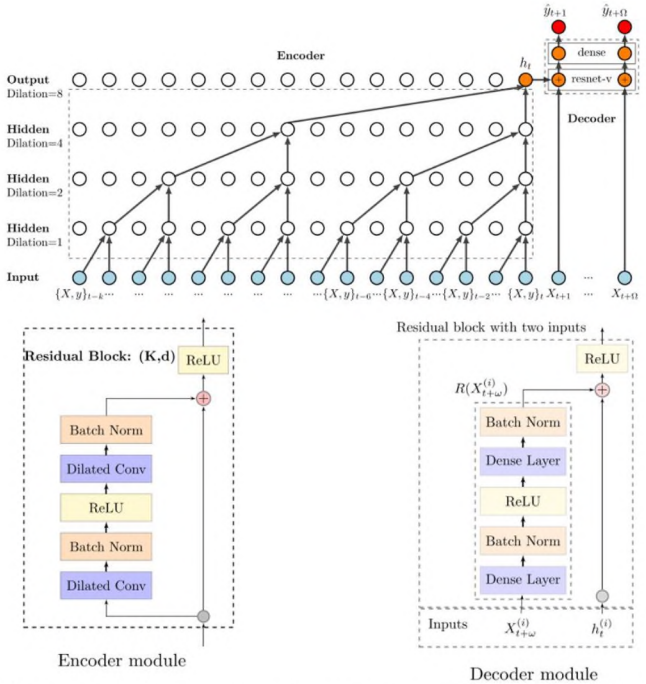


Рисунок 2.7 – Архітектура глибокої TCN

Реалізація модулів енкодера-декодера може допомогти в розробці прикладних великомасштабних програм.

2.3 Архітектура TimeGPT

Згідно [7], моделі на основі трансформерів [25] набувають популярності в останні роки, оскільки вони демонструють чудову продуктивність у великомасштабних налаштуваннях [26] і складних завданнях, таких як прогнозування тривалої послідовності. Попередні приклади включають TFT [27] і MQ-Transformer [28], обидва з багатоквантильними можливостями. The Informer представив Transformers для прогнозування довгої послідовності через механізм Prob-sparse self-attention [29]. З тих пір ця концепція була вдосконалена за допомогою різних форм індуктивного упередження та механізмів уваги в таких моделях, як Autoformer [30], FEDformer [31] і PatchTST [32].

Основна ідея представленої базової моделі полягає в тому, щоб використовувати ці принципи, навчаючи її на найбільшому загальнодоступному наборі даних часових рядів на сьогоднішній день, використовуючи закони масштабування набору даних і розмірів моделі. Різноманітний набір даних, з точки зору ширини та глибини, дозволяє TimeGPT збирати інформацію з безпрецедентного масиву часових шаблонів у багатьох доменах.

TimeGPT – це модель часових рядів на основі Transformer із механізмами самоуважності на основі [33]. TimeGPT використовує вікно історичних значень для створення прогнозу, додаючи локальне позиційне кодування для збагачення вхідних даних. Архітектура складається зі структури кодера-декодера з декількома рівнями, кожен із яких має залишкові зв'язки та нормалізацію рівня. Нарешті, лінійний рівень відображає вихід декодера на розмір вікна прогнозування. Загальна інтуїція полягає в тому, що механізми на основі уваги здатні вловити різноманіття минулих подій і правильно екстраполювати потенційні майбутні розподіли.

Розробка узагальненої глобальної моделі для часових рядів тягне за собою численні проблеми, головним чином через складне завдання обробки

сигналів, отриманих від широкого набору базових процесів. Такі характеристики, як частота, розрідженість, тренд, сезонність, стаціонарність і гетероскедастичність, представляють певні ускладнення як для локальних, так і для глобальних моделей. Тому будь-яка фундаментальна модель прогнозування повинна володіти здатністю керувати такою неоднорідністю. Наша модель, TimeGPT, розроблена для обробки часових рядів із різними частотами та характеристиками, одночасно враховуючи різні розміри вхідних даних і горизонти прогнозування. Ця адаптивність значною мірою пояснюється основною трансформаторною архітектурою, на якій побудовано TimeGPT. Слід зазначити, що TimeGPT не базується на існуючій великій мовній моделі (LLM). Хоча TimeGPT дотримується того ж принципу навчання великої моделі трансформатора на величезному наборі даних, його архітектура спеціалізується на обробці даних часових рядів і навчена мінімізувати помилку прогнозування (рис. 2.8).

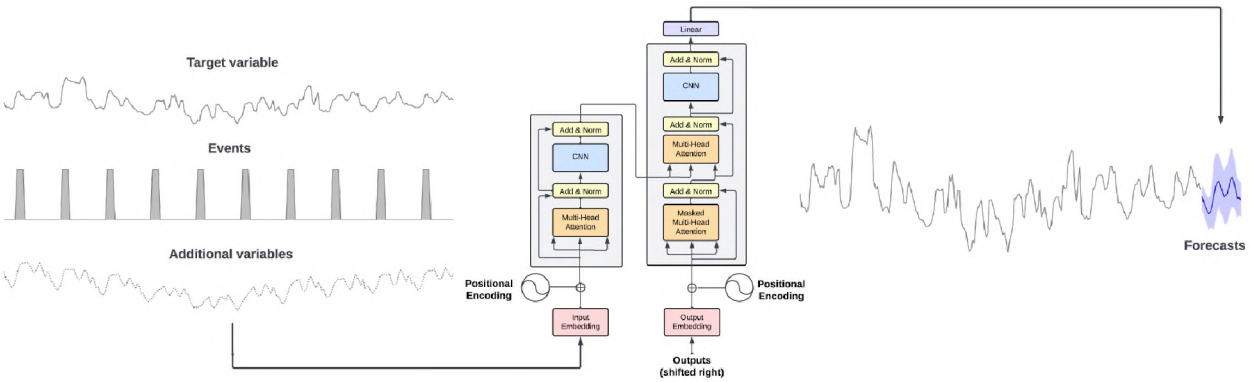


Рисунок 2.8 – Виведення нових часових рядів

TimeGPT використовує історичні значення цільових значень і додаткові екзогенні змінні як вхідні дані для створення прогнозів. Ми покладаємося на конформні прогнози, засновані на історичних помилках, щоб оцінити інтервали прогнозів. Як відомо, TimeGPT навчався на найбільшій колекції загальнодоступних часових рядів, що разом охоплюють понад 100 мільярдів точок даних. Цей навчальний набір включає часові ряди з широкого спектру областей, включаючи фінанси, економіку, демографію, охорону здоров'я, погоду, дані датчиків IoT, енергетику, вебтрафік, продажі,

транспорт і банківську справу. Завдяки цьому різноманітному набору доменів навчальний набір даних містить часові ряди з широким діапазоном характеристик. Що стосується часових закономірностей, навчальний набір даних містить ряди з кількома сезонами, циклами різної тривалості та різними типами трендів. Окрім часових шаблонів, набір даних також змінюється з точки зору шуму та викидів, пропонуючи надійне навчальне середовище. Деякі серії містять чисті регулярні шаблони, тоді як інші містять значний шум або несподівані події, що забезпечує широкий спектр сценаріїв, на яких модель може вчитися. Більшість часових рядів було включено в необробленому вигляді; обробка була обмежена стандартизацією формату та заповненням відсутніх значень для забезпечення повноти даних.

Вибір такого різноманітного навчального набору має вирішальне значення для розробки надійної базової моделі. Ця різноманітність охоплює складні реалії нестаціонарних даних реального світу, де тенденції та моделі можуть змінюватися з часом через безліч факторів. Навчання TimeGPT на цьому багатому наборі даних дає змогу працювати з широким спектром сценаріїв, підвищуючи його надійність і можливості узагальнення. Це фактично дозволяє TimeGPT точно прогнозувати невидимі часові ряди, усуваючи потребу в індивідуальному навчанні та оптимізації моделі.

TimeGPT пройшов багатоденний період навчання на кластері графічних процесорів NVIDIA A10G [34]. Щоб оптимізувати швидкість навчання, розміри пакетів та інші параметри необхідне підбирати гіперпараметри. При цьому спостерігається закономірність, яка узгоджується з висновками [35], де більший розмір партії та менший рівень навчання виявилися корисними. Реалізований у PyTorch, TimeGPT було навчено за допомогою Адама зі стратегією зниження швидкості навчання, яка знизила швидкість до 12 % від початкового значення.

Імовірнісне прогнозування стосується оцінки невизначеності моделі щодо прогнозів. Правильна оцінка калібрування моделі дає змогу оцінювати ризики та приймати обґрунтовані рішення. Конформне прогнозування,

непараметрична структура, пропонує переконливий підхід до створення інтервалів прогнозування із заздалегідь заданим рівнем точності покриття [36, 37]. На відміну від традиційних методів, конформне прогнозування не потребує строгих припущень розподілу, що робить його більш гнучким і агностичним щодо моделі або домену часових рядів. Під час визначення нового часового ряду ми виконуємо ковзні прогнози на основі останніх доступних даних, щоб оцінити помилки моделі в прогнозуванні конкретного цільового часового ряду (рис. 2.9). Кожен компонент на графіку представляє розподіл rMAE для групи. При цьому центральна лінія показує середнє значення. TimeGPT лідирує за продуктивністю, за нею йдуть методи глибокого навчання, статистичні дані, машинне навчання та базові моделі. Результати аналогічні для інших частот.

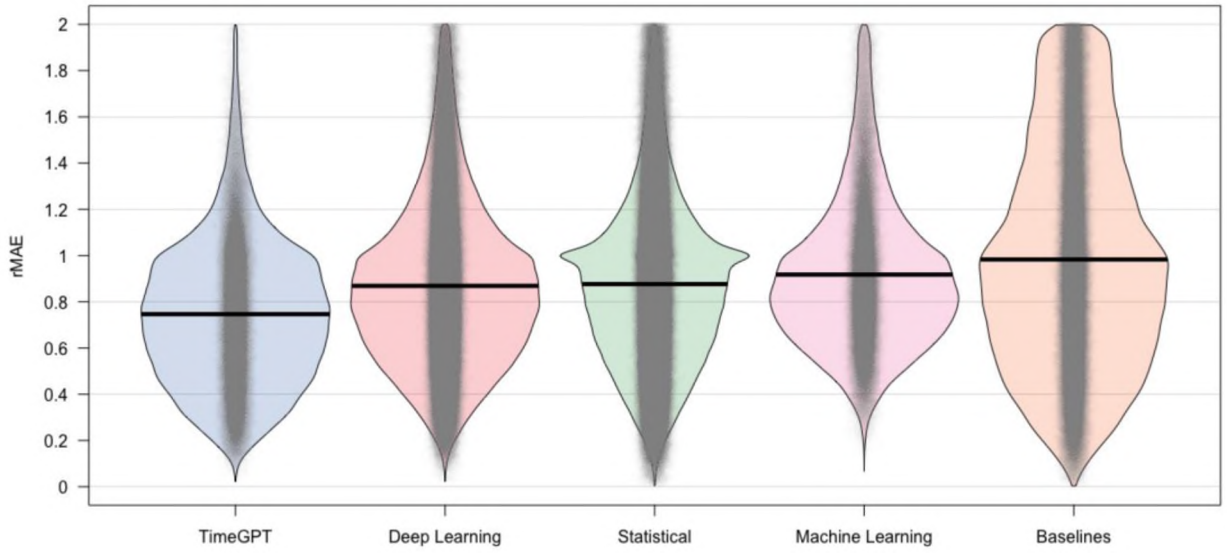


Рисунок 2.9 – Відносна середня абсолютна похибка (rMAE) для TimeGPT і різних груп моделей для місячної частоти

Класично, оцінка ефективності прогнозування базується на поділі кожного часового ряду набору даних на набори тренувань і тестів на основі визначеного відсічення. Такий принцип, навіть у версії перехресної перевірки, не є достатньо суворим для оцінки базової моделі, оскільки його основною властивістю є здатність точно передбачати повністю нові серії. Для TimeGPT

як базової моделі прогнозування, тестовий набір містить понад 300000 часових рядів із різних областей, включаючи фінанси, вебтрафік, Інтернет речей, погоду, попит та електроенергію. Оцінка виконується в останньому вікні прогнозування кожного часового ряду, довжина якого змінюється залежно від частоти вибірки. TimeGPT використовує попередні історичні значення як вхідні дані (див. рис. 2.9) без повторного навчання своїх ваг (нульовий постріл). При цьому визначається інший горизонт прогнозування на основі частоти для представлення загальних практичних застосувань: 12 для місячних, 1 для тижневих, 7 для щоденних і 24 для погодинних даних. TimeGPT було порівняно з широким спектром базових, статистичних, машинного навчання та нейронних моделей прогнозування, щоб забезпечити комплексний аналіз продуктивності. Базові лінії та статистичні моделі індивідуально навчаються для кожного часового ряду тестового набору з використанням історичних значень, що передували останньому вікну прогнозування. Було вибрано підхід глобальної моделі для методів машинного та глибокого навчання для кожної частоти, використовуючи всі часові ряди в тестовому наборі. Деякі популярні моделі, наприклад, Prophet [38] і ARIMA, були виключені з аналізу через їх непомірні обчислювальні вимоги та тривалий час навчання. Вибрані нами показники оцінки включають відносну середню абсолютну похибку (rMAE) і відносну середньо-квадратичну похибку (rRMSE), обидві нормалізовані щодо ефективності сезонної наївної моделі. Цей вибір виправданий додатковими відомостями, які пропонують ці відносні похибки, оскільки вони показують приріст продуктивності по відношенню до відомого базового рівня, покращуючи інтерпретацію наших результатів.

Показники відносної похибки забезпечують додаткову перевагу незалежності від масштабу, дозволяючи порівнювати результати для кожної частоти. Щоб забезпечити надійну чисельну стабільність і послідовність оцінювання, ми застосовуємо цю нормалізацію в глобальному масштабі для кожного комплексного набору даних. Конкретні обчислення для цих

показників, що застосовні до набору даних із n часовими рядами та прогнозним горизонтом h відповідають виразу:

$$rMAE = \frac{\sum_{i=1}^{n}\sum_{t=1}^{h}\left|y_{i,t} - \hat{y}_{i,t}\right|}{\sum_{i=1}^{n}\sum_{t=1}^{h}\left|y_{i,t} - \hat{y}_{i,t}^{base}\right|} \text{ та } rRMSE = \frac{\sum_{i=1}^{n}\sqrt{\sum_{t=1}^{h}\left(y_{i,t} - \hat{y}_{i,t}\right)^2}}{\sum_{i=1}^{n}\sqrt{\sum_{t=1}^{h}\left(y_{i,t} - \hat{y}_{i,t}^{base}\right)^2}}. \quad (2.1)$$

Спершу перевіряємо можливості TimeGPT при Zero-shot, тобто жодного додаткового тонкого налаштування тестового набору не виконуємо (рис. 2.10). При цьому, TimeGPT перевищує повну колекцію перевірених статистичних моделей і підходів глибокого навчання SoTA, увійшовши до трійки найкращих за частотами. Найкраща модель для кожної частоти та показника виділена жирним шрифтом, друга найкраща підкреслена, а третя найкраща підкреслена пунктиром.

	Monthly		Weekly		Daily		Hourly	
	rMAE	rRMSE	rMAE	rRMSE	rMAE	rRMSE	rMAE	rRMSE
ZeroModel	2.045	1.568	6.075	6.075	2.989	2.395	10.255	8.183
HistoricAverage	1.349	1.106	4.188	4.188	2.509	2.057	2.216	1.964
SeasonalNaive	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
Theta	0.839	0.764	1.061	1.061	0.841	0.811	1.163	1.175
DOTheta	0.799	0.734	1.056	1.056	0.837	0.806	1.157	1.169
ETS	0.942	0.960	1.079	1.079	0.944	0.970	0.998	1.009
CES	1.024	0.946	1.002	1.002	0.919	0.899	0.878	0.896
ADIDA	0.852	0.769	1.364	1.364	0.908	0.868	2.307	2.207
IMAPA	0.852	0.769	1.364	1.364	0.908	0.868	2.307	2.207
CrostonClassic	0.989	0.857	1.805	1.805	0.995	0.933	2.157	2.043
LGBM	1.050	0.913	0.993	0.993	2.506	2.054	**0.733**	**0.709**
LSTM	0.836	0.778	1.002	1.002	0.852	0.832	0.974	0.955
DeepAR	0.988	0.878	0.987	0.987	0.853	0.826	1.028	1.028
TFT	0.752	0.700	0.954	0.954	0.817	0.791	1.120	1.112
NHITS	0.738	0.694	0.883	0.883	**0.788**	**0.771**	0.829	0.860
TimeGPT	**0.727**	**0.685**	**0.878**	**0.878**	0.804	0.780	0.852	0.878

Найкраща модель для кожної частоти

Рисунок 2.10 – Продуктивність (rMAE і rRMSE) моделей з Zero-shot

Слід зазначити, що достовірність моделі прогнозування можна оцінити лише відносно її ефективності порівняно з конкуруючими альтернативами. Хоча точність зазвичай розглядається як єдиний відповідний показник, вартість обчислень і складність реалізації є ключовими факторами для

практичного застосування. У зв'язку з цим слід зазначити, що результати TimeGPT, які повідомляються, є результатом простого та надзвичайно швидкого виклику методу прогнозування попередньо навченої моделі. Для порівняння, інші моделі потребують повного конвеєра для навчання та прогнозування.

Тонка настройка є критично важливим кроком для ефективного використання базових моделей і трансформаторних архітектур. Основні моделі попередньо навчені на величезній кількості даних, охоплюючи широкі та загальні характеристики. Однак ці моделі часто потребують спеціалізації для певних контекстів або доменів. Шляхом точного налаштування ми коригуємо параметри моделі в наборі даних для конкретного завдання, дозволяючи моделі адаптувати свої величезні попередні знання відповідно до вимог нового завдання. Цей процес гарантує, що модель зберігає своє широке розуміння та відмінно справляється з конкретними завданнями. Завдяки властивій їм гнучкості та здатності вивчати складні шаблони, трансформаторні архітектури особливо виграють від тонкого налаштування, підвищуючи свою продуктивність у предметно-спеціальних програмах. Таким чином, точне налаштування служить найважливішим мостом, який пов'язує широкі можливості базових моделей із специфікою цільових завдань. На рис. 2.11 представлено результати щодо підвищення точності TimeGPT порівняно з кількістю кроків тонкого налаштування для підмножини часових рядів у тестовому наборі.

Для Zero-shot внутрішні тести зафіксували середню швидкість виводу графічного процесора 0,6 мс на серію для TimeGPT, що майже відображає швидкість простого Seasonal Naive. Для порівняння ми розглядаємо оптимізовані для паралельних обчислень статистичні методи, які, доповнені компіляцією Numba, усереднюють швидкість 600 мс на серію для навчання та виводу. З іншого боку, глобальні моделі, такі як LGBM, LSTM і NHITS, продемонстрували більш тривале середнє значення 57 мс на серію, враховуючи як навчання, так і вивід. Завдяки можливостям Zero-shot,

TimeGPT перевищує традиційні статистичні методи та глобальні моделі з загальною швидкістю на порядки.

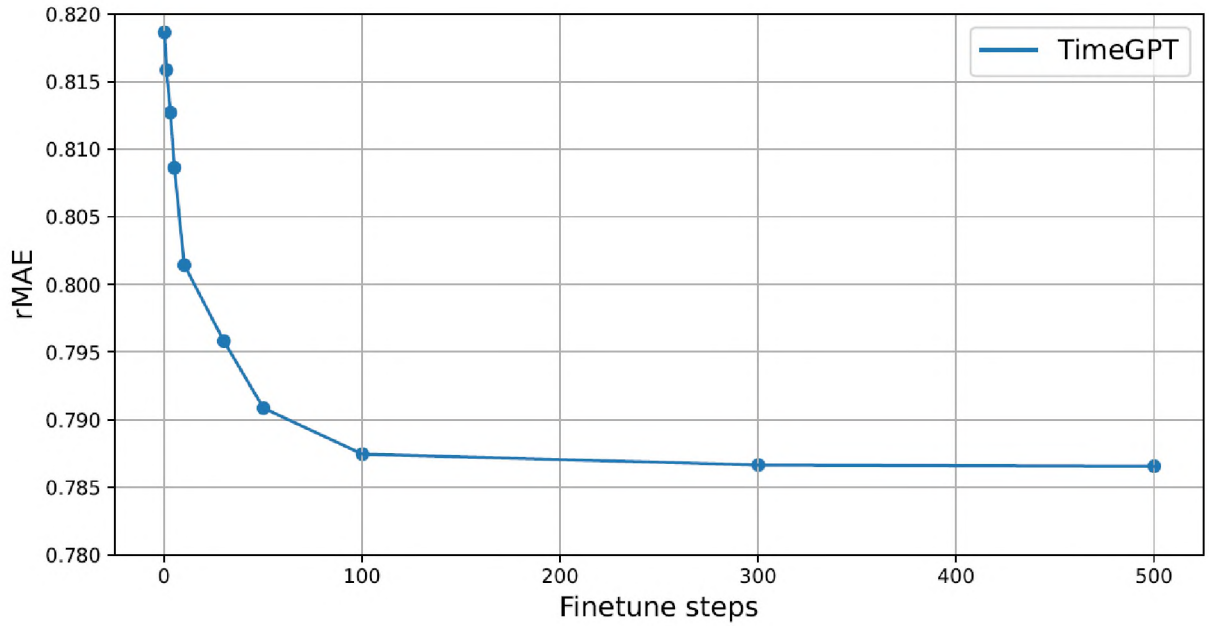


Рисунок 2.11 – Продуктивність TimeGPT з тонким налаштуванням на підмножині часових рядів із тестового набору, виміряного rMAE

Сучасна практика прогнозування, зазвичай, включає складний конвеєр, що містить кілька кроків від обробки даних до навчання моделі та вибору. TimeGPT значно спрощує цей процес, скорочуючи конвеєри до етапу виводу, суттєво зменшуючи складність і витрати часу, водночас досягаючи найсучаснішої продуктивності. Імовірно, важливим є те, що TimeGPT демократизує переваги великих моделей трансформерів, які сьогодні обмежені організаціями з величезними обсягами даних, обчислювальними ресурсами та технічними знаннями. Тобто, розглянуті моделі суттєво вплинуть на сферу прогнозування та можуть змінити поточні практики. Запровадження базової моделі в часових рядах, яка нагадує інші поля та відкриває можливий шлях для майбутніх удосконалень, можна вважати важливою віхою в галузі часових рядів. Хоча TimeGPT показує досить гарні результати, все ще є багато важливих обмежень і відкритих питань. Розглянуті результати узгоджуються з прогнозами щодо очікуваної продуктивності моделей великих часових рядів [39-41]. Ці результати

підтверджують закони масштабування стосовно співвідношення розміру моделі, розміру набору даних і продуктивності архітектури трансформерів. Вони пояснюють, чому простіші моделі можуть перевершувати трансформери на менших наборах даних, як спостерігалося в таких дослідженнях, як [42].

Таким чином, релевантність трансформерів залежить від контексту, і вони часто стають більш корисними зі збільшенням розміру набору даних. Ці закономірності пропонують важливу практичну інформацію для управління вибором моделі для конкретних завдань. У ситуаціях, коли існують обмеження щодо доступності великих наборів даних або обчислювальних ресурсів, простіші моделі можуть бути більш придатними.

2.4 Програмна реалізація моделі глибокого навчання

Згідно завдання, використовується набір даних про трафік сайту. Він містить 1094 рядки, кожен з яких містить дату та обсяг трафіку. Для перевірки можливості практичної реалізації запропонованого підходу в якості робочого середовища вибраний хмарний сервіс Google Colab, що дозволяє запускати програмний код на мові Python. В якості прикладу розглянемо модель глибокого навчання прогнозування часових рядів на основі базової архітектури CNN Conv1D. Спочатку завантажуємо файл з набором даних «traff.csv» на власний Google-диск. Для роботи необхідно підключити кілька бібліотек та/або модулів з них, наприклад: numpy, pandas, tensorflow.keras.models, tensorflow.keras.optimizers, sklearn.preprocessing, matplotlib.pyplot, tensorflow.keras.preprocessing.sequence. Надалі підключаємо власний Google-диск, щоб був доступ до даних. Потім через бібліотеку pandas зчитуємо ці дані та видаляємо непотрібну колонку з датою. Таким чином, маємо набір даних, що наведено на рис. 2.12. Надалі, формуємо необхідні вибірки для навчання нейронні мережі, масштабуємо дані,

використовуємо генератори для навчання та валідації, проводимо візуалізацію наявних результатів, формуємо модель та навчаємо її.

	traff
0	29593.0
1	31726.0
2	32684.0
3	31820.0
4	29354.0
...	...
1089	29801.0
1090	34913.0
1091	37665.0
1092	32643.0
1093	28212.0

1094 rows × 1 columns

Рисунок 2.12 – Модифікований набір даних

На при кінці будуємо графіки середньої абсолютної помилки та середньої помилки для навчальної та перевірочної вибірок, наприклад, рис. 2.13. При прогнозуванні трафіку сайту, зазвичай, обмежуються прогнозом на один або 10 кроків вперед.

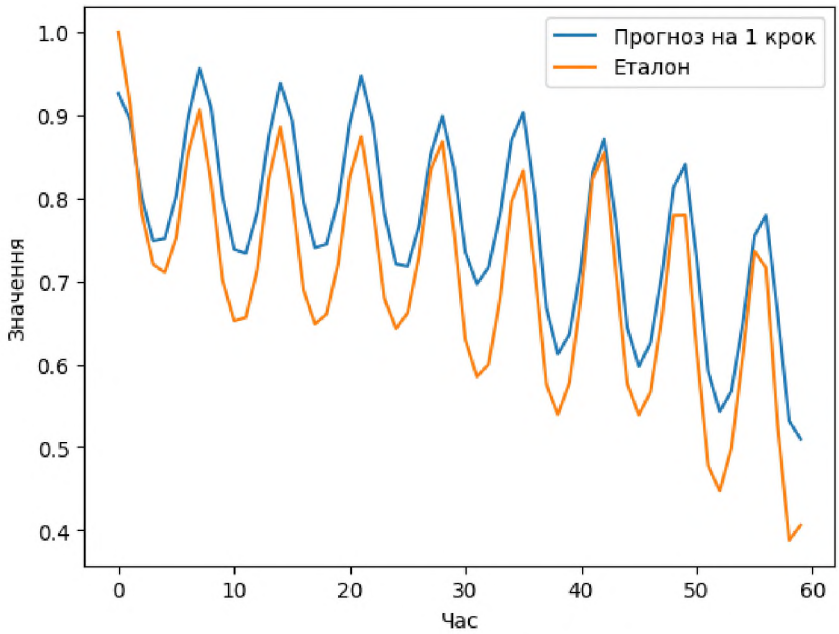


Рисунок 2.13 – Результат роботи CNN Conv1D

Це обумовлене кількома ключовими факторами. Зі збільшенням горизонту прогнозування точність, зазвичай, знижується. Це пов'язано з тим, що довгострокові прогнози схильні до більшої кількості невизначеностей і можуть бути менш надійними. Часові ряди можуть бути піддані змінам через сезонність, тренди, економічні та інші зовнішні фактори. При прогнозуванні на більшу кількість кроків ці фактори можуть значно змінитись, що ускладнює точне передбачення. У багатьох випадках, особливо у бізнесі та фінансах, короткострокові прогнози є більш цінними, оскільки вони дозволяють швидше реагувати на зміни та приймати ефективні рішення. Деякі методи прогнозування найкраще підходять для короткострокових прогнозів. Наприклад, методи ML можуть бути не такі ефективні для довгострокових прогнозів через обмеження в доступних даних і можливість моделювання складних залежностей на довгих часових інтервалах. Для довгострокового прогнозування потрібні дані високої якості на тривалий період часу. У багатьох випадках такі дані або недоступні, або їх якість погіршується з часом.

Загалом, обмеження прогнозування часових рядів одним чи десятьма кроками вперед обумовлено прагненням до балансу між точністю прогнозу та його практичною застосовністю.

Висновки до розділу 2

Однією з придатних до прогнозування трафіку сайту є архітектура RNN має гнучкість у моделюванні та забезпечує обробку послідовних даних, врахування часових залежностей. Однак, до її недоліків слід віднести проблему зникаючого градієнта, високі вимоги до обчислювальних ресурсів, складність у підборі архітектури та чутливість до шуму даних.

Використання техніки Embedding у прогнозуванні часових рядів дозволяє зменшити розмірності; виявити приховані шаблони; інтеграцію

різнорідних даних; використання глибокого навчання; підвищити інтерпретованість. Релевантність архітектури трансформерів залежить від контексту, і вони часто стають більш корисними зі збільшенням розміру набору даних. Ці закономірності пропонують важливу практичну інформацію для управління вибором моделі для конкретних завдань. У ситуаціях, коли існують обмеження щодо доступності великих наборів даних або обчислювальних ресурсів, простіші моделі можуть бути більш придатними.

При прогнозуванні трафіку сайту, зазвичай, обмежуються прогнозом на один або 10 кроків вперед. Це обумовлене кількома ключовими факторами: точність прогнозу; мінливість даних; практична значущість; методи прогнозування; доступність та якість даних. Загалом, обмеження прогнозування часових рядів одним чи десятьма кроками вперед обумовлено прагненням до балансу між точністю прогнозу та його практичною застосовністю.

РОЗДІЛ 3
РЕКОМЕНДАЦІЇ ЩОДО ВИКОРИСТАННЯ МОДЕЛІ ПРОГНОЗУВАННЯ ТРАФІКУ САЙТУ

3.1 Оцінка продуктивності моделі

Одним з напрямів дослідження моделі глибокого навчання нейронної мережі прогнозування трафіку сайту є аналіз впливу архітектури нейронної мережі та налаштування гіперпараметрів, наприклад, рис. 3.1 і 3.2 [4].

```
Model: "model_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 1 (InputLayer)              [(None, 30, 1)]           0

 Conv1D_3 (Conv1D)           (None, 30, 32)            192

 BatchNormalization_4 (Batch  (None, 30, 32)           128
 Normalization)

 MaxPool1D_5 (MaxPooling1D)  (None, 15, 32)            0

 Conv1D_6 (Conv1D)           (None, 15, 64)            10304

 BatchNormalization_7 (Batch  (None, 15, 64)           256
 Normalization)

 MaxPool1D_8 (MaxPooling1D)  (None, 8, 64)             0

 Conv1D_9 (Conv1D)           (None, 8, 128)            41088

 BatchNormalization_10 (Batc  (None, 8, 128)           512
 hNormalization)

 MaxPool1D_11 (MaxPooling1D)  (None, 4, 128)           0

 Conv1D_12 (Conv1D)          (None, 4, 128)            82048

 BatchNormalization_13 (Batc  (None, 4, 128)           512
 hNormalization)

 MaxPool1D_14 (MaxPooling1D)  (None, 2, 128)           0

 Conv1D_15 (Conv1D)          (None, 2, 128)            82048

 2 (Dense)                   (None, 2, 1)              129

=================================================================
Total params: 217,217
Trainable params: 216,513
Non-trainable params: 704
```

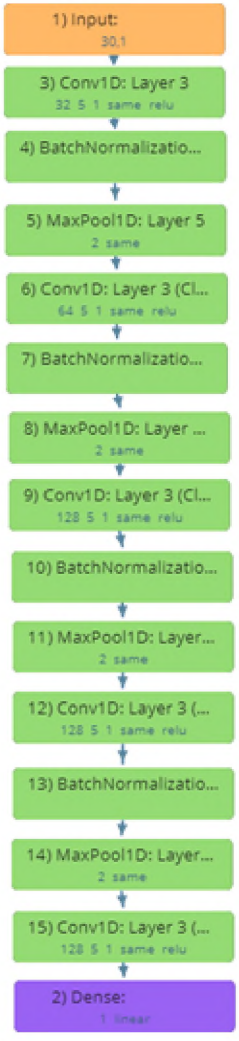


Рисунок 3.1 – Приклад візуалізації архітектури CNN

Оцінка продуктивності моделі виконувалась за параметром PercentMAE (рис. 3.3).

```
Model: "model_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
1 (InputLayer)               [(None, 30, 1)]           0

Conv1D_3 (Conv1D)            (None, 30, 32)            192

BatchNormalization_4 (Batch  (None, 30, 32)            128
Normalization)

MaxPool1D_5 (MaxPooling1D)   (None, 15, 32)            0

Conv1D_6 (Conv1D)            (None, 15, 64)            10304

BatchNormalization_7 (Batch  (None, 15, 64)            256
Normalization)

MaxPool1D_8 (MaxPooling1D)   (None, 8, 64)             0

Conv1D_9 (Conv1D)            (None, 8, 128)            41088

BatchNormalization_10 (Batc  (None, 8, 128)            512
hNormalization)

MaxPool1D_11 (MaxPooling1D)  (None, 4, 128)            0

Conv1D_12 (Conv1D)           (None, 4, 128)            82048

BatchNormalization_13 (Batc  (None, 4, 128)            512
hNormalization)

MaxPool1D_14 (MaxPooling1D)  (None, 2, 128)            0

Conv1D_15 (Conv1D)           (None, 2, 128)            82048

2 (Dense)                    (None, 2, 1)              129

=================================================================
Total params: 217,217
Trainable params: 216,513
Non-trainable params: 704
```

Рисунок 3.2 – Варіант CNN Conv1D

Рисунок 3.3 – Результати (PercentMAE) навчання моделі, що наведена на рис. 3.1

Згідно п. 2.1 і 2.2, досліджувались варіанти архітектури CNN, LSTM, що можуть містити техніку Embedding, TL на основі MobilNetv3Smal, EfficientNet (наприклад, рис. 3.4), а також архітектури з механізмом уваги (рис. 3.5) [18]. В розробленій моделі був вибраний оптимізатор – Adam. Крок навчання на перших епохах дорівнював 0,001, а потім 0,0001.

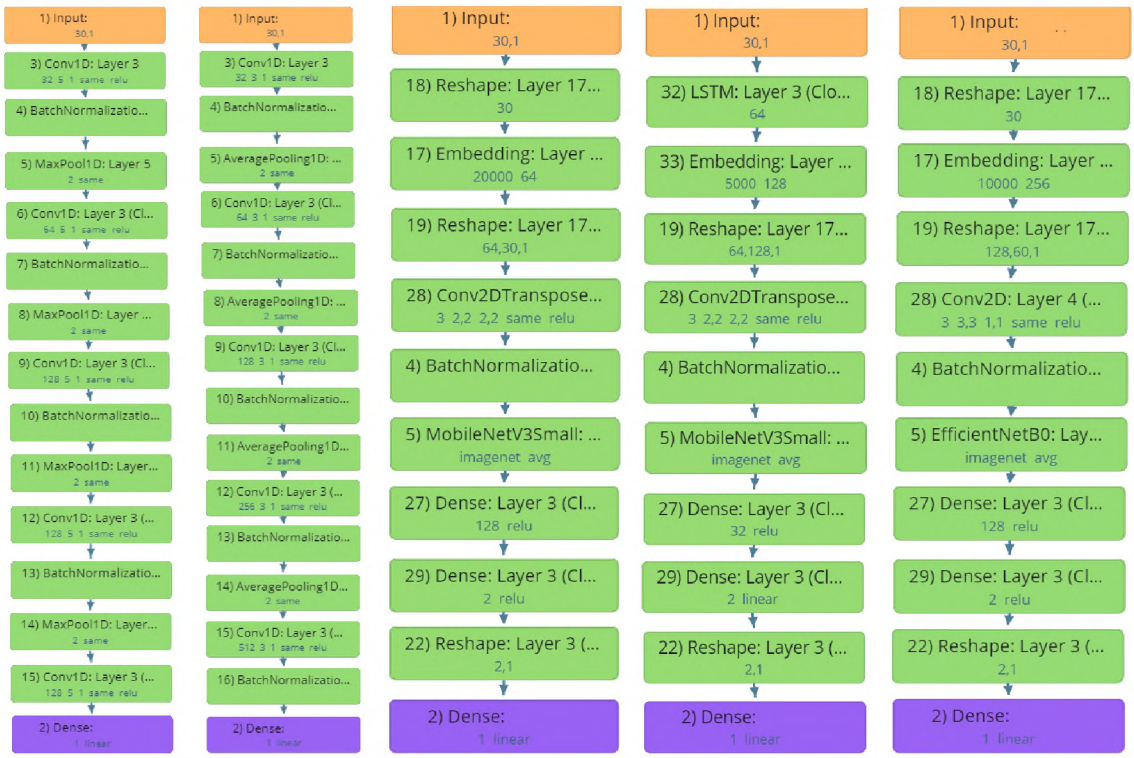


Рисунок 3.4 – Приклад досліджуваних архітектур

Розмір Batch мав два значення – 4, 8, 16 та ін. Навчання виконувалось на відеокарті NVIDIA A100-SXM4-40GB в середовищі Google Colab з використанням бібліотеки Keras. Аналізувались залежності автокореляції, та виконувалась візуалізація прогнозу сайту трафіку.

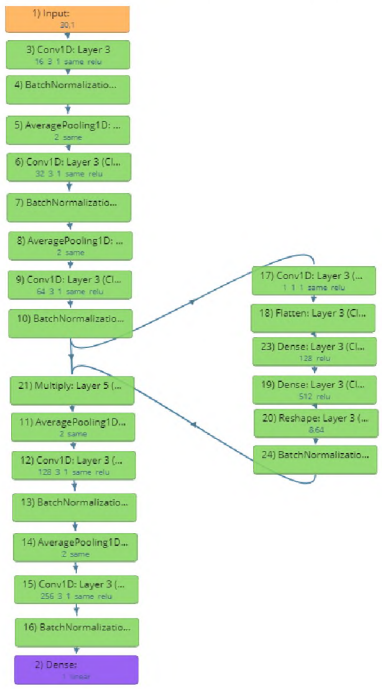


Рисунок 3.5 – Варіант архітектури нейронної мережі з механізмом уваги

Деталізацію виконання оцінки запропонованої моделі глибокого навчання прогнозу трафіку сайту можна розглянути за варіантом архітектури CNN Conv1D з технікою Embedding. Спочатку аналізувався прогноз на 1 крок. Наприклад, на рис. 3.6 наведений фрагмент графіку автокореляції, значення помилки під час навчання – рис. 3.7.

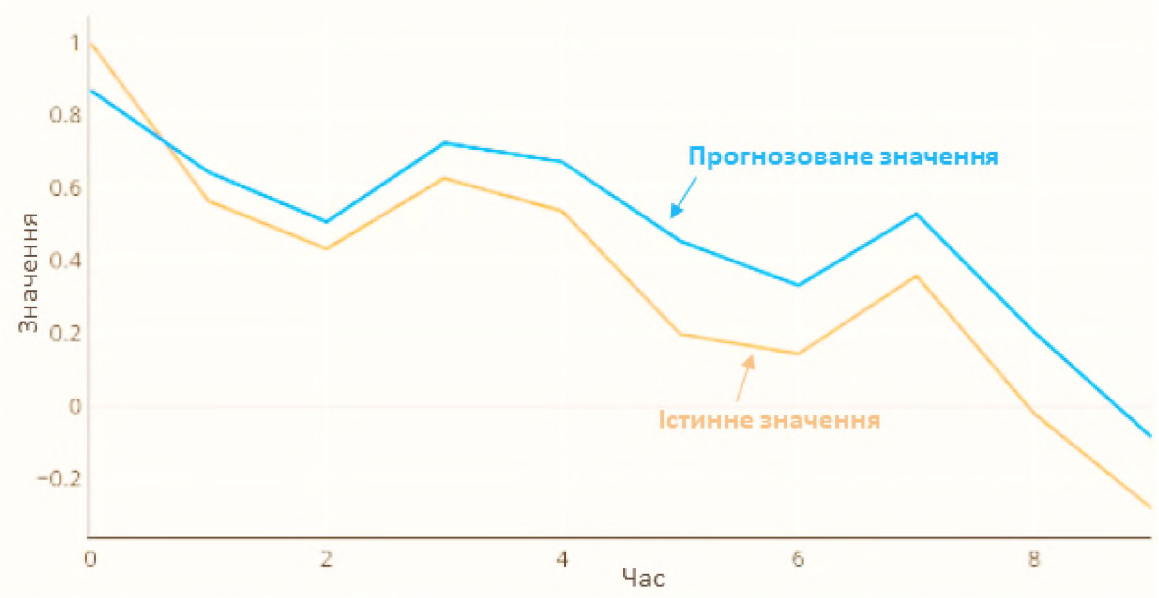


Рисунок 3.6 – Автокореляція для завдання прогнозу на 1 крок

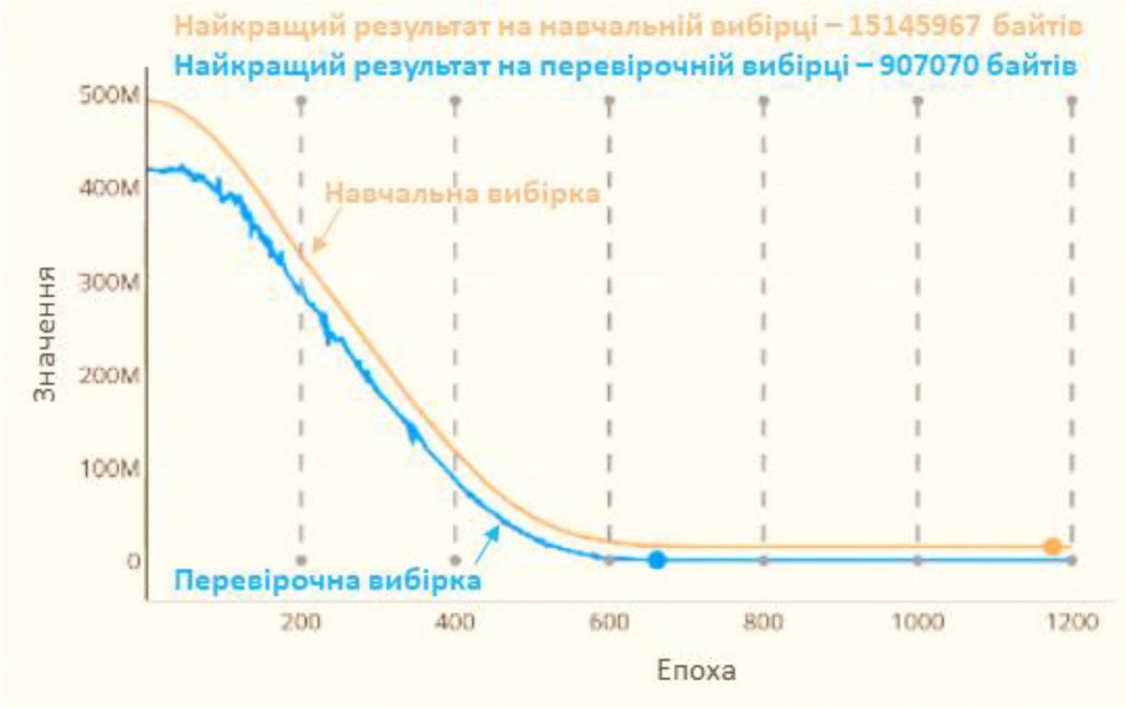


Рисунок 3.7 – Помилка для завдання прогнозу на 1 крок

Відповідно, на рис. 3.8 наведені вхідні дані та результат прогнозу на 1 крок на перевірочній виборці.

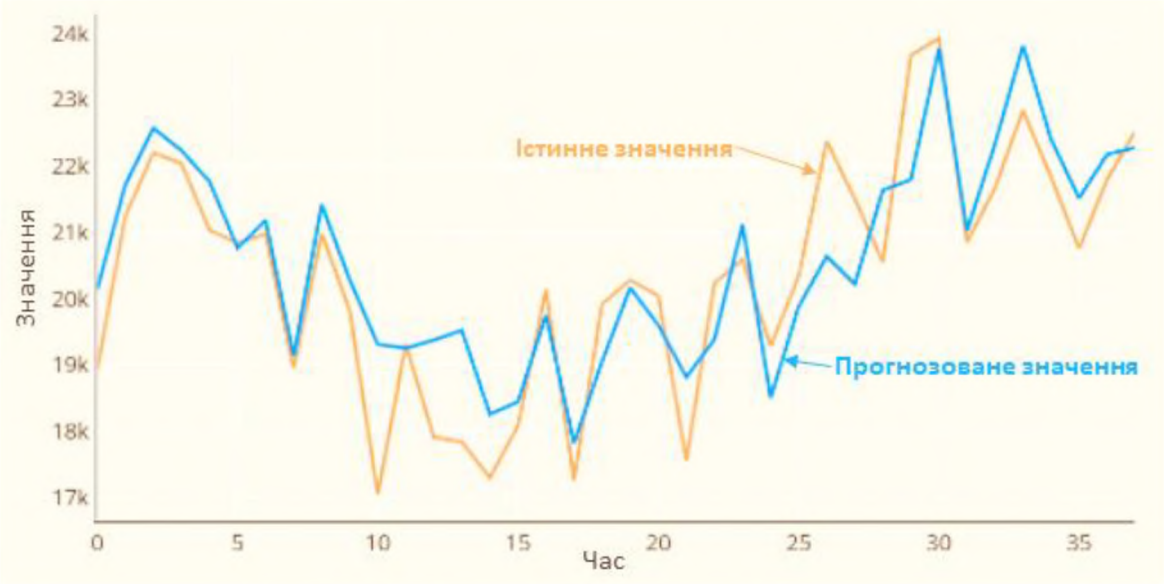


Рисунок 3.8 – Прогноз на 1 крок

Аналогічним чином було отримано відповідні залежності для завдання прогнозу на 2 кроки. Наприклад, на рис. 3.9 наведений фрагмент графіку автокореляції, значення PercentMAE під час навчання – рис. 3.10.

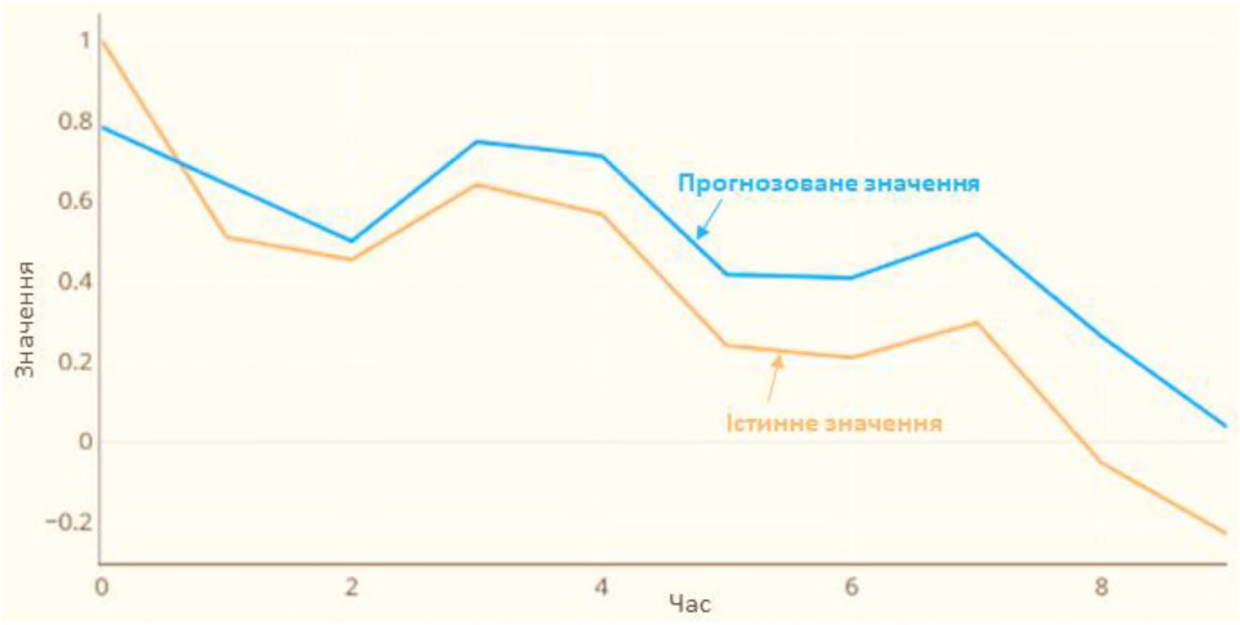


Рисунок 3.9 – Автокореляція для завдання прогнозу на 2 крок

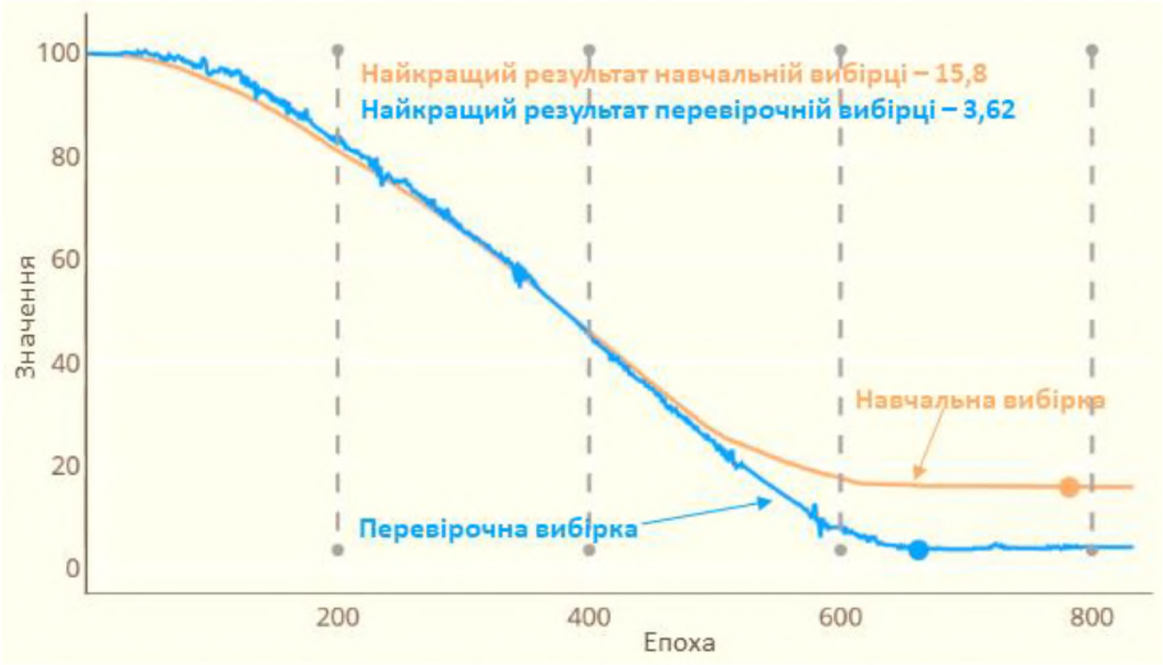


Рисунок 3.10 – Фрагмент графіку PercentMAE для прогнозу на 2 кроки

На рис. 3.11 наведено результат прогнозу на 2 кроки, а на рис. 3.12 – щільність трафіку сайту, що розрахована на відповідній кількості епох в процесі навчання нейронної мережі.

Рисунок 3.10 – Прогноз на 2 кроки

В цілому, на основі проведених досліджень можна зробити наступні висновки.

1. У разі прогнозу трафіку наростання розмірності Conv1D від входу до виходу нейронної мережі дає кращу точність, ніж при зниженні розмірності Conv1D від входу до виходу.

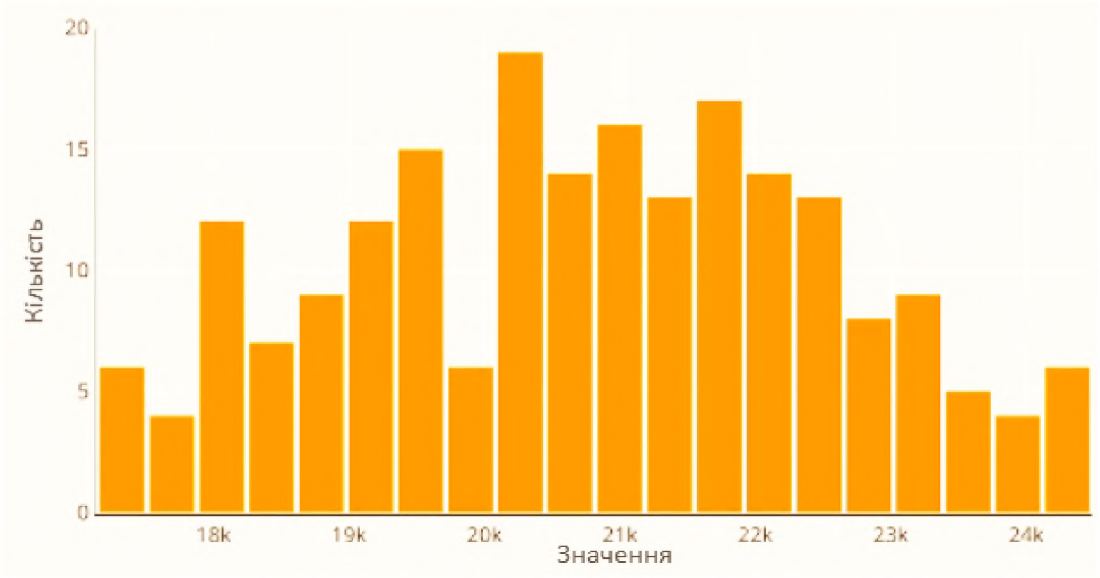


Рисунок 3.11 – Щільність трафіку сайту

2. Щодо механізму уваги. Короткозамкнутий варіант гірший, ніж кидок наперед через кілька шарів. Механізм уваги з відведенням у точці, що розташована ближче до входу, дозволив погасити викиди вище 100 відсотків у помилці. Найкращий результат виходить, коли обидва шари Dense (Dense+Relu та Dense+Softmax) мають однакову розмірність. Механізм уваги можна реалізувати на Dense+Relu+BatchNormalization замість комбінації Dense+SoftMax.

3.2 Варіант застосування нейронних мереж для розпізнавання маскованого трафіку

Класифікація мережного трафіку є важливим процесом, необхідним для правильної організації передачі даних між додатками, що його генерують. Визначення трафіку забезпечує основу для множини мережних функцій,

таких як управління, забезпечення безпеки, поділ послуг та ін. На даний час, традиційні методи класифікації включають кілька наступних підходів [43].

SNI. Цей метод, незважаючи на швидкість, характеризується недостатньою точністю, особливо у випадках, коли відбувається трансляція мережних адрес.

Інспекція корисного навантаження забезпечує високу точність, однак потребує постійного оновлення та може викликати побоювання щодо конфіденційності даних.

Статистичне машинне навчання використовує статистичні дані про трафік і також забезпечує високу точність, однак може потребувати багато ресурсів та бути повільним. Крім цього, він передбачає роботу з дуже чистими та розміченими даними.

Як наслідок, доцільно використовувати моделі глибокого навчання нейронних мереж класифікації прикладних протоколів у мережних пакетах. Наш новий підхід відрізняється від цих традиційних методів. Для цього можна використовувати архітектури CNN і ResNet з метою ідентифікації VPN- та proxy-трафіку. CNN відома здатністю обробляти складні патерни даних, тоді як ResNet пропонує переваги під час навчання глибоких нейронних мереж. Порівняння допомагає визначити придатність кожної архітектури аналізу мережевого трафіку. Порівнюючи продуктивність цих двох моделей, можна отримати рекомендації щодо їхньої придатності для аналізу мережного трафіку. В якості прикладу доцільно розглядати протокол IPFIX. Його назва означає «Експорт інформації про потоки інтернет-протоколу» (Internet Protocol Flow Information Export). Основні специфікації для IPFIX можна знайти в RFC 7011 RFC 7015 і RFC 5103. Це протокол, розроблений робочою групою інтернет-інженерів (IETF) для експорту інформації про потоки з роутерів, комутаторів та інших пристроїв до станції керування мережею. Потоки визначаються як набір пакетів IP, що проходять через пристрій, які мають загальні характеристики, такі як вихідні та цільові IP-адреси, порти вихідного та цільового призначення та тип протоколу. IPFIX

розроблений для того, щоб бути універсальним стандартом для моніторингу мережного трафіку та збору статистики про продуктивність мережі, допомагаючи адміністраторам мереж розуміти потоки даних та ефективніше керувати мережевими ресурсами. Це важливий інструмент для аналізу трафіку, моніторингу мережі, планування потужностей та аналізу безпеки.

Протокол є еволюцією версії 9 NetFlow від Cisco, прагнучи бути гнучкішим і забезпечувати універсальний стандарт експорту для інформації про потоки інтернет-трафіку. IPFIX надає дані, необхідні для різних типів аналізу мережі, включаючи профіль трафіку, планування мережі, білінг та облік використання.

Архітектура CNN (рис. 3.12) для класифікації IPFIX-трафіку складається з трьох компонентів: згортковий, повнозв'язний та вихідний шари. Запропонований метод використовують заснований на використанні класичної архітектури нейронної згортки.

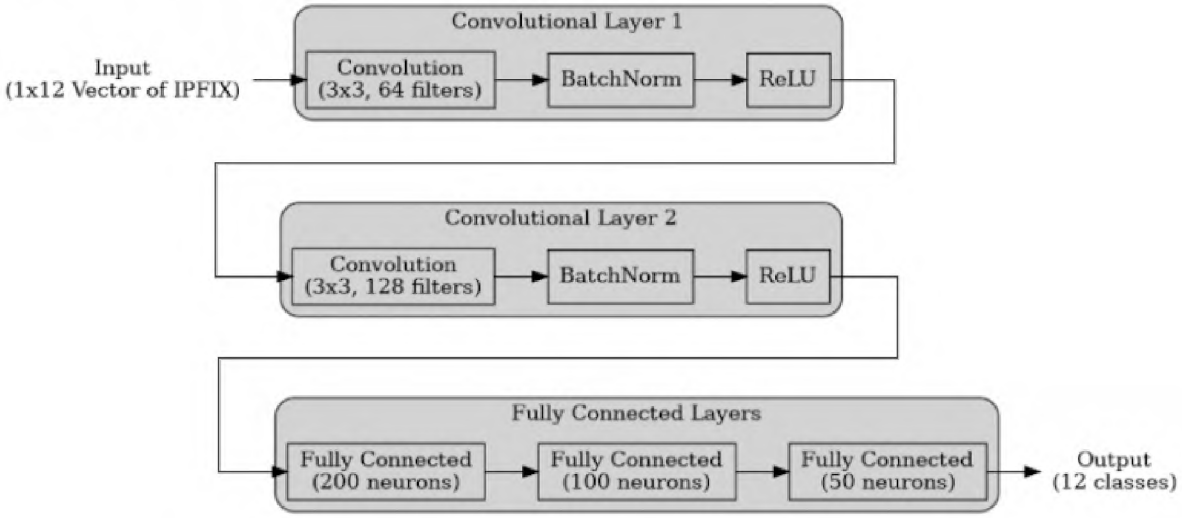


Рисунок 3.12 – CNN для класифікації IPFIX-трафіку

Нейронна мережа починається з двох послідовних згорткових шарів. Перший згортковий шар приймає на вхід дані з одним каналом, що є трафіком. Він застосовує операцію згортки із зазначеним числом вихідних каналів (c1_output_dim). Розмір ядра (c1_kernel_size) та крок (c1_stride) також є параметрами, які можна налаштовувати. Після операції згортки

застосовується функція активації ReLU для запровадження нелінійності. Так само другий згортковий шар працює з виходом першого шару. Він приймає на вхід каналів c1_output_dim і створює вихід з каналами c2_output_dim, використовуючи розмір ядра (c2_kernel_size) і крок (c2_stride), задані в конфігурації моделі. Операція підвиборки. Після кожного згорткового шару застосовується max-pooling. Шар виконує зменшення розмірів даних шляхом поділу вхідних карт ознак на регіони, що не перетинаються, розміром 2. Усередині кожного регіону вибирається максимальне значення як представницьке значення. Ця операція зменшує просторові розміри карт ознак, зберігаючи у своїй найбільш значиму інформацію. Повнозв'язані шари. Після шару max-pooling тензор перетворюється на одновимірну форму. Потім він проходить через серію трьох пов'язаних шарів (fc1, fc2, fc3). Розміри цих шарів заздалегідь визначені як 200, 100 та 50 відповідно. Кожному пов'язаному шару застосовується регулювання dropout з ймовірністю відсіву 0,05 для запобігання перенавчанню. Після кожного пов'язаного шару також використовуються функції активації ReLU. Вихідний прошарок. Фінальний пов'язаний шар, названий 'out', служить як вихідний шар. Він приймає вихід попереднього шару, що має 50 ознак, і відображає його в бажаний розмірний простір (output_dim), вказаний у гіперпараметрах моделі. У цілому нині, ця архітектура, орієнтована захоплення значних ознак і залежностей у даних мережному трафіку, що призводить до ефективним результатам класифікації. Конфігурація моделі та гіперпараметри можуть бути налаштовані для оптимізації продуктивності залежно від конкретного завдання класифікації.

Архітектура ResNet (Residual Network) хоча і використовується для класифікації зображень, але її адаптуємо під класифікацію мережного трафіку. Вона складається з кількох блоків з залишковими з’єднаннями, які полегшують навчання складним уявленням даних про трафік (рис. 3.13).

ResNet для класифікації мережного трафіку у форматі IPFIX включає два основних компоненти: блоки з залишковими з'єднаннями складаються з згортки, пакетної нормалізації та активації ReLU, що дозволяє прямому

потоку градієнта під час навчання і блоки ResNet являють собою ієрархію вилучення ознак зі збільшенням розміру фільтра (64, 128 та 256 фільтрів), захоплюючи складні патерни. Шари згортки: ResNet починається з серії згорткових шарів, які отримують ознаки з вхідних даних про трафік. Ці шари виконують згортки, які включають застосування фільтрів для захоплення локальних патернів і виявлення значущих ознак в даних.

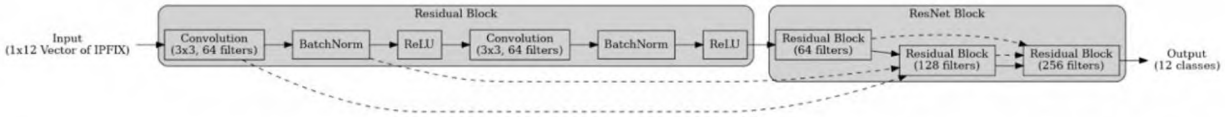


Рисунок. 3.13. Архітектура ResNet для класифікації мережного трафіку у форматі IPFIX

Кількість вихідних каналів (c_output_dim) і розмір ядра (c_kernel_size) є параметрами, що настроюються. Блоки із залишковими з'єднаннями: Ключовим компонентом ResNet є блок із залишковими з'єднаннями. Кожен блок із залишковими з'єднаннями складається з двох згорткових шарів. Кількість вихідних каналів для цих шарів (rb_output_dim) і розмір ядра (rb_kernel_size) також є параметрами, що настроюються. Залишкове з'єднання в блоці з залишковими з'єднаннями дозволяє мережі вивчати залишкові уявлення, які є різницею між входом та бажаним виходом. Пакетна нормалізація: Після кожного згорткового шару в блоках із залишковими сполуками застосовується пакетна нормалізація. Вона нормалізує активації для стабілізації та прискорення процесу навчання. Параметр моменту (bn_momentum) визначає внесок статистики попередньої партії у нормалізацію поточної партії. Функція активації: Функція активації (ReLU) застосовується після кожного згорткового шару та в блоках з залишковими з'єднаннями. Вона вводить нелінійність, що дозволяє мережі вивчати складні та нелінійні залежності даних про трафік. Пулінг: Іноді використовуються шари пулінгу, такі як максимальне об'єднання, для зменшення розмірів карток ознак, зменшуючи їх просторові розміри, зберігаючи при цьому

найбільш важливу інформацію. Розмір пулінга (p_pool_size) і крок (p_stride) є параметрами, що настроюються. Повнозв'язані шари: Після згорткових і пулінгових шарів карти ознак перетворюються на одновимірну форму і проходять через пов'язані шари. Розміри цих шарів (fc1_output_dim, fc2_output_dim, fc3_output_dim) наперед визначені. Вихідний шар: Останній шар архітектури ResNet це вихідний шар. Він приймає ознаки від попередніх шарів та відображає їх у певну кількість вихідних класів або міток (output_dim), що відповідають категоріям мережевого трафіку для класифікації. Загалом, архітектура ResNet з її згортковими шарами, блоками з залишковими з'єднаннями, пакетною нормалізацією та пов'язаними шарами націлена на захоплення та моделювання складних патернів та залежностей у даних про мережевий трафік. Використовуючи залишкові сполуки та метод навчання із залишками, ResNet успішно справляється із завданнями навчання глибоких нейронних мереж та досягає високої точності у задачах класифікації мережевого трафіку. Конкретні значення параметрів можна налаштовувати для оптимізації продуктивності залежно від вимог класифікації задачі.

Дослідження фокусується на порівнянні придатності окремих архітектур, а не дослідженні взаємодії різних моделей нейронних мереж. Воно оцінює ефективність кожної архітектури у аналізі мережного трафіку незалежно. Тому класифікація трафіку здійснюється за допомогою однієї нейронної мережі без взаємодії з іншими моделями. Слід зазначити, що класифікація виконується з накопиченням результату, при якому вибирається клас, що найчастіше зустрічається для пакетів з одним і тим же IP і портом. Класифікація мережного трафіку ґрунтується на ретельно підготовленому наборі даних, зібраному у форматі Netflow 10 (IPFIX). Процес збору даних включав запис дампів даних у форматі IPFIX на машині, обладнаній можливостями глибокої інспекції пакетів (DPI). Вона підключена до інших машин, що генерують трафік через різні VPN. Оскільки весь трафік з цих машин був охоплений VPN, згодом були зібрані IP та порт на цих машинах

спеціальним чином, щоб зібрати якнайбільше унікальних комбінацій IP та портів для кожної з VPN (табл. 3.1). Отже, було зібрано значну кількість даних з унікальними комбінаціями IP та портів для навчання нейронної мережі, оскільки VPN отримували нові IP та портові призначення при блокуванні.

Таблиця 3.1 – Структура даних IPFIX

Тип даних	Опис
octet_delta_count	Вхідний лічильник довжиною N×8 бітів кількості байт, пов'язаних з IP-потоком
packet_delta_count	Лічильник вхідних пакетів довжиною N×8 бітів для кількості пакетів, пов'язаних з IP-потоком
protocol_identifier	Байт IP-протоколу
ip_class_of_service	Клас IP або послуги
source_port	Порт відправника
source_ipv4	IPv4 відправника
destination_port	Порт одержувача
destination_ipv4	IPv4 одержувача
bgp_source_as_number	№ автоном. системи джерела BGP, де N може бути 2 або 4
bgp_destination_as_number	№ автоном. системи призначення BGP, де N може бути 2 або 4
input_snmp	Ідентифікатор віртуальної локальної мережі, пов'язаний із вхідним інтерфейсом
output_snmp	Ідентифікатор віртуальної локальної мережі, пов'язаний із інтерфейсом виходу
ip_version	Версія протоколу IPv4 або IPv6
post_nat_source_ipv4	IPv4 NAT джерела
post_nat_source_port	Порт NAT джерела
frgmt_delta_packs	Дельта фрагментованих пакетів
repeat_delta_pack	Дельта ретрансляцій
packet_deliver_time	Затримка (RTT/2), мс
protocol_code	Код протоколу з використанням класу автономної системи для нейронної мережі

Для завдань класифікації протоколів додатків та класифікації трафіку набір даних спочатку був розділений на навчальний та тестовий набори у співвідношенні 80 та 20 % відповідно. Для вирішення проблеми дисбалансу класів застосовувалося коригування кожного класу всередині навчального набору. Дані розмічалися за даними класів, отриманих зі стендів, де трафік збирався і перемішувався зі звичайним трафіком, класифікованим класом з IPFIX. Зібраний масив даних був використаний для навчання моделей

нейронних мереж двох різних архітектур з підбором гіперпараметрів. Загальне співвідношення протоколів у навчальній вибірці виглядає таким чином: DNS – 18,67 %; HTTP – 1,38%; HTTPS – 16,27%; DoH – 2,66%; ICMP – 4,83%; Bittorrent – 24,73%; AdGuard VPN – 2,34%; VPN Unlimited – 12,18%; Psiphon 3 – 12,41%; Lantern – 4,53. Після навчання моделі оцінювалися на окремому тестовому наборі для визначення їхньої продуктивності.

Метрики розраховувались наступним чином:

$$Recall = \frac{TP}{TP + FN},\ Precision = \frac{TP}{TP + FP},\ F1Score = 2\frac{Recall \cdot Precision}{Recall + Precision}, \quad (3.1)$$

де TP – істинно позитивний (клас який визначила нейронна мережа є правильним класом);

FN – хибнонегативний (нейронна мережа не визначила клас);

FP – хибнопозитивний (нейронна мережа визначила клас як позитивний, проте це виявилося помилкою).

Для аналізу VPN-трафіку, що становить значну частину зібраного масиву даних, використовувалася машина, оснащена технологією DPI. Для класифікації трафіку на машині із системою DPI працювала нейронна мережа. Трафік, що генерується машинами, підключеними до DPI через різні VPN, ефективно маршрутизується через DPI і згодом перехоплювався для аналізу як дампів. Нейронна мережа обробляла отримані дампи даних, які зберігалися у спеціально відведеному каталозі, та проводила аналіз мережних пакетів у форматі. Результати аналізу заносилися до бази даних, у якій створювалася повний запис класифікованих IP-адрес, портів та відповідних їм класів прикладних протоколів, і навіть номерів машин-джерел. Тестування проводилось з використанням VPN, на яких були навчені нейронні мережі, але з іншим діапазоном IP-адрес для чистоти експерименту. Нейронна мережа класифікувала трафік та заносила відповідний клас до бази даних. Потім реальний протокол користувача порівнювався з класом, визначеним нейронною мережею, для визначення кількості правильно класифікованих і помилкових протоколів.

За цими результатами розраховувалася точність, наприклад, саме для VPN і моделей кожної архітектури, які показали найкращий результат (табл. 3.2). Архітектура ResNet, при правильному налаштуванні гіперпараметрів перевершує CNN в класифікації протоколів користувача. Моделі класичної CNN зазнавали труднощів із протоколами, які мали мало унікальних рядків даних для навчання. Перевагу моделі ResNet можна пояснити її здатністю вловлювати та моделювати складні закономірності та залежності у даних мережевого трафіку за допомогою залишкових зв'язків.

Таблиця 3.2 – Продуктивність моделі класифікаці маскованого трафіку

Протоколи	CNN				ResNet			
	TP	FP	FN	F1 Score	TP	FP	FN	F1 Score
AdGuard VPN	28	9	50	0,49	60	5	18	0,84
VPN Unlimited	3	3	22	0,21	5	9	20	0,26
Psiphon 3	8455	160	399	0,97	8847	1030	7	0,95
Lantern	288	105	122	0,72	410	75	10	0,91

Крім того, ретельно підібрані гіперпараметри сприяли підвищенню продуктивності порівняно з вихідною моделлю. Отримані результати наголошують на ефективності архітектури ResNet і правильно підібраних гіперпараметрів для досягнення високої точності в задачах класифікації мережевого трафіку. Для подальшого поліпшення класифікації протоколів користувача в майбутньому можливе включення нових протоколів, таких як VPN в нейронну мережу архітектури ResNet. Крім того, збагачення даних трафіку додатковими вибірками може підвищити помітність пакетів трафіку, що призведе до підвищення точності класифікації протоколів користувача.

3.3 Техніко-економічне обгрунтування прийнятих рішень

Прогнозуванні трафіку сайту підвищує якість аналізу онлайн-бізнесу стосовно залучення клієнтів з врахуванням асу дня, дня тижня та інших чинників. Крім того, він може бути ініціатором включення додаткових

політки безпеки у разі різкого зростання кількості відвідувачів з соціальних мереж (це може вказувати на використання ботів).

З іншого боку, на основі створених прогнозів можливе формування вимог щодо апаратно-технічної складової платформи хостингу, де планується розміщення сайту.

В цілому, для практичної реалізації розглянутої моделі глибокого навчання нейронної мережі прогнозу трафіку сайту трудомісткість розробки коду складає 350 год. Крім цього, необхідно до 15 годин на тестування програмного продукту до 8 год для процедури DevOps. При цьому, вважаємо, що набір даних для формування датасету зібраний. Таким чином, загальна трудомісткість складає:

$$T = 350 + 15 + 8 = 373. \tag{3.2}$$

Відповідно, перерахунок трудомісткості у кількість робочих днів можна виконати з виразом:

$$M = \frac{T}{8} = 47 \text{ днів або} \approx 1{,}2 \text{ місяця}. \tag{3.3}$$

Враховуючи обсяг зазначених робіт, доцільно найняти програміста-фрілансера на мові Python з відповідною класифікацією. Згідно [44], в середньому по Україні, його заробітна плата складає $C = 45000$ грн. Як наслідок, витрати на заробітну плату програміста-фрілансера дорівнюють:

$$V = M \cdot C = 1{,}2 \cdot 45000 = 54000 \text{ [грн]}. \tag{3.4}$$

Згідно, п. 3.1, для реалізації процесу навчання нейронної мережі використовується хмарний сервіс Google Colab [45]. Щоб оптимізувати пошук оптимальних архітектури та гіперпараметрів синтезованої нейронної мережі доцільно отримати тарифний план Google Colab Pro+. Він забезпечує підключення серверних відеокарт (наприклад, NVIDIA A100-SXM4-40GB [46]) та/або TPU. За 2 місяця витрати на цей тарифний план складають 4560 грн. Таким чином, орієнтовні витрати на реалізацію запропонованої моделі складають ≈ 58600 грн.

Висновки до розділу 3

В основу оцінки продуктивності запропонованої в роботі моделі прогнозу трафіку покладені дослідження впливу архітектури нейронної мережі та налаштування гіперпараметрів. В якості базової використовується метрика PercentMAE.

При цьому розглядаються архітектури CNN, LSTM, в тому числі з інтеграцією техніки Embedding та/або механізму уваги. Також аналізуються складні архітектури, що містять TL на основі MobilNetv3Smal, EfficientNet.

Підвищити продуктивність можливо, якщо нарощувати розмірність Conv1D від входу до виходу. Для випадку використання механізму уваги розроблені рекомендації щодо його підключення.

Окремо розглянуто питання застосування нейронних мереж для розпізнавання маскованого трафіку. Для цього виконується класифікація мережного трафіку за даними протоколу IPFIX.

Орієнтовні витрати на реалізацію запропонованої моделі глибкого навчання прогнозу трафіку сайту складають ≈ 58600 грн.

ВИСНОВКИ

Для реалізації моделі глибокого навчання нейронної мережі прогнозування трафіку сайту Інтернет доцільно використовувати TL. Це дозволяє більш ефективно проводити навчання нейронної мережі та подолання обмежень даних, а також прискорити розробку та покращити продуктивність.

На етапі створення датасету для прогнозу часового ряду реалізується кілька процедур: структуризація та обробка даних; усунення впливу тренда шляхом чисельного диференціювання часових рядів; формування вибірок.

Найбільш придатними архітектурами нейронних мереж для прогнозування трафіку сайту є CNN, RNN та/або їх комбінації, а також TimeGPT. Крім того, підвищити продуктивність нейронних мереж можна за рахунок інтеграції до їх архітектури техніки Embedding та/або механізму уваги.

При прогнозуванні трафіку сайту, зазвичай, обмежуються прогнозом на один крок або не більше 10 кроків вперед, через забезпечення балансу між точністю прогнозу та його практичною застосовністю.

Під час оцінки продуктивності запропонованої в роботі моделі прогнозу трафіку сайт Інтернет досліджувався вплив архітектури нейронної мережі та налаштування гіперпараметрів.

При цьому встановлено, що підвищити продуктивність можливо, якщо нарощувати розмірність Conv1D від входу до виходу. Для випадку використання механізму уваги розроблені рекомендації щодо його підключення.

Окремо розглянуто питання застосування нейронних мереж для розпізнавання маскованого трафіку. Для цього виконується класифікація мережного трафіку за даними протоколу IPFIX.

Орієнтовні витрати на реалізацію запропонованої моделі глибкого навчання прогнозу трафіку сайту складають ≈ 58600 грн.

Таким чином, результатами роботи є розроблена архітектура нейронної мережі для прогнозування трафіку сайту Інтернет; рекомендації щодо використання нейронної мережі для прогнозування трафіку сайту Інтернет. Вони можуть бути використані для подальших досліджень за даною тематикою та при проектуванні сайту Інтернет.