Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
(повна назва)
Кафедра Штучного інтелекту
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

Дослідження підходів до аугментації текстових даних
(тема)

Виконав:
студент 2 курсу, групи СШІм-22-1
Абросімов Є.О.
(прізвище, ініціали)

Спеціальність 122 Комп'ютерні науки
(код і повна назва спеціальності)
Тип програми освітньо-наукова
(освітньо-професійна або освітньо-наукова)
Освітня програма Системи штучного інтелекту
(повна назва спеціалізації)
Керівник доц. Дейнеко А.О.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ В.О. Філатов
(підпис) (прізвище, ініціали)

2024 р.

Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
(повна назва)

Кафедра Штучного інтелекту
(повна назва)

Рівень вищої освіти другий (магістерський)

Спеціальність 122 Комп'ютерні науки
(код і повна назва)

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

Освітня програма Системи штучного інтелекту
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«_____» _____ 20 ___ р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Абросімову Єгору Олеговичу
(прізвище, ім'я, по батькові)

1. Тема роботи Дослідження підходів до аугментації текстових даних

затверджена наказом університету від 1 квітня 20 24 р. № 260Ст

2. Термін подання студентом роботи до екзаменаційної комісії 11 червня 20 24 р.

3. Вихідні дані до роботи Теоретичне дослідження рішення та опис технологій, робота з текстовими даними, задача в предментній галузі, вибір методів аугментації, аугментація текстових даних, попередня підготовка даних, імітаційне моделювання, вибір моделі, аналіз результатів

4. Перелік питань, що потрібно опрацювати в роботі

1) Аналіз предметної галузі та постановка задачі дослідження

2) Аналіз сучасних методів та підходів до вирішення задачі класифікації текстових даних

3) Засоби реалізації практичної частини дослідження

4) Імітаційне моделювання

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Отримання завдання на кваліфікаційну роботу	01.04.2024	виконано
2	Аналіз предметної галузі та постановка задачі	04.04.2024-15.04.2024	виконано
3	Аналіз методів вирішення поставленої задачі	15.04.2024-20.04.2024	виконано
4	Вибір методів вирішення задачі	20.04.2024-25.04.2024	виконано
5	Імітаційне моделювання	01.05.2024-15.05.2024	виконано
6	Підготовка пояснювальної записки	16.05.2024-05.06.2024	виконано
7	Захист перед ЕК	11.06.2024	виконано

Дата видачі завдання 1 квітня 20 24 р.

Студент _____
(підпис)

Керівник роботи _____ доц. Дейнеко А.О.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка: 66 с., 12 рис., 6 табл., 1 дод., 27 джерел.

АУГМЕНТАЦІЯ, НАВЧАННЯ ЗІ ВЧИТЕЛЕМ, ТЕКСТОВІ ДАНІ, BERT, LLM.

Об'єкт дослідження – процес навчання класифікатора на текстових даних для розпізнавання фейкових новин, вплив методів генерації синтетичних даних на результати роботи алгоритмів.

Предмет дослідження – методи генерації синтетичних текстових даних.

Метою даного дослідження є виявлення найбільш ефективного методу генерації синтетичних текстових даних.

Методи дослідження – теорія обчислювального інтелекту; теорія штучних нейронних мереж; теорія оптимізації і статистичний аналіз; імітаційне моделювання.

Припускається що задача класифікації текстових даних, а разом з тим і аугментації табличних даних для виявлення таких характеристик як маніпуляції, емоційне забарвлення та стиль тексту, є вкрай важливою задачею в 21 столітті адже абсолютна більшість інформації в мережі Інтернет представлена в текстовому вигляді і її оптимальна обробка обіцяє значні покращення для нашого виду.

ABSTRACT

Master’s thesis contains: 66 pp., 12 fig., 6 tabl., 1 ann., 27 references.

AUGMENTATION, BERT, IMBALLANCED LEARNING, LLM, SUPERVISED LEARNING, TEXTUAL DATA.

The object of the research is the process of fitting a classifier to textual data for fake news detection and the influence of different text augmentation methods on systems’s performance.

The subject of research is textual data augmentation methods.

The purpose of the research is to find the most suitable synthetic text data generation method.

Research methods – theory of computational intelligence; theory of artificial neural networks; optimization theory and statistical analysis; simulation modeling.

It is assumed that the task of text data classification as well as textual data augmentation for detecting such characteristics as manipulations, emotional context is essential and relevant as textual data is the most common way of representing the information and potential ability to extract information efficiently promises very significant enhancements to our species.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

РНМ – рекурентна нейронна мережа;

ШІ – штучний інтелект;

ШНМ – штучна нейронна мережа;

AI – Artificial Intelligence – штучний інтелект;

DA – Data Augmentation – аугментація даних;

DL – Deep Learning – глибинне навчання;

LLM – Large Language Model – велика мовна модель;

ML – Machine Learning – машинне навчання;

NLP – Natural Language Processing – обробка природньої мови;

TF-IDF – Term Frequency-Inverse Document Frequency – статистичний показник, що використовується для оцінки важливості слів у контексті документа, що є частиною колекції документів чи корпусу.

ВСТУП

В сучасному світі, де дані є «новою нафтою», неструктуровані дані складають близько 80% [1] всіх даних що зберігаються в комерційних сховищах. Текстові дані є одним з найбільш розповсюджених типів неструктурованих даних.

При цьому в більшості компаній їх використання є доволі обмеженим. Це достатньо сильно контрастує з задачами що передбачають використання табличних даних адже такі задачі наразі є дуже популярними в більшості індустрій. Однак, не завжди можна отримати достатньо текстових даних для тренування ефективних моделей МН, особливо в умовах, коли їх використання обмежено правилами інтелектуальної властності та приватності.

При цьому аугментація текстових даних несе в собі ряд викликів і ризиків. До них належать вибір адекватних методів аугментації, які б зберігали семантичну цілісність тексту, та інтеграція цих методів у процес підготовки даних без негативного впливу на якість кінцевої моделі. Крім того, важливим аспектом є збереження балансу між оригінальними та аугментованими даними, щоб уникнути перенавчання моделей на синтетичних прикладах. Враховуючи ці аспекти, вибір та налаштування методів аугментації вимагають ретельного аналізу та експериментування.

Ціллю даної роботи є не тільки дослідження існуючих підходів до аугментації текстових даних, але й аналіз їх ефективності у різних сценаріях використання, з метою ідентифікації оптимальних стратегій аугментації, які можуть бути застосовані для конкретних задач NLP. Окрім того, робота має на меті визначити потенційні напрямки для подальших досліджень у цій галузі, сприяючи розвитку ефективніших та надійніших методів обробки текстових даних.

1 АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ ТА ПОСТАНОВКА ЗАДАЧІ ДОСЛІДЖЕННЯ

Підвищення рівня доступності до значних обсягів даних через цифрові платформи, значення технологій обробки природньої мови (NLP) стрімко підвищується. Системи ШІ трансформують цілі сфери людського життя. В даному дослідженні розглянуто групу підходів для покращення інтелектуального аналізу текстових даних, а саме data augmentation для текстових даних. Головна ціль аугментації даних – боротьба із перенавчанням інтелектуальних моделей. Дана ціль досягається генеруванням штучних даних, що підвищує різноманітність тренувального набору даних, а, значить, і узагальнюючу здатність моделей що використовують такі дані.

1.1 Текстові дані як джерело знань

Text mining являє собою комплексний підхід до аналізу текстових даних з метою екстракції корисної інформації та знань, з неструктурованих текстових даних. Text mining використовує методи обробки природної мови (NLP), машинного навчання та статистичного аналізу для розпізнавання шаблонів, тенденцій та зв’язків в текстових даних.

У сфері управління послугами текстовий майнінг відкриває нові можливості для покращення якості обслуговування та розробки персоналізованих пропозицій. Аналіз відгуків клієнтів, отриманих через соціальні мережі або платформи відгуків, може допомогти організаціям краще зрозуміти потреби та очікування своїх користувачів, а також ідентифікувати слабкі місця в наданих послугах.

У маркетингу та рекламі інтелектуальний аналіз текстових даних дозволяє аналізувати споживацькі настрої та тренди в реальному часі, що сприяє розробці більш ефективних маркетингових стратегій та рекламних

кампаній. Виявлення та аналіз трендів в соціальних медіа допомагає брендам залишатися на крок попереду конкурентів, швидко реагуючи на зміни у споживацьких уподобаннях.

У фінансовій сфері, зокрема в управлінні ризиками та fraud-monitoring, моделі інтелектуального аналізу текстових даних дозволяють ідентифікувати потенційні загрози та шахрайські дії шляхом аналізу незвичайних шаблонів у трансакціях, комунікаціях клієнтів та інших документах.

У галузі охорони здоров'я впровадження інтелектуального аналізу текстових даних сприяють ефективному використанню інформації з медичних записів для ідентифікації патернів, що можуть вказувати на певні захворювання, тим самим поліпшуючи діагностику та лікування пацієнтів.

Оглядаючи ці та інші застосування, можна зробити висновок, що інтелектуальний аналіз текстових даних є важливим інструментом для аналізу великих обсягів текстових даних, що надає цінну інформацію та знання для підтримки прийняття рішень у різних сферах діяльності. Його роль стає все більш значущою у світі, де дані є немовірно цінним ресурсом, а здатність ефективно аналізувати та інтерпретувати неструктуровану інформацію може стати ключовим конкурентною перевагою.

1.2 Навчання зі вчителем

Системи з використанням МН можна розділити на декілька груп залежно від способу тренування [2]: навчання з учителем, навчання без учителя, semisupervised learning та Reinforcement learning.

Навчання зі вчителем (Supervised Learning) є класом задач машинного навчання та штучного інтелекту. Воно визначається використанням наборів даних з цільовою змінною і має за мету створення та навчання інтелектуальних моделей, що можуть пронозувати цільову змінну для нових прикладів даних.

Коли вхідні дані подаються в модель, вона коригує свої ваги до того моменту, поки не буде належним чином налаштована, що відбувається як частина процесу тренування моделі. Навчання з учителем допомагає організаціям вирішувати різноманітні проблеми реального світу, іноді значно підвищуючи ефективність процесів та економлячи кошти.

Типовою задачею навчання з учителем є Виявлення спаму (Spam detection). Вона полягає у класифікації електронних листів чи повідомлень на дві категорії: спам (небажані) та не спам (корисні). На рисунку 1.1 схематично зображено принцип роботи системи Spam detection.

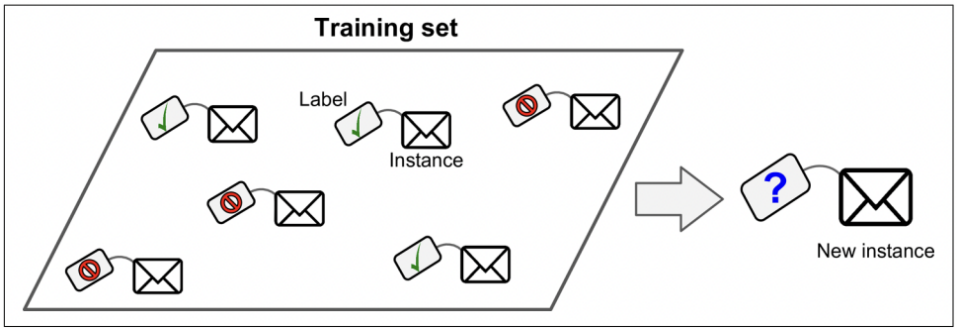


Рисунок 1.1 – Схема supervised learning на прикладі задачі детекції спаму

1.3 Підходи до класифікації текстових даних

Задача класифікації текстів була широко вивчена та розглянута протягом останніх кількох десятиліть [3]. Більшість систем класифікації текстів та категоризації документів можна розділити на наступні чотири основні етапи: генерація ознак, зниження розмірності, вибір моделі та оцінювання.

Вхідні дані являють собою певні сирі тексти. Формально, ми маємо набір даних: $D = \{X_1, X_2, \dots, X_n\}$, де X_i є окремим спостереженням в наборі даних. Кожне спостереження може бути представлено одним словом, послідовністю слів або речень. У відповідність до кожного спостереження

ставиться певний клас, при між реченням і міткою класу буде зв’язок типу «належить».

Далі задача полягає у перетворенні вхідних даних в структурований формат. Цей етап називається пошук числового представлення текстових даних. Зазвичай ціль – знайти найбільш оптимальне відображення текстових даних нефіксованої довжини у векторне представлення. Прикладом такого методу можна назвати алгоритм Term Frequency – Inverse Document Frequency (TF-IDF) [4]. Фомули 1.1–1.3 демонструють як обчислюється показник TF-IDF для кожного токена.

Частота терміну (Term Frequency – TF) обчислює, як часто термін з’являється у документі, що дозволяє визначити його значимість в рамках цього документа. Чим частіше слово з’являється у документі, тим вища його TF-оцінка. Однак, просто підрахунок частоти може спотворити аналіз, тому що загальновживані слова, як «і» або «також», можуть з’являтися часто, але не несуть значної інформативної ваги.

$$TFIDF = TF(t,d) \cdot IDF(t,D); \tag{1.1}$$

$$TF = \frac{count(t,d)}{|d|}; \tag{1.2}$$

$$IDF = log(\frac{log(|D|)}{|d \in D: t \epsilon d|}). \tag{1.3}$$

Обернена частота документів (Inverse Document Frequency – IDF) зменшує вагу термінів, які з’являються дуже часто в корпусі документів та, отже, можуть бути менш інформативними. IDF обчислюється як логарифм від частки загальної кількості документів у корпусі, поділеної на кількість документів, де зустрічається термін. Це зменшує вагу термінів, що зустрічаються майже у всіх документах, та збільшує вагу рідкісних термінів.

В результаті, TF-IDF є добутком двох цих показників, де кожне слово у документі отримує своє значення TF-IDF. Такі оцінки можуть використовуватися для ранжування важливості слова в контексті документа або для визначення, які слова є найбільш репрезентативними для даного документа у порівнянні з іншими.

Далі, необхідно створити структурований набір подальшого навчання моделі, який називаємо цей розділ feature extraction. Найважливішим кроком у класифікації тексту є вибір найкращого алгоритму класифікації. Наступна частина пайплайну – це крок оцінювання, який зазвичай відбувається з допомогою розділення вибірки на дві частини – тренувальна і тестова, які використовуються для тренування і тестування моделі відповідно. Загалом, система класифікації текстів містить чотири різні рівні обсягу, які можна застосувати:

– рівень документа: різні документи можуть мати різні класи;

– рівень абзацу: алгоритм має задачу класифікувати окремі частини документа, абзаци;

– рівень речення: алгоритм має задачу класифікувати окремі речення;

– рівень підречення: алгоритм має задачу класифікувати окремі частини речення.

1.4 Евристичні методи аугментації текстових даних

Аугментація це процес генерації нових синтетичних даних, схожих на реальні, та таких, які є репрезентативними для процесу що породжує реальні дані. Техніки аугментації використовуються в задачах, в яких наявних даних недостатньо.

Для більшості реальних задач МН недостатня кількість даних є значною проблемою адже алгоритми МН схильні до перенавчання. Основною ознакою перенавчання є зниження якості передбачення моделі на нових даних при збільшенні складності моделі. Аугментація є одним з

основних і дуже популярних засобів боротьби із перенавчанням в задачах Комп'ютерного зору. Однак, в задачах обробки природної мови застосування відповідних підходів є значно більш обмеженим.

1.4.1 Прості аугментації

Найпростішою групою підходів є прості аугментації (Simple data augmentations) що включають в себе випадкові видалення, вставку, заміну та обмін позиціями між словами або реченнями [5]. Вставка та заміна зазвичай відбувається випадковим словом зі словника. Дана група підходів використовується найчастіше, хоч і має обмежену ефективність.

Прості аугментації [6] що базуються на правилах, є доволі популярним інструментами. Це передбачає програми виду «if-else» що використовують шаблони для вставки та перестановки існуючих даних. В роботі пропонується чотири методи аугментації текстових даних. В таблиці 1.1 зображено приклади їх використання.

Таблиця 1.1 – Приклади роботи чотирьох основних методів простої аугментації даних.

Назва методу	Короткий приклад
Random Swap	I am jogging ⇒ I tiger jogging
Random Insertion	I am jogging ⇒ I am salad jogging
Random Deletion	I am jogging ⇒ I jogging
Random Synonym Replacement	I am jogging ⇒ I am running

Існує певна перспектива для розвитку даного напрямку генерації синтетичного тексту. З точки зору побудови кращих моделей МН, метод Випадкової вставки може бути доволі корисним. Він дозволяє розширювати майбутній словник, який використовуватиме модель, і, як результат підвищувати робастність моделі і її здатність до узагальнення патернів що

криються в навчальних даних. Даний напрям аугментації текстів, не зважаючи на свою простоту, може бути доволі корисним в реальних проектах.

1.4.2 Метод зворотнього перекладу

Досі було розглянуто підходи, що не вимагають моделей МН. Першим підходом що в більшості випадків використовує інтелектуальні моделі обробки даних є Back translation [7]. Суть підходу полягає в тому щоб перекласти текст з мови оригіналу на довільну (зазвичай схожої мовної групи) мову, а отриманий результат знову перекласти мовою оригіналу. Цей підхід є доволі доступним і його ефективність основною мірою залежить від специфіки задачі та моделі для перекладу тексту.

Попри значну ефективність даного методу, він також має свої обмеження. Наприклад, проблемними є слова або висловлювання, які існують та використовуються в одній мові, але відсутні в іншій. Це викликано принциповою різницею у морфології між певними мовами. Ця проблема проявляється найсильніше при використанні перекладу на мову іншої мовної групи. На рисунку 1.2 схематично представлено логіку роботи Back translation.

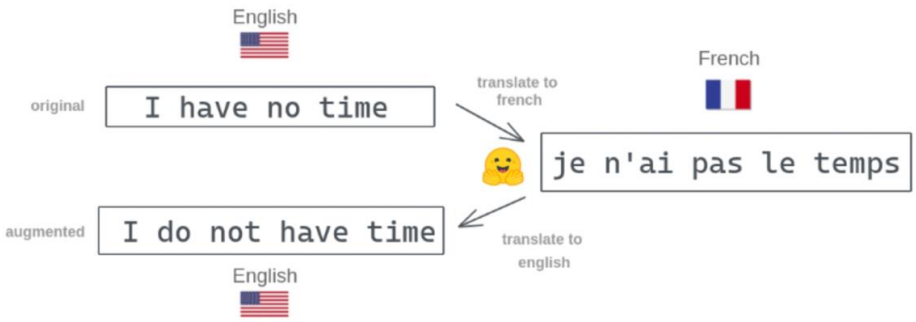


Рисунок 1.2 – Схематичне представлення роботи методу Back translation

1.5 Методи МН для аугментації текстових даних

В останні роки було здійснено величезний прогрес в багатьохзадачах інтелектуального аналізу текстових даних. Генерація синтетичних текстових даних є важливим методом аугментації датасету в задачах класифікації текстових даних. Цей підхід дозволяє збільшити обсяг тренувальних даних, що допомагає покращити продуктивність моделей машинного навчання, особливо в умовах обмеженої кількості реальних даних. Синтетичні дані можуть створюватися за допомогою різних технік, таких як парафразування, вставка чи видалення слів, використання синонімів або автоматичне генерування текстів за допомогою мовних моделей на кшталт GPT-3.

Окрім збільшення розміру датасету, синтетичні дані допомагають моделі стати більш стійкою до варіацій у текстах, що може поліпшити її здатність до узагальнення та зменшити ризик перенавчання. Це особливо корисно у випадках, коли модель може зіткнутися з новими, невідомими раніше виразами або стилями написання. Завдяки аугментації, модель навчається розпізнавати більш широкий спектр можливих варіантів вхідних даних, що підвищує її ефективність у реальних умовах.

Генеративна аугментація, безумовно, є найбільш перспективним фронтиром в контексті задачі що розглядається в даній роботі. Найпопулярніше рішення задачі аугментації генеративними моделями це використання pre-trained моделей, з опціональним донавчанням [8]. Зазвичай донавчання відбувається на supervised learning задачах. В подальшому приховані шари нейронної мережі (ШНМ) та їх ваги використовуються для створення нової моделі, що виконує задачу трансформації тексту. Доволі популярною, в рамках цього підходу є модель Robustly optimized BERT (RoBERTa).

Для використання pre-trained моделей ключовим моментом є якість попереднього навчання моделі. Чим ефективніша загальна модель, тим ефективнішим буде результат від використання Transfer Learning.

Мовне моделювання виступає як дуже важливий етап попередньої підготовки даних. Він дає можливість значно розширити тренувальний набір даних. Це особливо важливо що недостатні розміри дата сету можуть стати різницею між успіхом моделі та

Також великої популярності наразі набирає застосування великих мовних моделей (LLM) для даної задачі. Цей підхід передбачає використання promt-engineering для створення запитів, що змушують модель надавати найбільш оптимальну відповідь. Однією з найбільш популярних LLM моделей є GPT-3. Ця модель, розроблена компанією OpenAI, може виконувати широкий спектр завдань від відповідей на запитання до генерації синтетичних текстових даних.

2 АНАЛІЗ СУЧАСНИХ МЕТОДІВ ТА ПІДХОДІВ ДЛЯ ВИРІШЕННЯ ЗАДАЧІ КЛАСИФІКАЦІЇ ТЕКСТОВИХ ДАНИХ

2.1 Класифікація текстових даних

Класифікація тексту є дуже важливим завданням, і воно знаходить все більше застосування в реальних бізнес-задачах. Методи класифікації текстів мають спільну мету – визначення найбільш відповідної мітки класу для вхідного тексту.

Мітки класу завжди заздалегідь відомі і являють собою певну якісну характеристику [9]. Приклад такої якісної характеристики можна знайти в задачі Sentiment analysis, аналіз тональності тексту. Ціль методів Sentiment analysis полягає в класифікації текстів за емоційним забарвленням. Класичним прикладом задачі аналізу тональності можна назвати класифікацію відгуків інтернет-магазинів на позитивні та негативні. На базі такої моделі можна побудувати систему, що може значно економити людські ресурси, що витрачаються робітниками на обробку кожного з відгуків.

Середі інших класичних прикладів задач класифікації текстів можна перелічити наступні:

– sentiment analysis: задача класифікації афективних станів та суб'єктивної інформації, що міститься в тексті. Часто розділяючим фактором є саме емоційне забарвлення;

– topic classification: задач визначення однієї чи декількох тем для тексту (тобто його тем);

– класифікація новин: задач присвоєння категорій до новинних матеріалів, таких як теми або інтереси користувачів;

– відповіді на запитання: завдання вибору відповіді на запитання, з потенційних речень-кандидатів, зазвичай отриманих зі стороннього

документу. Це задач зазвичай формулюється як бінарна або багатокласова класифікація;

– розпізнавання іменованих сутностей: завдання виявлення іменованих сутностей у неструктурованому тексті, маркуючи їх визначеними категоріями;

– синтаксичний аналіз: серія задач, пов'язаних з прогнозуванням морфосинтаксичних властивостей слів, таких як маркування частин мови, семантичне маркування частин речення;

– детекція фейкових новин: задачі пов’язані з ідентифікацією неправдивої інформації та маніпуляцій у новинних статтях та заголовках.

Швидкість, з якою наразі створюється текстова інформація, давно перевершила «ручні» рішення для цих завдань. Це означає, що методи класифікації текстів є не лише корисними, але й суворо необхідними. Саме тому можна зробити висновок, що розробка точних та неупереджених систем класифікації текстів є доволі важливою задачею.

Існує велика кількість, як класичних, так і доволі нових методів класифікації текстових даних. Їх можна умовно розділити на ті що використовують Deep learning і ті що використовують класичні моделі МН, Shallow learning.

2.1.1 Ймовірнісні моделі

Класифікатори Наївного Баєса (Naïve Bayes) – це підмножина алгоритмів машинного навчання з учителем, які використовуються для завдань класифікації, зокрема в класифікації текстів [10].

Наївний Баєс належить до родини алгоритмів навчання зі вчителем, що має на меті моделювати розподіл вхідних даних певного класу або категорії. На відміну від дискримінативних класифікаторів, таких як логістична регресія, він не вчиться визначати, які ознаки найважливіші для розрізнення між класами.

Наївний Баєс також відомий як імовірнісний класифікатор, оскільки він базується на теоремі Баєса. Було б складно пояснити цей алгоритм без основ статистики Баєса. Ця теорема, яка також відома як формула Баєса, дозволяє нам «інвертувати» умовні ймовірності. Нагадаємо, що умовні ймовірності представляють ймовірність настання однієї події за умови, що інша подія вже відбулася, що представлено формулою:

$$P(Y|X) = \frac{P(X \cup Y)}{P(X)}. \quad (2.1)$$

Теорема Баєса використовує послідовні події, де додаткова інформація, отримана пізніше, впливає на початкову ймовірність. Ці ймовірності позначаються як апріорна ймовірність та апостеріорна ймовірність. Апріорна ймовірність – це початкова ймовірність події до її контекстуалізації за певною умовою або маргінальна ймовірність. Апостеріорна ймовірність – це ймовірність події після спостереження за даними.

Популярним прикладом застосування принципів описаних Баєсом можна назвати задачу тестування ефективності медичних препаратів. Наприклад, уявімо, що є особа на ім'я Джейн, яка проходить тест, щоб визначити, чи має вона діабет. Припустімо, що загальна ймовірність наявності діабету становить 5%; це буде наша апріорна ймовірність. Однак, якщо вона отримує позитивний результат тесту, апріорна ймовірність оновлюється, щоб врахувати цю додаткову інформацію, і тоді вона стає перетворюється на апостеріорну ймовірність яку надалі можна використовувати в розрахунках. Цей приклад може бути представлений наступним рівнянням за допомогою теореми Баєса:

$$P(\text{діабет} \mid \text{тест}+) = \frac{P(\text{тест}+ \mid \text{діабет})\, P(\text{діабет})}{P(\text{тест}+)}. \quad (2.2)$$

Де «тест+» означає наявність позитивного тесту на діабет, а «діабет» – наявність діабету.

Класифікатори Наївного Баєса працюють трохи інакше, адже базуються на декількох ключових припущеннях, через що їх і називають «наївними». Вони передбачають, що предиктори у моделі Наївного Баєса є умовно незалежними, тобто не пов'язані з будь-якою іншою ознакою в моделі. Також припускається, що всі ознаки рівнозначно впливають на результат. Хоча ці припущення часто порушуються у реальних сценаріях (наприклад, наступне слово в електронному листі залежить від попереднього слова), це спрощує проблему класифікації, роблячи її більш обчислювально виконуваною. Тобто, тепер буде потрібна лише одна ймовірність для кожної змінної, що, в свою чергу, спрощує обчислення моделі. Незважаючи на це нереалістичне припущення про незалежність, алгоритм класифікації добре себе показує, особливо при малих розмірах вибірки [11].

З урахуванням цього припущення, ми можемо зараз більш детально розглянути частини класифікатора Наївного Баєса. Подібно до теореми Баєса, він використовуватиме умовні й апріорні ймовірності для обчислення апостеріорних ймовірностей за наступною формулою:

$$P(Y|X) = \frac{P(X|Y)P(Y)}{P(X)}. \quad (2.3)$$

А значить, для того аби отримати найбільш імовірний Y необхідно використати наступну формулу:

$$\hat{y} = arg\ max\ P(Y|X) = argmax\ \frac{P(X|Y)P(Y)}{P(X)}. \quad (2.4)$$

А якщо зважати що для кожного з варіантів Y, знаменник P(X) буде незмінним, можемо прибрати його з розрахунку та отримати:

$$\hat{y} = arg\ max\ P(Y|X)\ =\ argmax\ P(X|Y)P(Y). \qquad (2.5)$$

2.1.2 Метричні моделі

Наступна група методів – підходи засновані на KNN. В основі алгоритму K-Nearest Neighbors (KNN) полягає класифікація нерозміченого зразка шляхом знаходження класу з найбільшою кількістю зразків серед k найближчих спостережень [12]. Це простий класифікатор, який не вимагає побудови моделі та може зменшувати складність через швидкий процес визначення k найближчих сусідів. Рисунок 2.1 демонструє принцип роботи KNN. Ми можемо знайти k навчальних текстів, які наближаються до певного тексту, який потрібно класифікувати, шляхом оцінювання відстані між ними. Таким чином, текст може бути поділений на найпоширеніші категорії, знайдені в текстах навчального набору k.

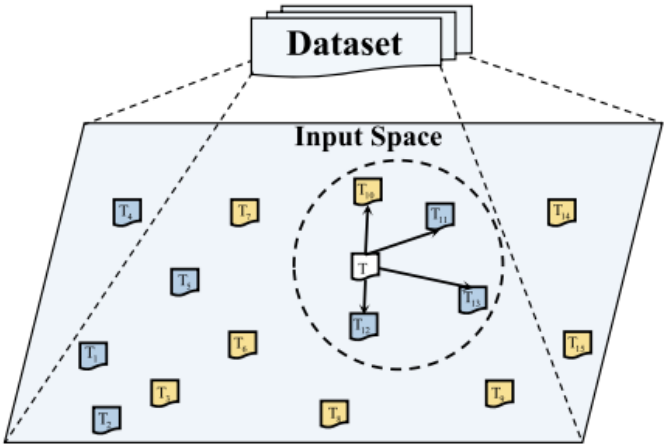


Рисунок 2.1 – Схематичне представлення роботи методів сімейства KNN

Для даних методів характерним є квадратичне зростання кількості обчислень пропорційно до кількості спостережень, а тому їх

використання на великих наборах даних часто вважається недоцільним. При цьому дане сімейство алгоритмів має схильність до зміщення передбачень на користь класів що представлені більшою кількістю спостережень.

Серед метрик які найчастіше використовують для вимірювання відстані можна назвати Євклідову відстань та Манхетенську відстань [13]. Їх графічні ілюстрації принципу їх роботи наведено на рисунку 2.2.

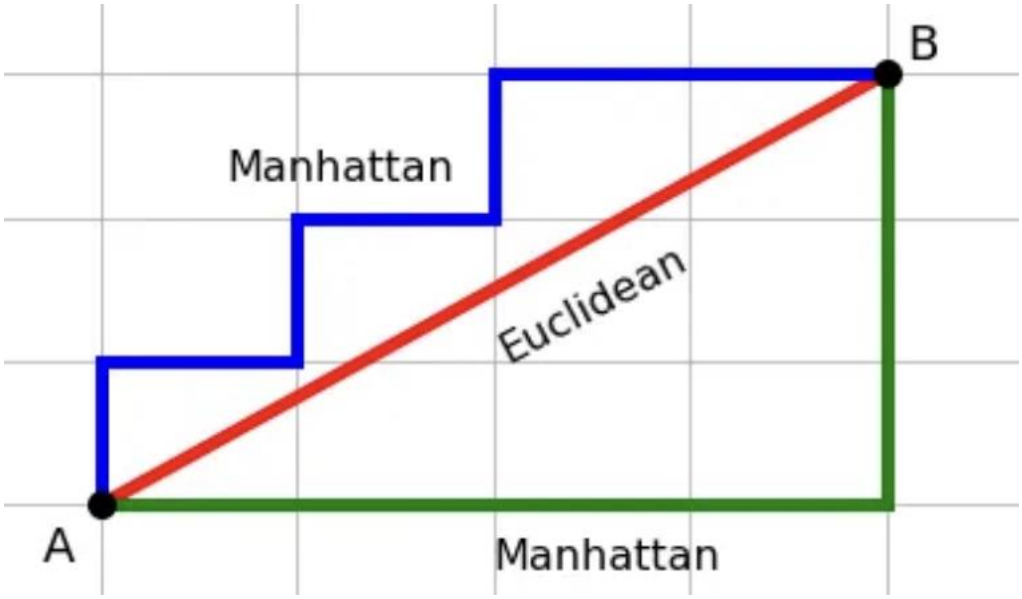


Рисунок 2.2 – Схематичне представлення принципу роботи метрик відстані

2.1.3 Деревовидні моделі

Дерева рішень є одним із потужних методів, які широко використовуються в різних галузях, таких як Fraud detection або оцінка ризику. Дерева рішень представляють собою послідовну модель, яка на кожному етапі порівнює числову ознаку з пороговим значенням і, знайшовши найбільш оптимальний поділ, ітеративно розподіляє спостереження, створюючи правила класифікації. Такі концептуальні

правила можна отримувати доволі ефективно методом послідовного перебору можливих сплітів.

Дерево рішень являє собою модель що ітеративно застосовує вивчені правила для класифікації нових спостережень [14]. Важливо також зазначити що існує дуже багато інструментів для інтерпретації Tree-based алгоритмів. Кожне дерево складається з вузлів, що символізують собою правило поділу і гілки, що являють собою результуючі підвибірки отримані від поділу. Завдяки їх простоті аналізу та точності на багатьох формах даних, дерева рішень знайшли багато сфер застосування. Рисунок 2.3 є демонстрацією принципу роботи алгоритму дерев рішень.

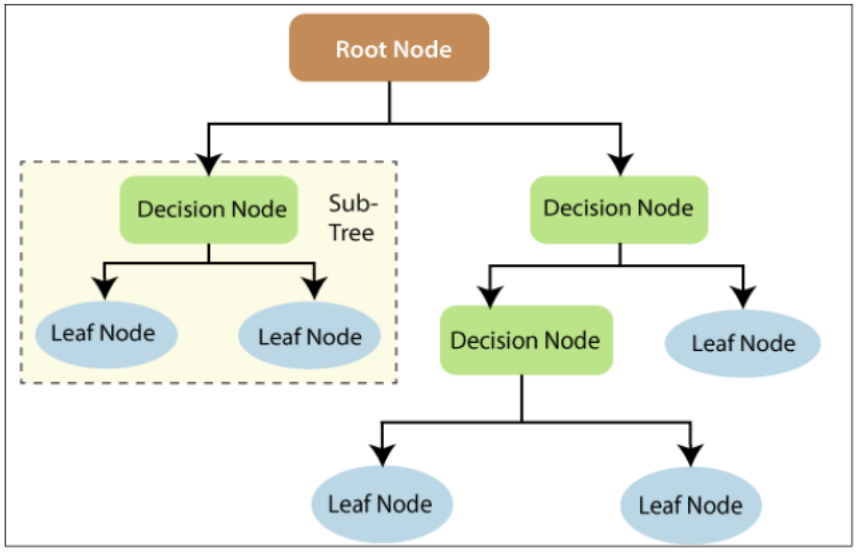


Рисунок 2.3 – Схематичне представлення алгоритму дерев рішень

2.1.4 Модель багатошарового перцепртрона

Багатошаровий перцептрон [15], відомий також як MLP, який іноді неофіційно називають «ванільною» нейронною мережею, має повністю з'єднані шари нейронів, які можуть з допомогою ряду математичних перетворень трансформувати дані будь-якої вхідної розмірності до бажаної

розмірності. Багатошаровий перцептрон — це нейронна мережа, що містить мінімум 3 шари нейронів. Для створення нейронної мережі ми комбінуємо нейрони так, що виходи нейронів попереднього шару подаються на вхіднейронам інших шарів.

Багатошаровий перцептрон має один вхідний шар, один вихідний шар, і може мати будь-яку кількість прихованих шарів, кожен з яких може містити будь-яку кількість нейронів. На рисунку 2.4 нижче зображено ілюстрацію принципу роботи та структури багатошарового перцептрона (MLP).

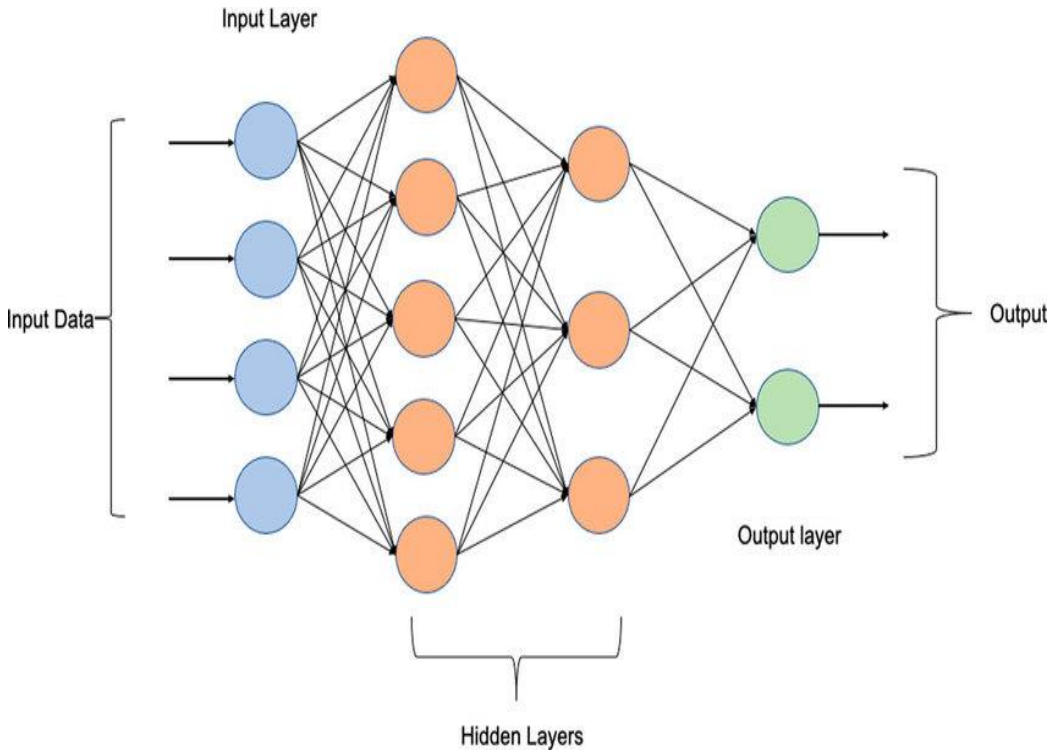
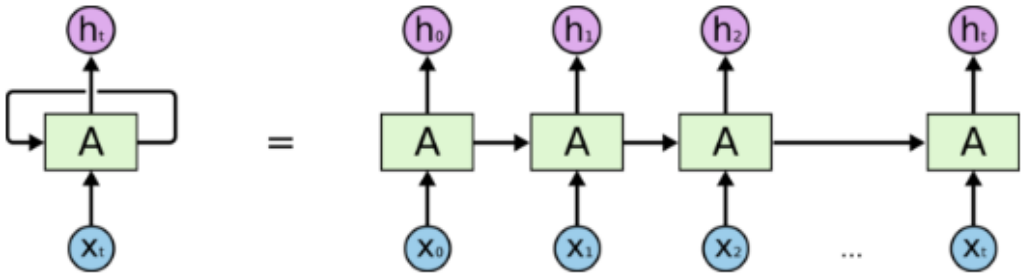


Рисунок 2.4 – Схематичне представлення багатошарового перцептрону (MLP)

Варто також зазначити той аспект що для успішного застосування MLP, як і інших моделей наведених вище, для текстових даних, певним чином перетворити текстові дані на числові.

2.1.5 Рекурентні нейронні мережі

Рекурентна нейронна мережа (RNN) – це тип штучної нейронної мережі, яка використовує послідовні дані або дані часових рядів [16]. Ці алгоритми глибинного навчання часто застосовуються для вирішення порядкових або часових завдань, таких як переклад мови, обробка природної мови, розпізнавання мови та підписування зображень; вони включені до популярних застосунків, таких як Siri, голосовий пошук та Google Translate. На відміну від нейронних мереж прямого поширення і згорткових нейронних мереж, рекурентні нейронні мережі вирізняються своєю «пам'яттю», оскільки вони використовують інформацію з попередніх входів для впливу на поточний вхід та вихід. На рисунку 2.5 наведено загальну ілюстрацію принципу роботи рекурентної нейронної мережі.



An unrolled recurrent neural network.

Рисунок 2.5 – Схематичне представлення принципу роботи рекурентної нейронної мережі (RNN)

Інша особливість рекурентних мереж полягає в тому, що між усіма шарами мережі співпадають параметри. На відміну від MLP, де ваги різняться для кожного вузла, рекурентні нейронні мережі мають однакові вагові параметри у кожному шарі мережі. Тим не менш, ці ваги все ще коригуються через Back Propagation для сприяння навчанню з підкріпленням.

Рекурентні нейронні мережі використовують алгоритм зворотного поширення помилок у часі (BPTT) для визначення градієнтів, що дещо відрізняється від традиційного зворотного поширення, оскільки він спеціалізований на послідовних даних. Принципи BPTT такі самі, як у традиційному зворотному поширенні, де модель навчається, розраховуючи помилки від свого вихідного шару до вхідного. Ці розрахунки дозволяють належно налаштувати параметри моделі. BPTT відрізняється від традиційного підходу тим, що він сумує помилки на кожному часовому кроці, тоді як мережі прямого поширення не потребують сумування помилок, оскільки вони не ділять параметри між шарами.

Цей процес може призвести до двох проблем у РНМ, відомих як exploding gradients та vanishing gradients. Ці проблеми визначаються розміром градієнта, який є тангенсом кута нахилу лінії дотичної до гіперплощини значень функції втрат. Коли градієнт занадто малий, він стає ще меншим, оновлюючи вагові параметри до тих пір, поки вони не стануть незначущими, тобто 0. Це призводить до того, що алгоритм більше не навчається. «Вибухові градієнти» виникають, коли градієнт занадто великий, що створює нестабільну модель. У цьому випадку ваги моделі зростають надто великими і з часом комп’ютер просто може втратити здатність оброблювати такі значення.

2.2 Методи попередньої підготовки даних

Техніки попередньої обробки даних відіграють ключову роль у видаленні небажаних слів, символів чи пунктуації, які є нерелевантними для моделей МН [17]. Таким чином, у сфері обробки природної мови існує чимало технік, які можна обрати в залежності від конкретної задачі. Послідовність застосування цих технік також має велике значення у деяких ситуаціях. Техніки попередньої обробки даних або тексту переважно

потрібні для перетворення сирих даних до структурованого вигляду, який є підходящим для використання моделями МН.

2.2.1 Сегментація речень

Розбиття на речення, яке також відоме як виявлення меж речень, є процесом поділу текстового документу або корпусу на окремі речення. Це сприяє визначенню меж речень, що дозволяє обробляти кожне речення окремо. Розділення відбувається в основному за наявності крапки або інших розділових знаків за допомогою токенізатора.

2.2.2 Переведення слів до нижнього регістру

Тексти зазвичай містять абревіатури та великі літери. Даний етап на практиці часто ігнорується, але він є одним із найпростіших і ефективних кроків попередньої обробки тексту, особливо у випадках, коли дата сет є значно розрідженим. Так, наявність у тексті варіацій одного слова, що відрізняються тільки великими літерами (наприклад, 'India' проти 'india') можуть призводити до значного зниження ефективності моделей. Це означає, два однакові слова, одне з яких написано великими літерами, а інше маленькими, модель сприймає як два різні слова, і на етапі створення Word Embeddings ми можемо отримати два різні вектори слів. Тому переведення до нижнього регістру є простою, але, дуже ефективною практикою в попередній обробці тексту.

2.2.3 Токенізація

Токенізація визначається як розбиття речень на слова, символи, розділові знаки, які в цілому називаються токенами. Розбиття здебільшого базується на наявності пробілу або розділового знаку. Цей крок сприяє

видаленню небажаних і непередбачуваних результатів у подальших етапах обробки. Наприклад, речення «NLP is the future of Speech Recognition Systems!» буде токенізовано як: «NLP», «is», «the», «future», «of», «Speech», «Recognition», «Systems», «!». Відтак, токенізація дозволяє обробляти слова окремо, аналізуючи речення по частинах, а не в якості одного цілого.

2.2.4 Видалення стоп-слів

Слова, такі як «the», «are», «is», «and» тощо, зазвичай не несуть значної кількості інформації, за винятком певних специфічних задач. Наприклад, у випадку класифікації текстів або документів цим додатковим словам не надається ваги. Виділяються лише ключові слова, які формують семантичну структуру висловювання. Таким чином, чим краще ідентифіковані та видалені ці стоп-слова, тим кращі результати алгоритмів класифікації.

При цьому варто зауважити, що в певних випадках видалення стоп-слів, як-от у розмовних моделях, наявність певних слів негації, таких як «no», «cannot», «won’t», «not», має вирішальне значення для визначення контексту речення та його сенсу.

2.2.5 Видалення розділових знаків

Алгоритми МН зазвичай не здатні правильно інтерпретувати наявність в тексті розділових знаків, тому їх наявність до певної міри зашумлює текст. Зокрема, неструктуровані документи часто містять численні знаки оклику, апострофи, коми тощо. Регулярні вирази (Regular expressions) є найбільш поширеним методом пошуку шаблонів, який застосовується до рядків для ідентифікації та заміни розділових знаків за загальним правилом, наприклад, на знак «пробіл».

2.2.6 Стемінг та лематизація

Лематизація полягає у видаленні або заміні суфікса слова для перетворення його в базову форму, яку називають лемою. Лема завжди є значущим словом, на відміну від слова, отриманого шляхом стемінгу. Лематизація є важливим та ефективним етапом попередньої обробки тексту в області обробки природної мови. Наприклад, лемою слова «Caring» буде «Care», що є значущим словом.

При цьому ще варто зазначити що момент застосування стемінгу або лематизації також має вирішальне значення. Так, наприклад, якщо стоп-слова, такі як «is», «the», «in», видаляються з морфологічного аналізу, то морфологічний аналізатор може неправильно обробити певні частини тексту, що призведе до некоректних результатів.

2.3 Метрики класифікації

Вибір підходящої метрики оцінювання є критично важливим аспектом у розробці моделей машинного навчання, оскільки він значною мірою впливає на інтерпретацію їхньої ефективності та слугує головним критерієм в процесі вибору моделі. Метрики, які використовуються для оцінки моделей, повинні відображати реальні вимоги до системи та бути узгодженими з бізнес-цілями [18]. Наприклад, у задачах бінарної класифікації, таких як визначення шахрайства, вибір між точністю (precision) та відновленням (recall) може залежати від вартості помилок. Якщо вартість помилки першого роду (наприклад, невиправдане блокування користувача) нижча за помилку другого роду (невиявлене шахрайство), то одним з більш оптимальних варіантів може бути використання метрики Recall.

Також, для комплексної оцінки моделі, використання єдиної метрики може бути недостатнім. У комплексних задачах, таких як медична

діагностика, використання комбінації Sensitivity та Specificity або розрахунок площі під кривою ROC (AUC-ROC) може надати більш всебічне розуміння продуктивності моделі. Такий підхід дозволяє оцінювати здатність моделі розрізняти між класами з урахуванням різних порогів класифікації, що особливо важливо, коли витрати на помилки відрізняються.

У сферах, де помилки мають асиметричні наслідки, як у фінансовому моделюванні чи охороні здоров'я, вибір метрики, яка може адекватно врахувати вартість помилок, є особливо критичним. Наприклад, у кредитному скорингу, помилка у відмові кредиту може мати менші наслідки, ніж помилкове схвалення ризикованого кредиту. Тому, метрики, такі як F1-оцінка або точність, стають більш релевантними, оскільки вони забезпечують баланс між відновленням та точністю.

Крім того, вибір метрики може впливати на спосіб тренування моделі та її оптимізацію. Наприклад, у завданнях регресії вибір між середньоквадратичною помилкою (MSE) та середньою абсолютною помилкою (MAE) може залежати від розподілу помилок у даних. MSE підсилює вплив великих помилок, що може бути корисним, коли великі відхилення є критичними для додатку, але може призводити до чутливості до викидів. Це підкреслює важливість узгодження метрики з особливостями поставленої задачі.

2.3.1 Матриця помилок

Confusion matrix (або матриця помилок) є одним із способів узагальнення результатів роботи класифікатора у задачах бінарної класифікації. Ця квадратна матриця складається зі стовпців та рядків, які відображають кількість випадків у формі абсолютних або відносних відношень «фактичний клас» проти «прогнозований клас». Нехай P буде міткою класу 1, а N буде міткою другого класу або міткою всіх класів, які

не є класом 1 у багатокласовому налаштуванні. Тоді матриця помилок буде схематично мати вигляд як на рисунку 2.6.

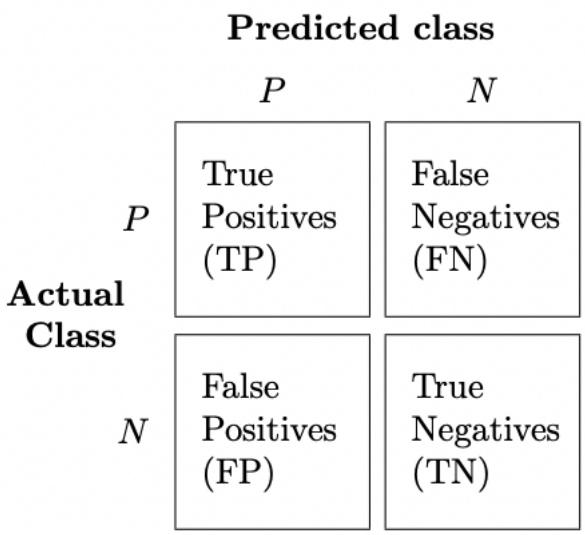


Рисунок 2.6 – Матриця помилок (Confusion matrix)

2.3.2 Точність

Як помилка прогнозування (Error rate), так і зворотна метрика, точність (Accuracy) надають загальну інформацію про кількість некоректно класифікованих записів. Помилка може бути інтерпретована як сума всіх неправильних прогнозів, поділена на загальну кількість прогнозів, тоді як точність розраховується як сума правильних прогнозів, поділена на загальну кількість прогнозів. Обидві метрики є ключовими показниками для оцінки ефективності моделей машинного навчання.

$$\text{Error rate} = \frac{FP+FN}{FP+FN+TP+TN} = 1 - Accuracy; \quad (2.6)$$

$$Accuracy = \frac{TP+FN}{FP+FN+TP+TN} = 1 - Error\ rate. \quad (2.7)$$

2.3.3 Пропорція істинно позитивних та хибно позитивних прикладів

Пропорція істинно позитивних результатів (TPR) та пропорція хибно позитивних результатів (FPR) є метриками, що є особливо корисними для задач в яких присутній дисбаланс класів. Як приклад, у задачі детекції спам повідомлень (Spam Detection) нас, звичайно, перш за все цікавить виявлення та фільтрація спаму. Однак також важливо зменшити кількість повідомлень, які були неправильно класифіковані як спам (хибно позитивні результати): ситуація, коли людина пропускає важливе повідомлення, вважається «гіршою», ніж ситуація, коли у поштовій скриньці людини з'являється кілька спам-повідомлень. На відміну від FPR, швидкість істинно позитивних результатів надає корисну інформацію про частку позитивних (або відповідних) зразків, які були правильно ідентифіковані з загальної кількості позитивів.

$$\mathrm{F}PR = \frac{FP}{N} = \frac{FP}{FP+TN}, \tag{2.8}$$

$$\mathrm{T}PR = \frac{TP}{P} = \frac{TP}{FN+TP}. \tag{2.9}$$

2.3.4 Влучність, Повнота та F1-оцінка

Влучність та Повнота є показниками, які широко використовуються в інформаційних технологіях та пов'язані з показниками хибно позитивних та істинно позитивних результатів. Насправді, Відновлення є синонімом Істинно Позитивної Пропорції і також іноді називається Чутливістю. F1-score може бути інтерпретований як комбінація обох цих показників, Точності та Відновлення.

$$\mathrm{P}recision = \frac{TP}{TP+FP}; \tag{2.10}$$

$$Recall = \frac{TP}{TP+FN}; \quad (2.11)$$

$$Accuracy = \frac{TP+TN}{TP+FP+FN+TN}; \quad (2.12)$$

$$F1\ Score = 2 \times \frac{Precision \times Recall}{Precision+Recall}. \quad (2.13)$$

2.3.5 ROC

Графіки Receiving operating characteristic (ROC) є корисними інструментами для вибору моделей класифікації на основі їхньої продуктивності з урахуванням істинно позитивних та хибно позитивних пропорцій. Діагональ графіка ROC може бути інтерпретована як випадкове передбачення, і моделі класифікації, показники яких знаходяться нижче за діагональ, вважаються гіршими за випадкове передбачення. Ідеальний класифікатор займе верхній лівий кут графіка з Істинно Позитивною Пропорцією, рівною 1, та Хибно Позитивною Пропорцією, рівною 0. У практичних застосуваннях, крива ROC часто використовується для порівняння різних моделей, оскільки вона наочно демонструє їхню здатність розрізняти між класами.

Крива ROC може бути розрахована шляхом ітеративної зміни порога рішення класифікатора (наприклад, апостеріорні ймовірності класифікатора наївного Баєса). На основі кривої ROC може бути розрахована так звана площа під кривою (AUC), що дозволяє оцінити продуктивність моделі класифікації. Чим більша площа під кривою, тим кращою є модель у розрізненні між позитивними і негативними класами, причому AUC, що наближається до 1, вказує на високу якість моделі. На рисунку 2.7 представлено приклад графіка кривої ROC.

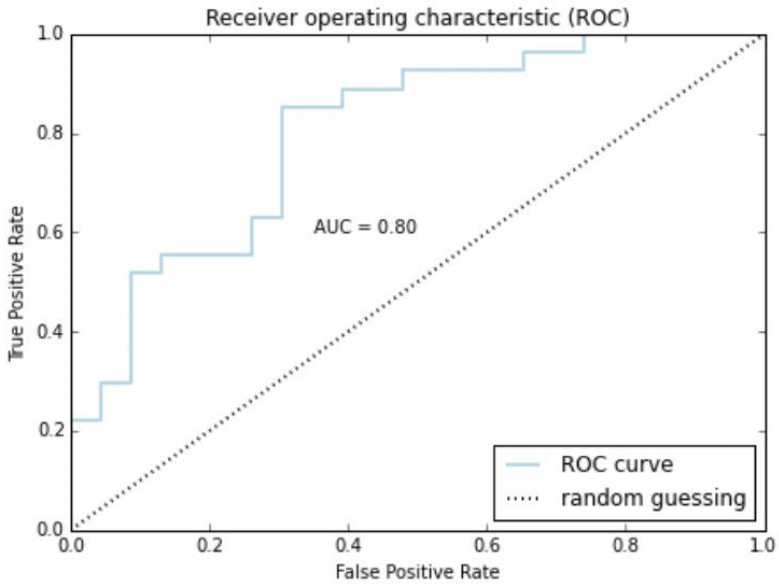


Рисунок 2.7 – ROC крива

2.4 Методи аугментації текстових даних

Аугментація текстових даних є важливим інструментом в області обробки природної мови (NLP), що дозволяє розширити тренувальний набір даних шляхом створення нових синтетичних прикладів. Це досягається за допомогою різних технік, таких як парафразування, вставка або видалення слів, а також генерація тексту за допомогою алгоритмів машинного навчання. Використання таких методів забезпечує моделі більшу різноманітність даних, що допомагає уникнути перенавчання і покращує здатність моделі до генералізації. Таким чином, аугментація сприяє підвищенню стабільності та точності моделей при обробці текстів, особливо в умовах обмеженої кількості вихідних даних.

Методи аугментації текстових даних потенційно можуть значно впливати на продуктивність моделей NLP. Розширений тренувальний набір дозволяє моделі навчитися краще розпізнавати патерни та семантичні зв’язки в текстах, що призводить до покращення якості класифікації, перекладу, аналізу тональності та інших задач. Крім того, аугментація

допомагає зробити модель більш стійкою до варіацій вхідних даних, що є важливим у реальних сценаріях застосування, де текстова інформація може мати різноманітні стилі та формати. В результаті, використання методів аугментації текстових даних є ключовим фактором для підвищення ефективності та надійності сучасних систем обробки природної мови.

2.4.1 Прості текстові аугментації

Прості аугментації тексту (Easy Data Augmentation – EDA) являють собою чотири основні підходи: заміна синонімів, випадкове вставлення, випадкове переміщення та випадкове видалення [19]. Заміна синонімів передбачає випадковий вибір слів з речення, які не є стоп-словами, і їх заміну на випадково обрані синоніми. Випадкове вставлення включає вибір випадкового слова в реченні, знаходження його синоніма та вставку цього синоніма в випадкову позицію в реченні. Випадкове переміщення полягає у випадковому виборі двох слів в реченні і їх взаємному переміщенні. Випадкове видалення – це випадкове видалення кожного слова в реченні з певною ймовірністю.

Сила методів EDA полягає в їх здатності зменшувати перенавчання та покращувати загальну продуктивність моделей, особливо на малих наборах даних. Завдяки збільшенню різноманітності тренувальних даних, моделі стають більш стійкими та краще розпізнають патерни і семантичні зв’язки в текстах. Проте, існують і обмеження. Надмірне застосування цих методів може призвести до введення небажаного шуму, що може погіршити якість моделі. Саме тому необхідно обирати методи аугментацій, які підходять до кожної окремої задачі та здатні продукувати саме такі синтетичні дані, які допоможуть моделям МН узагальнювати патерни та приховані взаємозв’язки, що криються в даних. Ретельний підхід до вибору методів аугментації є критичним для забезпечення успішного застосування EDA в реальних сценаріях.

2.4.2 Зворотній переклад для аугментації текстових даних

Аугментація текстових даних за допомогою методу зворотного перекладу (Back Translation) є ефективною технікою, що дозволяє значно покращити продуктивність моделей обробки природної мови (NLP) [20]. Ця техніка передбачає переклад тексту з початкової мови на цільову мову, а потім повернення його на початкову мову за допомогою іншої моделі перекладу. Цей процес створює паралельні корпуси, що допомагають моделі виявляти та усувати помилки, які могли виникнути під час початкового перекладу.

Основною перевагою методу зворотного перекладу є доволі значна поширеність та широке різноманіття моделей машинного перекладу. Крім того, цей метод дозволяє генерувати текст, який зберігає змістову сутність оригіналу, але містить різні мовні конструкції, що підвищує стійкість моделей до варіацій тексту. На рисунку 2.8 зображено загальну схему роботи методу зворотнього перекладу на прикладі англійської, турецької, датської та фінської мов.

Рисунок 2.8 – ілюстрація роботи Зворотнього перекладу

Проте, метод зворотного перекладу має й свої обмеження. Основною проблемою є можливість введення шуму під час перекладу, що може негативно вплинути на точність моделі. Крім того, для застосування цього методу необхідні високоякісні моделі перекладу, що можуть бути складними у розробці та налаштуванні. Іншою проблемою є те, що для

деяких мовних пар може бути складно знайти достатню кількість високоякісних монолінгвальних корпусів, що обмежує ефективність аугментації.

Цей метод забезпечує гнучкість і потужність у створенні тренувальних наборів даних, що дозволяє значно покращити якість і продуктивність моделей обробки природної мови.

2.4.3 Моделі трансформерів для аугментації текстових даних

BERT (Bidirectional Encoder Representations from Transformers) є моделлю для обробки природної мови, розробленою Google AI Language, яка значно покращила результати в багатьох задачах обробки тексту [21]. Основна ідея BERT полягає у використанні двонаправленого підходу для контекстного розуміння мови. Традиційні моделі, такі як OpenAI GPT, використовували однобічні (лівонаправлені) моделі, що обмежувало їх здатність повноцінно розуміти контекст. BERT вирішує цю проблему, навчаючи глибокі двонаправлені представлення, які враховують контекст як зліва, так і справа від кожного слова у всіх шарах моделі.

Однією з ключових характеристик BERT є використання завдання «маскованого мовного моделювання» (Masked Language Model, MLM). У цьому підході деякі слова у вхідному тексті випадково замінюються масками, і модель навчається передбачати оригінальні слова на основі їхнього контексту. Це дозволяє моделі отримувати більш повні представлення, що враховують як лівий, так і правий контекст. Додатково BERT використовує завдання передбачення наступного речення (Next Sentence Prediction, NSP), що допомагає моделі краще розуміти відносини між парами речень. На рисунку 2.9 наведено загальну схему навчання та роботи моделей BERT.

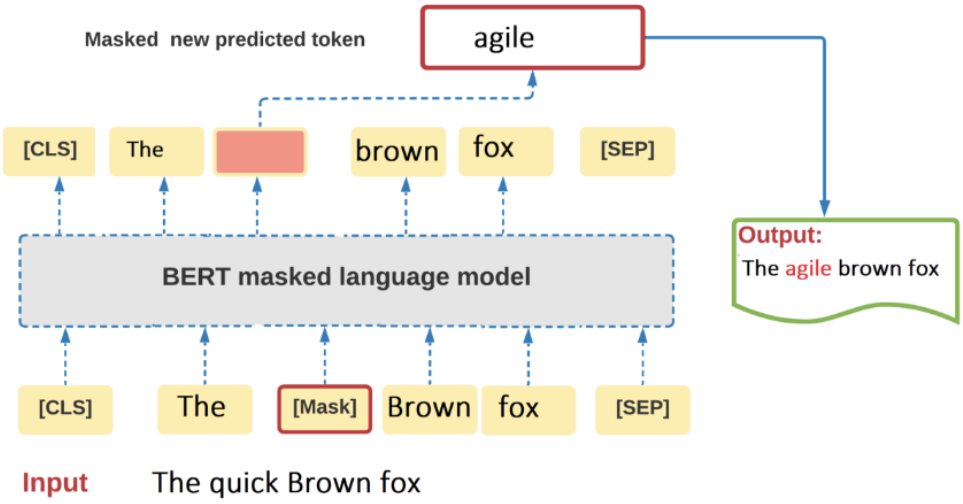


Рисунок 2.9 – ілюстрація принципу навчання та роботи BERT

BERT має значну перевагу в порівнянні з іншими моделями, оскільки він демонструє чудові результати в широкому спектрі завдань, таких як запитання-відповіді, розпізнавання іменованих сутностей та аналіз настроїв. Висока точність BERT у цих завданнях показала важливість двонаправленого навчання для представлення мовних моделей. Однак, BERT також має свої недоліки, серед яких значні вимоги до обчислювальних ресурсів та часу на тренування, що робить його менш доступним для використання у менш потужних системах.

Основні відмінності між RoBERTa та BERT полягають у різних підходах до попереднього навчання та обробки вхідних даних. RoBERTa (Robustly Optimized BERT Approach) є вдосконаленою версією BERT, яка використовує кілька ключових змін для покращення продуктивності [22]. По-перше, RoBERTa тренується довше, з більшими пакетами даних і на більших обсягах даних, що дозволяє моделі краще вивчити контекст та зменшити похибки. По-друге, RoBERTa відмовляється від задачі передбачення наступного речення (Next Sentence Prediction), яка використовується в BERT, оскільки дослідження показали, що ця задача незначно впливає на покращення продуктивності.

Ще однією важливою зміною є динамічне маскування, що застосовується до тренувальних даних у RoBERTa, на відміну від статичного маскування в BERT. Це означає, що кожного разу, коли послідовність слів подається до моделі, маски генеруються заново, що покращує навчання моделі. Крім того, RoBERTa використовує довші послідовності під час тренування, що дозволяє моделі краще розуміти зв'язки на довгих діапазонах. Ці зміни в навчальних процедурах та підходах дозволили RoBERTa досягти кращих результатів у різних завданнях обробки природної мови порівняно з оригінальною моделлю BERT.

2.4.4 Великі мовні моделі для аугментації текстових даних

Великі мовні моделі (LLM), такі як GPT-3, LLAMA та інші, продемонстрували значний потенціал у виконанні задач аугментації текстових даних. Вони здатні генерувати семантично збережені трансформації тексту на основі інструкцій, що дозволяє створювати різноманітні та інформативні варіанти оригінальних текстів.

Аугментація тексту з використанням LLM передбачає застосування моделей для генерації нових текстових зразків на основі наявних даних. Спочатку формулюються інструкції, які описують, як трансформувати текст (наприклад, заміна синонімами, зміна стилю, перефразування). Потім модель використовує ці інструкції для створення нових текстових зразків, таких як заміна певних слів на синоніми або перефразування речень. Згенеровані тексти перевіряються на відповідність заданим критеріям, таким як збереження семантики та релевантність до початкового тексту.

Переваги включають підвищення якості та різноманітності даних, оскільки LLM можуть генерувати тексти, які зберігають початкову семантику, але містять різноманітні формулювання. Це покращує контекстуальну сумісність текстів із завданнями та автоматизує процес підготовки даних, знижуючи потребу у ручному створенні варіантів тексту.

Однак, існують і недоліки. LLM потребують значних обчислювальних ресурсів для тренування та генерації текстів, що може бути проблемою для деяких застосувань. Процес тренування моделей може бути тривалим, що впливає на швидкість отримання результатів. Якість згенерованих текстів сильно залежить від якості та чіткості інструкцій: неправильно сформульовані інструкції можуть призвести до генерації нерелевантних або некоректних текстів. Хоча моделі добре працюють з контекстом, вони можуть стикатися з труднощами при обробці дуже специфічних або складних контекстів, що може вплинути на якість згенерованих даних.

Аугментація тексту за допомогою великих мовних моделей є потужним інструментом для покращення якості та різноманітності тренувальних даних. Вона дозволяє автоматизувати процес підготовки даних, забезпечуючи високу контекстуальну сумісність та семантичну збереженість. Однак, використання цього методу аугментації даних (DA) пов'язане з певними технічними викликами, такими як потреба у високих обчислювальних ресурсах та залежність від якості інструкцій.

2.5 Методологія дослідження

Для виявлення найбільш ефективного методу аугментації текстових даних найкращим варіантом є застосування відповідних методів в рамках незмінного пайплайну обробки даних. Саме такий підхід, який має лише одну змінну дозволить адекватно оцінити ефект методів аугментації які ми плануємо порівняти.

В рамках вирішення задачі навчання з учителем, одним з найкращих варіантів оцінки якості моделі, є використання тестового набору даних. Використання тестового набору даних для оцінювання продуктивності моделі є критично важливим аспектом у процесі розробки моделей машинного навчання [23]. Цей підхід дозволяє перевірити здатність моделі генералізувати знання на нові, невідомі раніше дані, які не

використовувались під час тренування. Основна мета тестового набору - забезпечити об'єктивний, незалежний спосіб оцінювання моделі.

Перш за все, тестовий набір допомагає виявити перенавчання, яке відбувається, коли модель занадто добре адаптується до тренувальних даних, втрачаючи здатність до адекватної генералізації. Якщо продуктивність моделі висока на тренувальному наборі, але значно падає на тестовому наборі, це є ознакою перенавчання. Таким чином, тестування дає змогу оцінити реальну ефективність моделі перед її застосуванням у реальному світі.

Наостанок, тестовий набір слугує для валідації кінцевої продуктивності моделі на даних, що відображають реальні умови її застосування. Це критично важливо для забезпечення відповідності моделі вимогам та очікуванням кінцевих користувачів. Такий підхід допомагає уникнути можливих збитків або недоліків, що можуть виникнути через неефективність моделі у практичному застосуванні, тим самим забезпечуючи більшу довіру та надійність у рішеннях, заснованих на аналізі даних.

Також, варто зазначити що, використовуючи різні методи аугментації в різних експериментах, ми маємо переконатись, що тестовий набір даних є незмінним між експериментами. Це допоможе уникнути впливу невизначеності та нестабільності, до яких могла б призвести зміна тестового набору даних.

2.6 Загальний опис експерименту

Як було зазначено раніше, експериментальну частину дослідження варто побудувати як порівнняння класифікаторів, побудованих з використанням різних методів аугментації текстових даних. При цьому тестовий набір даних має залишитись незмінним для тестування разних класифікаторів.

Набір даних, який ми використаємо, має обов'язково містити в собі текстову частину та мітку класу. Також бажано щоб записів було достатньо багато щоб ми мали можливість побачити покращення в цільовій метриці, навіть якщо воно менше 0.1%. Якщо припустити що ми маємо в тестовому наборі даних 500 записів, то в такому випадку найменша зміна, яка реально відобразиться на цільовій метриці дорівнює 0.2%. Виходить, що для виконання цієї вимоги тестовий набір повинен мати більше 1000 записів. При цьому тренувальний набір також повинен мати достатній розмір, чим більше тим краще.

Загальна схема експерименту зображена на рисунку 2.10. На ній можна побачити що аугментація даних відбувається до будь-якої обробки тренувального набору даних. При цьому варто зауважити що тестовий набір даних ні в якому разі не має використовуватись для генерації синтетичних даних, адже наявність сигналів з тестовогу набору при навчанні може зробити наш експеримент некоректним. Цей пайплайн є абсолютно ідентичним для кожного з експериментів.

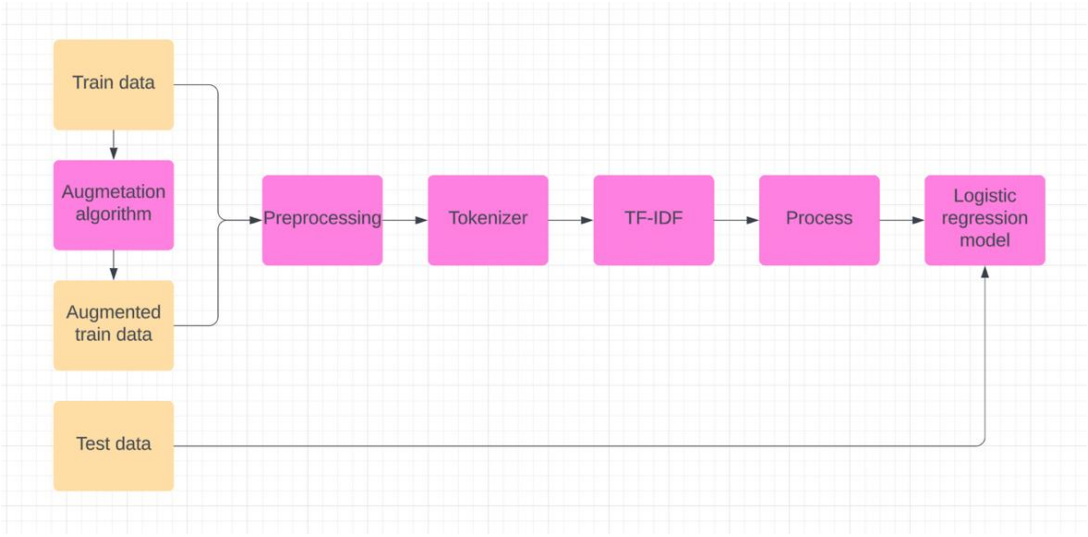


Рисунок 2.10 – Загальна схема пайплайну обробки даних та навчання моделей МН

2.7 Постановка задачі

Виходячи з усього вище зазначеного, можна сказати що аугментація текстових даних – один з найбільш актуальних та перспективних підходів до покращення моделей МН які працюють з текстом. Сучасні методи, маючи свої певні недоліки, здатні генерувати якісні синтетичні приклади, що у свою чергу підвищує різноманітність дата сету.

Основною проблемою яка існує на даний момент – вибір найбільш підходящого методу аугментації з врахуванням обмежень в обчислювальних ресурсах та з потребами конкретної задачі.

Оскільки мета цієї роботи – виявлення найбільш оптимального методу аугментації текстових даних, було прийнято рішення розглянути сучасні підходи DA та перевірити їх ефективність в контексті задачі класифікації.

Для досягнення поставленої мети необхідно опрацювати наступні питання:

– обрати підходящий для імітаційного моделювання набір даних;

– згенерувати синтетичні дані, використовуючи описані раніше методи аугментації тексту;

– навчити моделі, використавши, по черзі кожен з синтетичних наборів даних у комбінації з початковим тренувальним дата сетом;

– протестувати кожну з моделей, використавши тестову частину дата сету і проаналізувати результати.

Таким чином, основним завданням магістерської кваліфікаційної роботи є дослідження, експериментальне тестування та аналіз ефективності популярних методів аугментації тексту в контексті задачі класифікації текстових даних.

3 ОГЛЯД МЕТОДІВ ТА ЗАСОБІВ РЕАЛІЗАЦІЇ ДОСЛІДЖЕННЯ

3.1 Мова програмування Python

Python є домінуючим мовою програмування для досліджень у галузі машинного навчання (ML) та обробки природної мови (NLP) завдяки широкому спектру можливостей для маніпуляції з даними та моделювання. Обширна колекція бібліотек Python, таких як NumPy, pandas, scikit-learn, а також спеціалізовані бібліотеки NLTK і spaCy для NLP, забезпечує дослідникам комплексні інструменти, що постійно оновлюються новітніми алгоритмами.

Крім того, інтуїтивно зрозуміла і чиста синтаксична структура Python сприяє зниженню помилок у коді та полегшує експериментальний етап наукових проєктів у галузі NLP, дозволяючи швидко тестувати та удосконалювати гіпотези. Це значно пришвидшує процеси дослідження.

Нарешті, здатність Python інтегруватися з іншими технологіями та його широке використання в різних галузях роблять його ідеальним для міждисциплінарних досліджень. Ця інтероперабельність є ключовою при роботі з великими наборами даних або інтеграції моделей ML у більші програмні системи, що зміцнює позиції Python як фундаментального інструмента у дослідженнях NLP.

3.2 Бібліотеки та фреймворки для обробки текстових даних

3.2.1 Фреймворк NLTK

NLTK [24] є провідним фреймворком для роботи з природньою мовою. Він має зручний інтерфейс для доступу до понад 50 корпусів і лексичних ресурсів, таких як WordNet, а також комплект бібліотек обробки тексту для класифікації, токенізації, стемінгу, міткування, синтаксичного та

семантичного аналізу, обгортки для потужних бібліотек NLP, та активний форум для обговорень.

Завдяки практичному посібнику, який вводить основи програмування поряд із темами комп'ютерної лінгвістики і всеосяжною документацією API, NLTK підходить лінгвістам, інженерам, студентам, освітянам, дослідникам і користувачам з промисловості. NLTK доступний для Windows, Mac OS X та Linux і є безкоштовним, відкритим джерелом, керованим спільнотою проектом.

Серед важливих інструментів які надає NLTK варто назвати:

– word_tokenize. Дана функція дозволяє розбити текст на окремі слова, враховуючи пунктуацію та інші символи. Зазвичай вона використовується для попередньої обробки тексту перед аналізом або побудовою моделей;

– sent_tokenize. Дана функція розбиває текст на окремі речення. Цей інструмент знаходить широке застосування для аналізу структури тексту та визначення меж речень;

– pos_tag. Ця функція додає теги частин мови до кожного слова в тексті. Використовується для граматичного аналізу тексту та інших задач обробки природньої мови, які потребують визначення частин мови;

– ne_chunk. Виконує розпізнавання іменованих сутностей (імен, місць, організацій) в тексті. Допомагає витягати конкретні сутності з тексту для подальшого аналізу або обробки;

– PorterStemmer. Зводить слово до його кореневої форми. Використовується для зменшення варіативності слів та підвищення точності пошуку та класифікації текстів;

– WordNetLemmatizer. Приводить слово до його базової форми з урахуванням частини мови. Використовується для нормалізації тексту, зменшення варіативності слів;

– FreqDist. Обчислює частотний розподіл елементів в наборі даних. Використовується для аналізу частоти появи слів або інших елементів в тексті;

– ConcordanceIndex. Створює індекс з тексту для швидкого пошуку входжень конкретних слів або фраз. Використовується для пошуку контексту використання слів, аналізу їх розподілу в тексті.

3.2.2 Бібліотека дл роботи з текстовими даними SpaCy

SpaCy є безкоштовною, відкритою бібліотекою для передової обробки природної мови (NLP) на Python [25]. У випадках коли задача пов’язана з великими обсягами тексту, згодом може виникнути потреба детально його проаналізувати. Наприклад, зрозуміти, про що йдеться в тексті, що означають слова в контексті, хто що робить і кому, які компанії та продукти згадуються, які тексти схожі між собою.

SpaCy розроблено спеціально для використання у виробництві та допомагає створювати програми, які обробляють та «розуміють» великі обсяги тексту. Її можна використовувати для побудови систем вилучення інформації або розуміння природної мови, а також для попередньої обробки тексту для глибинного навчання.

3.3 Фреймворки Scikit-learn та PyTorch

Відкриті фреймворки машинного навчання (ML) суттєво сприяли розвитку Data Science, надаючи дослідникам та розробникам потужні інструменти для створення, навчання та розгортання моделей ML. Фреймворки такі як TensorFlow, PyTorch, Scikit-Learn та Keras дозволяють використовувати передові алгоритми машинного навчання та глибинного навчання, забезпечуючи при цьому високу продуктивність та масштабованість. Завдяки відкритому вихідному коду, ці фреймворки підтримують активну спільноту розробників, що сприяє швидкому впровадженню нових методів та покращень.

Роль відкритих фреймворків ML у розвитку Data Science не можна переоцінити. Вони роблять передові технології доступними для широкого кола користувачів, від академічних дослідників до інженерів у промисловості, прискорюючи інновації та впровадження машинного навчання у різних галузях. Відкриті фреймворки також сприяють співпраці та обміну знаннями, що дозволяє створювати більш точні, надійні та ефективні моделі, які можуть розв'язувати складні задачі аналізу даних і штучного інтелекту.

3.3.1 Scikit-Learn

Scikit-Learn є однією з найбільш популярних бібліотек для машинного навчання в Python, відомою своєю простотою у використанні та широкими можливостями [26]. Хоча Scikit-Learn не спеціалізується виключно на задачах обробки природної мови (NLP), вона надає потужні інструменти для вирішення багатьох задач у цій галузі. Бібліотека містить реалізації різних алгоритмів машинного навчання, таких як класифікація, регресія, кластеризація та зниження розмірності, які можуть бути ефективно застосовані до текстових даних.

Scikit-Learn пропонує зручні інструменти для попередньої обробки текстових даних, такі як `CountVectorizer` та `TfidfVectorizer`, які перетворюють текстові дані у числові вектори, що можуть бути використані в алгоритмах машинного навчання. Крім того, бібліотека надає модулі для моделей на основі наївного Байєса, логістичної регресії та підтримки векторних машин (SVM), які є основними інструментами для задач класифікації тексту. Завдяки своїй інтеграції з іншими бібліотеками Python, такими як Pandas та Numpy, Scikit-Learn забезпечує гнучкість та ефективність у створенні потужних NLP рішень. Окрім цього, Scikit-Learn пропонує великий набір інструментів для МН.

3.3.2 PyTorch

PyTorch є потужним відкритим фреймворком для машинного навчання, розробленим компанією Facebook AI Research (FAIR) [27]. Цей фреймворк відзначається своєю гнучкістю та динамічною обчислювальною графікою, що дозволяє дослідникам легко змінювати та налагоджувати моделі під час виконання. PyTorch широко використовується як у наукових дослідженнях, так і в промислових застосуваннях завдяки здатності обробляти великі обсяги даних і забезпечувати високу продуктивність обчислень. Підтримка різноманітних алгоритмів глибинного навчання, зокрема згорткових та рекурентних нейронних мереж, робить PyTorch ефективним інструментом для вирішення складних задач у галузі штучного інтелекту.

У контексті задач обробки природної мови (NLP), PyTorch надає потужні інструменти та бібліотеки для створення та навчання моделей для аналізу тексту. Бібліотека `torchtext` спрощує підготовку текстових даних, їх векторизацію та побудову пакетів для навчання моделей. PyTorch також підтримує інтеграцію з бібліотеками трансформерів, такими як Hugging Face Transformers, що надає доступ до передових моделей для різноманітних задач NLP, включаючи розпізнавання іменованих сутностей, аналіз тональності та автоматичне перекладання текстів. Гнучкість та висока продуктивність PyTorch дозволяють створювати інноваційні рішення для обробки природної мови, забезпечуючи точність і ефективність у вирішенні найскладніших завдань.

4 ІМІТАЦІЙНЕ МОДЕЛЮВАННЯ

Імітаційне моделювання є критичним етапом дослідження, оскільки дозволяє перевірити ефективність запропонованих методів аугментації текстових даних в контрольованому середовищі. Основна мета експериментів полягає у вивченні впливу різних методів аугментації на якість та продуктивність моделей машинного навчання. Це включає оцінку точності, узгодженості та загальної ефективності моделей при застосуванні різних технік аугментації.

Експерименти спрямовані на створення умов, максимально наближених до реальних, щоб результати могли бути легко екстрапольовані на практичні застосування. Це передбачає використання різноманітних дата сетів, які відображають різні аспекти завдань обробки природної мови, таких як класифікація текстів, розпізнавання сутностей, та машинний переклад. У процесі моделювання враховуються різні параметри та гіперпараметри, що дозволяє всебічно оцінити переваги та недоліки кожного методу аугментації.

Таким чином, імітаційне моделювання забезпечує глибоке розуміння впливу методів аугментації на продуктивність моделей та допомагає визначити оптимальні стратегії для покращення якості текстових даних. Це важливий крок до створення більш надійних та ефективних систем обробки природної мови, які здатні працювати з різноманітними та нерівномірно розподіленими даними.

4.1 Опис використаного дата сету та визначення цільової метрики

В якості предметної області було обрано детекцію пропаганди та хибних новин (Fake News detection). Задача виявлення фейкових новин набуває все більшої актуальності в сучасному інформаційному просторі, оскільки швидке поширення неправдивої інформації може мати серйозні

наслідки для суспільства. Особливо важливою дана задача стала з початком війни росії проти України в 2014 році, адже інформаційна війна є частиною гібридної стратегії агресора. З розвитком соціальних мереж та цифрових медіа, фейкові новини можуть швидко досягати широкої аудиторії, викликаючи дезінформацію, соціальні паніки, політичні маніпуляції та підрив довіри до легітимних джерел інформації. Це ставить під загрозу демократичні процеси, здоров'я населення та суспільну стабільність.

Технологічні рішення для виявлення фейкових новин є критично важливими для захисту інформаційного середовища. Використання методів машинного навчання та обробки природної мови дозволяє автоматизувати процес ідентифікації фейкових новин, підвищуючи точність та швидкість виявлення. Такі системи можуть аналізувати великі обсяги даних, виявляючи патерни, характерні для неправдивих повідомлень, що сприяє зниженню впливу фейкової інформації та підтримці інформаційної безпеки.

В якості дата сету для експерименту було обрано набір даних ISOT Fake News. Він є комбінацією тисяч правдивих та фейкових статей. Правдивість статей в даному випадку визначається їх походженням, а саме тим, з якого електронного видання вони походять. Надійність ресурсів було визначено спеціалізованою агенцією Politifact.com, що займається перевіркою новин на правдивість.

Набір даних містить два типи статей: фейкові та справжні новини. Цей набір даних було зібрано з реальних джерел; правдиві статті були отримані шляхом збору матеріалів з сайту Reuters.com (новинний сайт). Щодо фейкових новин, вони були зібрані з різних джерел. Фейкові новини були зібрані з ненадійних вебсайтів, які були відзначені Politifact (організацією з перевірки фактів у США) та Вікіпедією.

Набір даних складається з двох файлів CSV. Перший файл під назвою «True.csv» містить понад 12,600 статей з reuters.com. Другий файл під назвою «Fake.csv» містить понад 12,600 статей з різних ненадійних ресурсів фейкових новин. Кожна стаття містить наступну інформацію: заголовок

статті, текст, тип та дата публікації статті. Для узгодження з даними фейкових новин, зібраними для kaggle.com, ми зосередилися переважно на зборі статей з 2016 по 2017 роки. Зібрані дані були очищені та оброблені, проте пунктуація та помилки, які існували у фейкових новинах, були збережені в тексті. Таблиця 4.1 зображує розподіл категорій та кількість статей в кожній категорії.

Таблиця 4.1 – Характеристики набору даних ISOT Fake News

Правдивість	Розмір (к-сть статей)	Тема	
		Тип	Розмір (к-сть статей)
Правдиві новини	21417	Світові новини	10145
		Політичні новини	11272
Фейкові новини	23481	Державні новини	1570
		Новини Близького Сходу	778
		Новини США	783
		Новини «лівого» спектра	4459
		Політика	6841
		Новини	9050

Враховуючи те що даний набір даних є збалансованим по кількості записів в класах «Правдиві новини» та «Фейкові новини», в якості цільової метрики обрано точність (Accuracy), адже вона відображає загальну якість класифікації, при цьому, не віддаючи перевагу жодному з наявних в дата сеті класів.

4.2 Застосування методів аугментації та підготовка даних

Згідно з раніше визначеним дизайном даного дослідження, перший етап експерименту – генерація синтетичних даних для збільшення вибірки. Для цього спершу відділити тестову частину дата сету від тренувальних даних. Для цього використаємо рандомізований поділ, реалізований в Scikit-Learn у вигляді функції train_test_split. Тестова вибірка – частина даних, яка не зовсім не використовується для тренування моделі.

Далі необіхдно згенерувати синтетичні дані. Для створення нових записів необхідно використати вже наявні дані, певну їх підмножину. Тому наступний крок – створити семпл з тренувального набору даних. Семпл в даному випадку матиме розмір 33% від загального розміру тренувальної вибірки. Отриманий семпл надалі було використано для отримання синтетичних даних, використовуючи наступні підходи:

– відсутність аугментації (контрольний дата сет);

– випадкова заміна синонімами;

– випадкове видалення слів;

– випадкова вставка слів зі словника;

– зворотній переклад;

– аугментація з застосуванням моделі BERT;

– аугментація з застосуванням моделі RoBERTa;

– аугментація з застосуванням GPT-2.

Для того щоб продемонструвати роботу кожного з методів аугментації в наступних розділах буде використано речення «Proactive president Trump Just Took huge step to make america safe…while democrats are determined to make us more like France, UK». Це речення є заголовком фейкової новини і має клас «Fake». Воно містить абревіатури, розділові знаки та є доволі змістовним в плані наповнення.

4.2.1 Контрольний дата сет та застосування простих аугментацій

Для перевірки справжнього ефекту аугментацій та порівняння підходів між собою необхідно, додавши до тренувального набору синтетичні дані, навчити окремі моделі для кожної аугментованої вибірки та протестувати перфоманс моделей на тестовому наборі даних без аугментацій.

Для перевірки простих аугментацій необхідно послідовно, використовуючи кожен метод аугментації згенерувати синтетичні записи. В таблиці 4.2 наведено одне із речень з набору даних та те що вийшло після послідовного застосування відповідних методів аугментації.

Таблиця 4.2 – Ілюстрація простих методів аугментації текстових даних що розглядаються в дослідженні.

Метод	Речення
Початкове речення	proactive president trump Just Took huge step to make america safe…while democrats are determined to make us more like france, uk
Випадкова заміна синонімами	proactive chairman trump just took huge step to piss america safe…while democrats are determined to piss us more like france, uk'
Випадкові перестановки	proactive president trump just are huge step to make america safe…while determined took democrats to make us more like france, uk
Випадкове видалення	proactive president just took step to make america are determined us More like france, uk

4.2.2 Застосування зворотнього перекладу, трансформерів та великих мовних моделей

Використання Back Translation та LLM дозволяє отримувати більш різноманітні варіанти текстів. LLM можуть створювати тексти з різними стилістичними та синтаксичними варіаціями, що значно підвищує різноманітність аугментованих даних. Це особливо корисно для моделей, які потребують великої кількості даних з різними варіаціями для підвищення їх стійкості до шумів та покращення узагальнення. Таблиця 4.3 ілюструє результати роботи ряду підходів з використанням моделей МН для аугментації даних.

Таблиця 4.3 – Ілюстрація роботи методів аугментації текстових даних з використанням Великих Мовних Моделей.

Метод	Речення
Початкове речення	PROACTIVE PRESIDENT TRUMP Just Took Huge Step To Make America Safe…While Democrats Are Determined To Make Us More Like France, UK
Back translation	PROACTIVE PRESIDENT TRUMP has just taken a big step to make America safe... While Democrats are determined to make us more like France, Britain
BERT	proactive president trump soon took huge step to proving china safe while democrats stopped proposing that make us more accessible france ,

Продовження таблиці 4.3.

RoBERTa	PROACTIVE PRESIDENT Putin Saw A Huge Action To Make Us Safe…While Journalists Were Dying To Make Us More Like Reading, UK
GPT-2	PROACTIVE PRESIDENT TRUMP Just Took Huge Step To Make America Safe…While Democrats Are Determined To Make Us More Like France, UK, China And Russia, It's Time To Move Forward.\n\nAnd what do you think?\n\nComments\n\ncomments

Навіть на прикладі одного речення можна побачити наскільки більш ефективними є великі мовні моделі. Вони розширюють словник доступних слів і роблять це з допомогою слів, наближених до теми обговорення.

4.2.3 Подальша підготовка даних

Після отримання нових тренувальних наборів даних, до кожного з них було застосовано наступні кроки:

1) переведення до нижнього регістру;
2) видалення стоп-слів;
3) токенізація;
4) лематизація токенів;
5) перетворення до векторного представлення з допомогою TF-IDF.

Для видалення стоп-слів, токенізації, лематизації було використано відповідний функціонал з бібліотеки NLTK. Таблиця 4.4 ілюструє послідовне застосування кожного з етапів препроцесінгу. Для приведення

тексту у векторний формат було використано модель TF-IDF. Цей крок в таблицю не було додано через неінформативність доступних візуальних представлень.

Таблиця 4.4 – Приклади роботи пайплайну попередньої підготовки даних реалізованого в експериментальній частині дослідження.

Метод	Речення
Речення отримане після Back Translation	PROACTIVE PRESIDENT TRUMP has just taken a big step to make America safe... While Democrats are determined to make us more like France, Britain
Переведення до нижнього регістру	proactive president trump has just taken a big step to make america safe... while democrats are determined to make us more like france, britain
Видалення стоп-слів	proactive president trump just taken a big step make america safe... while democrats determined make us more like france, britain
Токенізація	«proactive» «president» «trump» «just» «taken» «a» «big» «step» «make» «america» «safe» «while» «democrats» «determined» «make» «us» «more» «like» «france» «britain»
Лематизація токенів	proactive president trump just taken a big step make america safe while democrat determine make us more like france britain

4.3 Навчання моделі машинного навчання

Наступний крок після отримання для кожного набору даних векторного представлення текстів це навчання моделі. В якості класифікатора в даній задачі детекції фейків використаємо лінійну модель, що використовує L2 регуляризатор, RidgeClassifier.

4.4 Аналіз результатів

Експеримент мав на меті дослідити вплив різних технік аугментації текстових даних на точність (Accuracy) та повноту (Recall) класифікаційної моделі. Нижче наведено аналіз отриманих результатів. В таблиці 4.5 наведено результати кожного з підходів.

Таблиця 4.5 – Показники продуктивності моделей навчених із додаванням синтетичних даних отриманих різними методами.

Метод	Accuracy	Recall
Без аугментації	0.848	0.802
Випадкова заміна синонімами	0.71	0.73
Випадкові перестановки	0.848	0.802
Випадкове видалення	0.807	0.781
Back translation	0.858	0.83
BERT	0.852	0.838
RoBERTa	0.863	0.84
GPT-2	0.86	0.838

Техніки простої аугментації (випадкова заміна синонімами, випадкові перестановки, випадкове видалення) вплинули на перфоманс моделі доволі негативно, що відобразилося на значеннях цільових метрик.

Back translation показала покращення обох показників порівняно з базовим рівнем. Це свідчить про те, що ця техніка допомагає зберігати семантичний зміст тексту, при цьому додаючи варіативності, що покращує навчання моделі.

Використання BERT для аугментації призвело до покращення обох показників, особливо Recall. Це свідчить про те, що BERT добре справляється з додаванням варіативності до тексту, зберігаючи його зміст.

RoBERTa показала найкращі результати серед усіх технік, покращивши як точність, так і повноту. Це може бути пов'язано з покращеними архітектурними рішеннями моделі RoBERTa порівняно з іншими техніками.

GPT-2 також показала високі результати, покращуючи обидва показники. Це свідчить про те, що GPT-2 ефективно генерує варіативні тексти, які добре доповнюють навчальні дані.

ВИСНОВКИ

Резюмуючи, можна зазначити що аугментація текстових даних є підходом, що, не зважаючи на наявні дослідження, є перспективним та недостатньо дослідженим. Сучасні методи, маючи свої певні недоліки, здатні генерувати якісні синтетичні приклади, що у свою чергу підвищує різноманітність дата сету.

Як було зазначено вище, основною проблемою на даний момент є вибір найбільш підходящого методу аугментації з врахуванням обмежень в обчислювальних ресурсах та з потребами конкретної задачі.

Було проаналізовано актуальний стан проблеми, а також проведено ознайомлення з найбільш актуальними методами аугментації текстових даних. В результаті було визначено ряд підходів, що представляють для даного дослідження найбільший інтерес.

Для проведення експериментального дослідження було визначено принципи дизайну експерименту, цільові показники та набір даних. Згідно з постановкою задачі, було проведено імітаційне дослідження, що розглядає ефективність методів аугментації текстових даних в контексті задачі класифікації тексту.

Наступним кроком було проведено аналіз результатів імітаційного моделювання і визначено підходи, що зарекомендували себе найкраще в рамках даного експерименту. Також, було проаналізовано можливі чинники, що могли вплинути на результати експерименту та визначено ряд питань які потребують дослідження у майбутньому.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1) Structured vs. Unstructured Data Types Explained. *Oracle | Cloud Applications and Cloud Platform*. URL: https://www.oracle.com/big-data/structured-vs-unstructured-data/ (дата звернення: 19.04.2024).

2) Géron A. Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems. O'Reilly Media, Incorporated, 2022.

3) Text Classification Algorithms: A Survey / Kowsari et al. *Information*. 2019. Vol. 10, no. 4. P. 150. URL: https://doi.org/10.3390/info10040150 (дата звернення: 31.03.2024).

4) Qaiser S., Ali R. Text Mining: Use of TF-IDF to Examine the Relevance of Words to Documents. *International Journal of Computer Applications*. 2018. Vol. 181, no. 1. P. 25–29. URL: https://doi.org/10.5120/ijca2018917395 (дата звернення: 31.03.2024).

5) Shorten C., Khoshgoftaar T. M., Furht B. Text Data Augmentation for Deep Learning. *Journal of Big Data*. 2021. Vol. 8, no. 1. URL: https://doi.org/10.1186/s40537-021-00492-0 (дата звернення: 31.03.2024).

6) Wei J., Zou K. EDA: Easy Data Augmentation Techniques for Boosting Performance on Text Classification Tasks. *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, Hong Kong, China. Stroudsburg, PA, USA, 2019. URL: https://doi.org/10.18653/v1/d19-1670 (дата звернення: 31.03.2024).

7) Text Data Augmentation for the Korean Language / D. T. Vu et al. *Applied Sciences*. 2022. Vol. 12, no. 7. P. 3425. URL: https://doi.org/10.3390/app12073425 (дата звернення: 31.03.2024).

8) Large, Small or Both: A Novel Data Augmentation Framework Based on Language Models for Debiasing Opinion Summarization / Y. Zhang та

ін. *ArXiv*. 2024. URL: https://doi.org/10.48550/arXiv.2403.07693 (дата звернення: 31.03.2024).

9) Large, Small or Both: A Novel Data Augmentation Framework Based on Language Models for Debiasing Opinion Summarization / Y. Zhang та ін. *ArXiv*. 2024. URL: https://doi.org/10.48550/arXiv.2403.07693 (дата звернення: 31.03.2024).

10) What Are Naïve Bayes Classifiers? | IBM. *IBM in Deutschland, Österreich und der Schweiz*. URL: https://www.ibm.com/topics/naive-bayes (дата звернення: 28.02.2024).

11) A Survey on Text Classification: From Traditional to Deep Learning / Q. Li та ін. *ACM Transactions on Intelligent Systems and Technology*. 2022. Т. 13, № 2. С. 1–41. URL: https://doi.org/10.1145/3495162 (дата звернення: 28.04.2024).

12) Mueller J. P., Massaron L. Data Science Programming All-In-One for Dummies. Wiley & Sons Canada, Limited, John, 2020. 608 с.

13) Distances Dissimilarity Measures for Interval Data. *IBM in Deutschland, Österreich und der Schweiz*. URL: https://www.ibm.com/docs/hu/spss-statistics/saas?topic=measures-distances-dissimilarity-interval-data (дата звернення: 10.05.2024).

14) Jijo B. T., Abdulazeez A. M. Classification Based on Decision Tree Algorithm for Machine Learning. *Journal of Applied Science and Technology Trends*. 2021. Т. 2.

15) Galke L., Scherp A. Bag-of-Words vs. Graph vs. Sequence in Text Classification: Questioning the Necessity of Text-Graphs and the Surprising Strength of a Wide MLP. *ArXiv*. 2021. URL: https://doi.org/10.48550/arXiv.2109.03777 (дата звернення: 08.04.2024).

16) A comparative review on deep learning models for text classification / M. Zulqarnain та ін. *Indonesian Journal of Electrical Engineering and Computer Science*. 2020. Т. 19, № 1. С. 325.

URL: https://doi.org/10.11591/ijeecs.v19.i1.pp325-335 (дата звернення: 08.05.2024).

17) Tabassum A., Patil D. R. R. A Survey on Text Pre-Processing & Feature Extraction Techniques in Natural Language Processing. *International Research Journal of Engineering and Technology (IRJET)*. 2020. С. 4864–4867.

18) Raschka S. An Overview of General Performance Metrics of Binary Classifier Systems. *ArXiv*. 2014. URL: https://arxiv.org/pdf/1410.5330 (дата звернення: 09.04.2024).

20) Wei J., Zou K. EDA: Easy Data Augmentation Techniques for Boosting Performance on Text Classification Tasks. *ArXiv*. 2019.

21) Sugiyama A., Yoshinaga N. Data Augmentation using Back-translation for Context-aware Neural Machine Translation. 2019. URL: https://aclanthology.org/D19-6504.pdf (дата звернення: 08.05.2024).

22) Distances Dissimilarity Measures for Interval Data. *IBM in Deutschland, Österreich und der Schweiz*. URL: https://www.ibm.com/docs/hu/spss-statistics/saas?topic=measures-distances-dissimilarity-interval-data (дата звернення: 10.05.2024).

23) RoBERTa: A Robustly Optimized BERT Pretraining Approach / Y. Liu та ін. *ArXiv*. 2019. URL: https://arxiv.org/pdf/1907.11692 (дата звернення: 18.04.2024).

24) Ng A. Machine Learning Yearning. 2018.

25) NLTK :: Natural Language Toolkit. *NLTK :: Natural Language Toolkit*. URL: https://www.nltk.org/ (дата звернення: 28.02.2024).

26) Raschka S. An Overview of General Performance Metrics of Binary Classifier Systems. *ArXiv*. 2014. URL: https://arxiv.org/pdf/1410.5330 (дата звернення: 09.04.2024).

27) scikit-learn: machine learning in Python – scikit-learn 1.5.0 documentation. *scikit-learn: machine learning in Python – scikit-learn 0.16.1 documentation*. URL: https://scikit-learn.org/stable/ (дата звернення: 28.02.2024).

28) PyTorch. URL: https://pytorch.org/foundation (дата звернення: 28.04.2024).