Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Центр Післядипломної освіти
(повна назва)
Кафедра Штучного інтелекту
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

Дослідження методів нейромережевої класифікації новинних текстів

(тема)

Виконав:
студент 2 курсу, групи СШІмзд-22-1
Кіракосян Е.М.
(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 проф. Удовенко С.Г.
(посада, прізвище, ініціали)

Допускається до захисту

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

2024 р.

Харківський національний університет радіоелектроніки

Центр Післядипломної освіти
(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 Кіракосяну Едуарду Меружановичу
(прізвище, ім'я, по батькові)

1. Тема роботи Дослідження методів нейромережевої класифікації новинних текстів

затверджена наказом університету від 22 квітня 20 24 р. № 61Стз

2. Термін подання студентом роботи до екзаменаційної комісії 13 червня 20 24 р.

3. Вихідні дані до роботи Автоматичні методи розпізнавання фейкових новин, науково-технічні публікації, дані інтернет-джерел та відомих наукових проектів щодо виявлення фейкової складової у новинних текста, Python документація та набір данних з реальними, фейковими новинами

4. Перелік питань, що потрібно опрацювати в роботі
1) Аналіз предметної галузі та постановка задачі
2) Узагальнена модель нейромережевої класифікації новинних текстів
3) Програмна реалізація та результати моделювання бінарних та багатокласових класифікаторів новинних текстів

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Отримання завдання на кваліфікаційну роботу	22.04.2024	виконано
2	Аналіз предметної галузі	23.04.2024 – 27.04.2024	виконано
3	Огляд машинного навчання проти поширення	28.05.2024 – 02.05.2024	виконано
	фейкових новин		
4	Дослідження алгоритмів виявлення фейків	03.05.2024 – 08.05.2024	виконано
5	Розроблення моделей та підготовка до	09.05.2024 – 17.05.2024	виконано
	експериментів		
6	Реалізації експериментів проекту	18.05.2024 – 29.05.2024	виконано
7	Підведення підсумків експерименту	30.05.2024 – 31.05.2024	виконано
8	Оформлення пояснювальної записки	01.06.2024 – 05.06.2024	виконано
9	Оформлення презентації	06.06.2024 – 09.06.2024	виконано
10	Попередній захист	10.06.2024	виконано
11	Захист перед ЕК	13.06.2024	

Дата видачі завдання 22 квітня 20 24 р.

Студент _____ _____
(підпис)

Керівник роботи _____ проф. Удовенко С.Г.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка: 97 с., 29 рис., 2 дод., 32 джерела.

АНАЛІЗ НОВИННИХ ТЕКСТІВ, БАГАТОКЛАСОВА КЛАСИФІКАЦІЯ, БІНАРНА КЛАСИФІКАЦІЯ, ДАТАСЕТ НОВИННИХ ТЕКСТІВ, НЕЙРОМЕРЕЖЕВІ КЛАСИФІКАТОРИ ТЕКСТІВ, GOOGLE COLAB, KERAS, PYTHON.

Об’єкт дослідження – системи класифікації новинних текстів з використанням методів машинного навчання.

Предмет дослідження – методи побудови бінарних та багатокласових класифікаторів новинних текстів.

Мета роботи – дослідження методів нейромережевої класифікації новинних текстів.

Методи дослідження – існуючі моделі машинного навчання для попередньої обробки та класифікації новинних текстів; методи побудови, аналізу та навчання нейромережевих моделей класифікації текстів.

Практична значимість даної кваліфікаційної роботи полягає в розробці узагальненої моделі нейромережевої бінарної та багатокласової класифікації новинних текстів. Така модель може сприяти проведенню додаткових досліджень з розробки та аналізу ефективних класифікаторів текстових новин, виявленню фейкових новин та підвищенню якості класифікаторів новинних текстів. В кваліфікаційній роботі здійснено: аналіз існуючих типів новинних текстів та їх особливостей; аналіз методів попередньої обробки новинних текстів; програмну реалізацію та тестування розробленої узагальненої моделі на прикладах бінарної та багатокласової класифікації новинних текстів.

ABSTRACT

Master’s thesis contains: 97 pp., 29 fig., 2 ann., 32 references.

ANALYSIS OF NEWS TEXTS, BINARY CLASSIFICATION, GOOGLE COLAB, KERAS, MULTICLASS CLASSIFICATION, NEURONETWORK TEXT CLASSIFIERS, NEWS TEXT DATASET, PYTHON.

The object of the research is classification systems of news texts using machine learning methods.

The subject of the research is methods of building binary and multi-class classifiers of news texts.

The purpose of the work is to research methods of neural network classification of news texts.

Research methods – existing machine learning models for pre-processing and classification of news texts; methods of construction, analysis and training of neural network models of text classification.

The practical significance of this diploma project lies in the development of a generalized neural network model of binary and multi-class classification of news texts. Such a model can contribute to conducting additional research on the development and analysis of effective text news classifiers, the detection of fake news, and the improvement of the quality of news text classifiers. The qualification work carried out: analysis of existing types of news texts and their features; analysis of methods of preliminary processing of news texts; software implementation and testing of the developed generalized model on examples of binary and multi-class classification of news texts.

ЗМІСТ

ВСТУП

В області автоматичної обробки електронних текстів набув розвитку напрямок інтелектуального аналізу, в рамках якого здійснюється пошук нових корисних знань з неструктурованої текстової інформації. Під неструктурованими текстовими даними розуміють електронні документи будь-якого типу: Web-сторінки, електронну пошту, нормативні документи тощо [1]. Завдання організації ефективного доступу до неструктурованої тематичної інформації безпосередньо пов'язано із завданням класифікації електронних новинних текстів, які надходять здебільшого з ресурсів мережі Інтернет. На сьогодні розроблено чимало ефективних методів, деякі з яких характеризуються якістю класифікації, що можна навіть перевищувати якість класифікації текстів, що виконується кваліфікованими експертами.

Актуальність розробки таких методів визначається необхідністю обробки великих масивів текстової інформації, накопичених у глобальному інформаційному просторі. Обробка слабоструктурованої текстової інформації є нетривіальним завданням, що виходить за рамки традиційної алгоритмічної обробки структурованих даних. Щоб отримати з новинних текстів корисну інформацію, необхідно їх структурувати, впорядкувати, систематизувати та забезпечити ефективність пошуку текстів за запитами користувачів.

До основних завдань обробки текстової інформації можна віднести класифікацію текстів, їх попередню обробку, анотування текстових документів, аналіз емоційної складової текстів, порівняння тематичних текстів автоматичний переклад тощо. Слід відзначити, що всі ці завдання у певній мірі пов’язані саме з класифікацією тексту.

Використовуючи текстові класифікатори, можна автоматично структурувати усі різновиди текстових новинних документів.

Існує чимало напрямів практичного застосування методів обробки та класифікації новинних текстів з використанням засобів штучного інтелекту (зокрема, спроба відрізняти надійні новини від фейкових новин).

Доцільно розглянути важливі аспекти аналізу цієї проблеми, розробки перспективних методів обробки текстів та їх практичного застосування у деяких предметних галузях.

Для цього в пропонованому дослідженні вирішуються такі завдання:

– аналіз існуючих типів новинних текстів та їх особливостей;

– аналіз існуючих методів попередньої обробки новинних текстів;

– дослідження засобів класифікації новинних текстів з використанням машинного навчання;

– розроблення узагальненої моделі нейромережевої класифікації новинних текстів;

– програмна реалізація та тестування розробленої моделі на прикладах бінарної та багатокласової класифікації новинних текстів.

1 АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ ТА ПОСТАНОВКА ЗАДАЧІ

1.1 Особливості процесів комп’ютерної обробки текстової інформації

Процедура класифікації новинних текстів є частиною загальної проблеми обробки текстової інформації з використанням методів машинного навчання [2], [3]. В області автоматичної обробки електронних текстів набув розвитку напрямок інтелектуального аналізу даних Text Mining, в рамках якого здійснюється пошук нових корисних знань з неструктурованої текстової інформації. Під час вирішення задачі автоматичної класифікації текстів зазвичай використовується їх представлення у векторному вигляді. Існує чимало методів векторизації тексту (зокрема, Word2Vec, TF-IDF, Skip-gram, Continuous Bag of Words, MUSE тощо). Втім слід зазначити, що ці методи не дозволяють вирішити задачу багатомовної векторизації текстів, адже їх використання передбачає необхідність попереднього визначення мови, якою написано текст, перед опрацюванням тексту з застосуванням відповідною до мови моделі. Однак такий спосіб потребує написання багатої кількості коду для підтримки цієї логіки, а також наявності моделі для векторизації та для обробки векторів під кожну мову, з якою потрібно працювати.

Розповсюдженими прикладами електронних репозитаріїв для текстової інформації є електронні архіви, бібліотеки, системи управління документами та інші. Обробка та аналіз текстового контенту зазвичай виконувався вручну, але зі збільшенням обсягу інформації вони стали неефективними і виникала необхідність автоматизації цих процесів. Однією з актуальних задач аналізу текстових даних є класифікація – визначення тематики заданого тексту і віднесення його до однієї з наперед визначених категорій. Новинні тексти генеруються та зберігаються здебільшого глобальною мережею Інтернет.

Класифікація текстових даних передбачає визначення належності текстових даних будь-якій темі, якій присвячений текст. Тематична класифікація часто використовується для фільтрації текстових даних, коли необхідно виключити з розгляду записи, що відносяться до тем, які не цікавлять користувача [4], [5], [6].

Вирішення завдань автоматичної класифікації текстів в останні роки стало одним із пріоритетних напрямків розвитку досліджень в області інформаційного пошуку та штучного інтелекту. Засоби автоматичної класифікації текстів знаходять застосування не тільки при відборі найбільш релевантних результатів пошукових запитів, але і при вирішенні таких прикладних завдань, як фільтрація спаму, складання персональних добірок новин, виявлення фейкової складової в новинних текстах.

Якість класифікації текстової інформації багато в чому залежить від ефективної реалізації процесів її попередньої обробки, до яких, насамперед, слід віднести токенізацію, нормалізацію, стемінг та вбудовування слів.

Токенізація – це процес розбивання тексту на дрібніші фрагменти, що називаються токенами. Ці фрагменти можуть бути реченнями, словами або підсловами (залежно від задачі, що вирішується). Токенізація є важливим етапом попередньої обробки в більшості програм для обробки природномовних текстів.

Наприклад, щоб підрахувати кількість слів у тексті, текст розбивається на частини за допомогою токенізаторів. У глибокому навчанні та традиційних методах токенізація використовується для розробки ознак. Зокрема, вхідний текст перед подачею до нейромережевої архітектури BERT обробляється за допомогою токенізатора підслів WordPiece.

Типовим етапом після токенізації йде видалення стоп-слів, які не впливають суттєво на контекст або можуть бути занадто короткими.

Нормалізація – це процес перетворення токена в його базову форму. Нормалізація тексту дозволяє зменшити його випадковість, наближаючи його до стандартної форми, що підвищує ефективність подальшої

класифікації. Нормалізацію можна поділити на два основних підходи: стемінг і лематизацію. Ці методи ставлять перед собою задачу привести всі використанні в тексті словоформи до однієї, але використовують різні методи для цього. Наприклад, початковою формою прикметника є форма називного відмінка чоловічого роду однини, форма називного відмінка однини є початковою формою іменника.

Стеммінг – це процес зведення слів до їхньої основи або кореневої форми. Метою створення основ є скорочення споріднених слів до однієї основи, навіть якщо основа не є словом зі словника. Під стеммінгом зазвичай розуміється евристичний процес, в якому закінчення і словотворні суфікси відкидаються від слів в розрахунку на те, що в більшості випадків результат залишиться стебловим. Варто відзначити, що значення поняття «стемма» трактується по-різному. У обчислювальній лінгвістиці під стемою розуміють частину слова без словотворного суфікса (похідного суфікса) і закінчення (флективного суфікса).

Методи стеммінгу можна класифікувати на три групи: ті, що використовують відсікання афіксів, статистичні методи та змішані методи. Кожна з цих груп має свій підхід до знаходження основи слова.

Алгоритми першого типу відрізають префікси та суфікси слова, які є різновидами афікса. Найпростішим алгоритмом цього типу є Truncate (n) stemmer, який усікає слово, що починається з n-го символу. Ще однією простою реалізацією такого підходу є S-stemmer, який перетворює англійські слова з множини в однину, відсікаючи суфікси деякими правилами [7].

Статистичні алгоритми використовують лінгвістичний корпус, щоб прийти до набору правил. Правила будуються на основі розбору слів, які мають однакову стему. Стемінг не завжди є ефекивним методом нормалізації, оскільки іноді може призвести до виникнення так званих ситиуацій надмірного або недостатнього стемінгу.

Надмірний стемінг характеризується відрізанням набагато більшої частини слова, ніж потрібно, що у свою чергу призводить до того, що слова неправильно скорочуються до того самого кореневого слова. Наприклад, слова ‘university’ та ‘universe’, які скорочуються до ‘univers’.

Недостатній стемінг відбувається, коли два або більше слів можуть бути помилково скорочені до більш ніж одного кореневого слова, коли вони повинні бути скорочені до одного і того ж самого. Наприклад, слова «data» и «datum», які скорочуються до «dat» та «datu» відповідно (замість того самого кореня «dat»).

Лематизація, на відміну від стемінгу, скорочує слова до їхньої основи, правильно скорочуючи розділені слова та гарантуючи належність кореневого слова до мови. Зазвичай вона складніша, ніж стеммінг, оскільки лематизація працює над окремим словом без знання контексту. При лематизації кореневе слово називається лемою. Лема – це канонічна форма, словникова форма або форма цитування набору слів.

Для кожного слова визначається його лексична категорія в реченні, після чого застосовуються евристичні правила, що вживаються в стемінгу, але ці правила застосовуються відповідно до категорії. Таким чином, для пропозицій, де слово «біг» вживається як дієслово і як іменник, лема буде різною – «бігти» і «біг» відповідно. Такий підхід сильно залежить від точного визначення частини мови. Хоча існує часткове перекриття між правилами нормалізації для деяких лексичних категорій, зазначення неправильної категорії або нездатність визначити правильну категорію зводить нанівець перевагу цього підходу перед алгоритмом усічення.

Вбудовування слів (word embedding) є одним з найбільш важливих етапів конвертації слів в числові значення при обробці тексту. Вбудовування слів – це спосіб представлення документів у вигляді чисельного вектору. Важливим є те, що слова (речення), близькі по сенсу, повинні відображатися в схожі числові вектори. Існують готові до використання моделі вбудовування слів, такі як Word2Vec і GloVe.

Метод Word2Vec пропонує векторизацію текста на рівні ембедингів слів. Метод є досить популярним завдяки легкій інтеграції до основних фреймворків машинного навчання та наявності векторів ембедингів слів для багатьох мов. Word2Vec використовує неглибоку нейронну мережу, що навчена відновлювати лінгвістичний контекст слів. Вона приймає на вхід великий корпус слів і створює векторний простір. Цей векторний простір має декілька сотень вимірів, кожному слову в цьому вимірі призначено відповідний унікальний вектор. Вектори слів що надає даний метод розміщуються у векторному просторі таким чином, щоб слова, які мають схожий контекст, мали малу векторну відстань та були розміщенні поруч.

Word2Vec має дві алгоритмічно схожі моделі: Continuous-Bag-of-Word (CBOW) та Skip-gram (рисунок 1.1).

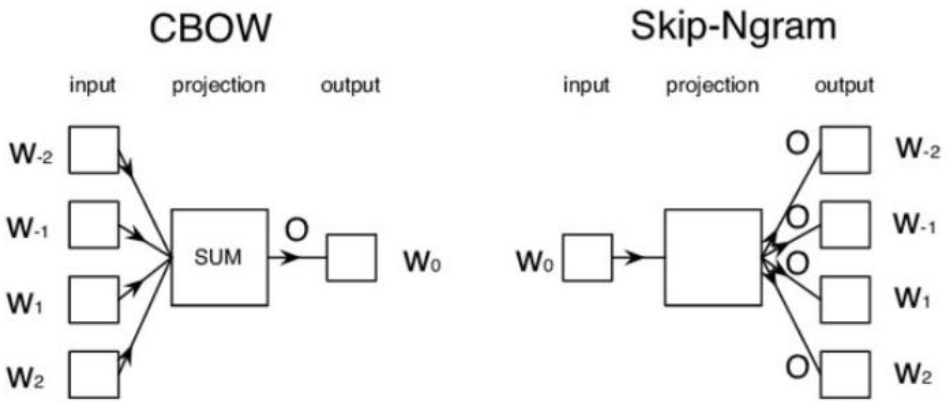

Рисунок 1.1 – Моделі CBOW та Skip-gram

CBOW – це алгоритм, який передбачає слово за контекстом сусідніх слів. З огляду статистики це призводить до того, що CBOW згладжує більшу частину інформації про розподіл внаслідок опрацьовування всього контексту як одного спостереження (це є корисним при невеликому розмірі оброблюваних текстів).

Skip-gram модель тренується на зворотній задачі – вона передбачає контекст слів, використуючи цільові слова. Статистично Skip-gram

розглядає кожну пару «контекст-ціль» як нове спостереження. Така модель краще враховує контекст на великих наборах даних.

GloVe – метод векторизації тексту, який побудований на отриманні семантичних зв'язків між словами з матриці спільного входження. Наведемо короткий опис цього методу. Нехай дано мовний корпус, який містить V слів, матриця спільного входження X буде матрицею $V \times V$, де i-й рядок і j-й стовпець X, X_ij позначає, скільки разів слово i зустрічалося разом зі словом j. Приклад матриці спільного входження наведено на рисунку 1.2.

	the	cat	sat	on	mat
the	0	1	0	1	1
cat	1	0	1	0	0
sat	0	1	0	1	0
on	1	0	1	0	0
mat	1	0	0	0	0

Рисунок 1.2 – Приклад матриці спільного входження
для методу GloVe

Наведена в цьому прикладі матриця спільного входження для речення «the cat sat on the mat» з одиничним розміром вікна є симетричною матрицею. Для отримання метрики, яка буде відображати семантичну подібність між словами з цієї матриці, слід взяти три слова одночасно. Перспективною моделлю вбудовування слів є нейромережева модель з використанням бібліотеки Keras для її навчання. Для реалізації вбудовування слів бібліотека Keras містить шар вбудовування (Embedding). Цей шар вбудовування реалізується у вигляді класу в Keras і використовується як перший шар в послідовній моделі для задач обробки текстів.

1.2 Завдання та типи класифікації новинних текстів

Задача класифікації у широкому сенсі полягає в розбитті деякої множини об’єктів (зокрема, аналізованих даних) на завчасно задані групи. Такі групи зазвичай називають класами, а спостереження мають однакові характерні властивості, ознаки, атрибути тощо, на основі яких і відбувається класифікація [8], [9]. При цьому деякі об’єкти, які мають схожі атрибути – будуть відноситися до одного класу, а інші – до наступних класів.

В задачах класифікації новинних текстів під множиною об’єктів мається на увазі множина оперативних новин, що повсякденно отримується користувачами з використанням засобів масової інформації (ЗМІ). Зазначимо, що новинна інформація розповсюджується здебільшого з застосуванням ресурсів мережі Інтернет.

Соціальний зміст новинних текстів обумовлений актуальністю і новизною, відповідністю запитам користувачів, глибиною аналізу соціальної дійсності, випереджуючою постановкою значущих проблем, , досконалістю форми подачі матеріалу тощо.

Специфіка мови новинних текстів полягає у взаємодії вербальних та графічних компонентів. Параметри шрифту, наявність ілюстрацій, використання кольору, розташування матеріалів на смугах графічно-текстових документів щільно поєднуються зі словесним рядом, утворюючи специфічну мову преси.

Для детального опису того чи іншого тексту ЗМІ виділено систему параметрів, а саме:

а) канал розповсюдження:

– преса;

– телебачення;

– Інтернет;

б) функціонально-жанровий тип тексту:

– новини;

– коментар;

– публіцистика (features);

– реклама;

в) тематична домінанта;

г) жанри текстів:

– оперативно-новинні;

– оперативно-дослідні;

– дослідницько-новинні;

– дослідницькі;

– дослідницько-образні.

Тексти ЗМІ мають стійкі форми, хоча їх зміст змінюється щодня. Отримувач новинних текстів знає заздалегідь, за допомогою яких мовних засобів йому можуть повідомити, наприклад, про майбутні вибори, міжнародні події , спортивні новини тощо.

Ядром оперативно-новинних текстів є новина, що містить інформацію, раніше невідому аудиторії. Сутність будь-якої новини утворюють факти, що мали місце в часі і просторі. Сприйняття факту залежить від характеру подачі новинного повідомлення, а також від рівня аудиторії та її інтересів (соціальних, вікових, професійних). Найважливішими вимогами до інформаційного повідомлення є його оперативність, релевантність, зрозумілість для аудиторії, стислість подання інформації.

В оперативно-дослідних текстах при загальному збереженні інтересу до новини на перший план виступає не оперативне значення інформації, а її тлумачення.

Дослідницько-новинні тексти об’єднує актуалізація проблеми, тобто прагнення зберегти новинне ядро переданої інформації та дати оцінку описуваним фактам. Акцент переноситься з новизни на актуальність, а також з викладу факту на його інтерпретацію.

В дослідницьких текстах узагальнення носить більш широкий характер з наданням всебічного аналізу явищ, процесів і проблем.

В дослідницько-образних текстах досліджуються закономірності соціально-морального буття людини і розвитку суспільних процесів, а також конкретні ситуації реальної дійсності. Предметом дослідження виступає людина і проблема, тобто конкретні життєві ситуації, що вимагають свого вирішення.

Головними функціями як традиційних ЗМІ, так і мережевих новин, визнаються інформаційна, впливова та інтерпретаційна. Зміст зазначених функцій ЗМІ змінюється у зв'язку зі специфічними характеристиками мережевих новин, такими як гіпертекстуальність (взаємопов'язаність), мультимедійність, інтерактивність, оперативність та вимірність.

Інформаційна функція полягає в інформуванні адресата про поточні події, а також у фокусуванні його уваги. Для мережевих новин характерним є: по-перше, наявність великої кількості ресурсів, серед яких кожен користувач може вибирати той, який вважатиме за потрібне; по-друге – швидке оновлення контенту в режимі реального часу; по-третє, неодночасне (асинхронне) отримання інформації користувачами.

Впливова функція мережевих новин проявляється через відбір інформації, коментування, розташування в певній послідовності.

В мережевих новинах можуть бути присутні як інформаційні жанри, так і аналітичні. Зазвичай, тексти новинного характеру крім інформативної функції виконують ще і впливову функцію.

Жанри новинних текстів можуть бути покладені в основу визначення класів в завданнях їх класифікації.

Якщо кількість класів у задачі класифікації дорівнює двом, має місце бінарна класифікація, якщо більше – багатокласова (або мультикласова). В цілому кількість класів може бути значною, наприклад, у випадку класифікації новинних текстів, таких класів може інколи складати декілька десятків.

Класифікація тексту загалом не відрізняється від класифікації звичних даних по атрибутам, оскільки в деякому сенсі такими атрибутами являються частини мови – слова, або леми (базова форма слова) [10].

Класифікувати новинний текст за сенсом допомагає семантика та статистичний аналіз. У контексті NLP терміни «семантика», «тема», «сенс» є досить схожими, і часто можуть заміняти один одного. Розпізнавати семантику тексту можна за допомогою латентно-семантичного аналізу, статистичних та ймовірнісних показників (TF-IDF, баєсів класифікатор) [11]. Для попередньої обробки текстів, що мають бути класифіковані, можуть використовуватися статистичні показники та здійснюватися токенізація тексту, розрахунок входження усіх слів в текст, виділення основних слів тощо. Слова, які найчастіше зустрічаються у тексті, можуть стати ключовим моментом до розкриття семантики.

Оцінка новинних текстів характеризується особливою структурою, де можна виділити ряд базових і периферійних елементів. Загальновизнаним фактом є наявність в структурі оцінки таких складових як суб’єкт, об’єкт і шкала оцінок.

В тексті мережевих новин оцінка є одним з основних засобів мовного впливу в зв’язку з тим що саме вона здатна регулювати поведінку як одержувача, так і відправника повідомлення. Потенційний впливовий ефект експліцитної або імпліцитної оцінки задається комунікативним наміром відправника повідомлення. Імліцитна оцінка носить завуальований характер і не нав’язується адресату як щось дане на відміну від експліцитного вираження авторської оцінки, яка може бути і не прийнята реципієнтом, викликаючи відторгнення. До лексичних засобів вираження оцінки в тексті мережевих новин відносяться лексичні одиниці з експліцитно і імпліцитно-оцінним значенням.

Серед особливостей вираження оцінки за допомогою лексичних засобів слід виділити тенденцію до різнополярних найменувань. Так, в залежності від спрямованості оцінки за параметром «об’єкт – суб’єкт», одні

і ті ж референти дійсності, наприклад, особи, учасники (сторони) військових конфліктів і різних військових операцій, можуть позначатися по-різному.

Тексти мережевих новин включають такі основні категорії, як «короткий зміст» (заголовок та анотація) та «основний зміст» (опис головної події, фону та коментарі.

Короткий зміст спрямований на інформування читачів про зміст подальшого повідомлення та на пригортання їх уваги.

Згідно основного змісту новинні повідомлення поділяються на «hard news – soft news» («тверді – м'які» новинні повідомлення), «local news – foreign news» (події у країні та за кордоном), а також на стійкі тематичні блоки типу політика, бізнес, розваги, спорт, технологія, наука, здоров'я, подорожі тощо.

Повідомлення «hard news» мають тверду фактологічну основу і відповідають на питання «що», «де», «коли»; повідомлення «soft news» ґрунтуються на факторі людського інтересу та орієнтовані на те, щоб викликати співчуття, здивування, захоплення. Повідомлення типу «hard news» зазвичай складають основу будь-якого новинного тексту, а повідомлення типу «soft news» доповнюють факти зверненням до загальнолюдських цінностей і емоцій. Повідомлення «hard news» характеризуються чіткою регламентованною структурою, офіційною стилізацією, наявністю великого числа цитат і посилань. Повідомлення «soft news» мають менш регламентовану структуру, розмовну стилізацію, менше число цитат і посилань. Тип повідомлень «local news – foreign news» стосується подій в країні і за кордоном. Реалізація даної категорії в різних національних ЗМІ відображає особливості національного світосприйняття.

Важливим різновидом новинних текстів є фейкові новини, поява яких є зростаючою проблемою в сучасному світі, викликаною різноманітними факторами, зокрема падінням довіри до визнаних ЗМІ, значним збільшенням кількості інформації, доступної для користувачів, а також зростанням онлайн-новин в соціальних медіамережах.

Користувачі мережі Інтернет мають доступ до великої кількості інформації, що це дає їм можливість миттєво бути добре поінформованими з будь-якої теми, однак при цьому не завжди можна оцінити ступінь достовірності такої інформаціїї. Все більше користувачів отримують новини з платформ соціальних медіа та онлайн-джерел. При цьому фейкові новини поширюються інколи швидше, ніж справжні. Багато приватних організацій, які перевіряють факти, намагаються боротися з цією проблемою, позначаючи фейкові новини та надаючи альтернативну інформацію, однак через величезну кількість інформації більшість фейкових новин отримує певне розповсюдження.

Одним з напрямів вирішення проблеми розпізнавання фейкових новин є використання машинного навчання. Методи машинного навчання вже успішно вирішують чимало проблем класифікації тексту, які вимагають швидкої класифікації великої кількості інформації (наприклад, для блокування спаму електронною поштою).

Класифікаційні завдання – це способи розв'язання різних прикладних задач. На практиці часто використовуємо алгоритми машинного навчання, щоб звести реальні задачі до завдань класифікації для їх вирішення. Прикладом такої задачі є ідентифікація спаму. Щоб вирішити цю проблему, ми часто класифікуємо електронні листи як «спам» або «не спам». Отже потрібно аналізувати типи задач класифікації з конкретної предметної області, і кожен тип задач може використовувати спеціальний метод моделювання. Це впливає на остаточне вирішення проблеми. Зіткнувшись із різними класифікаційними мітками зразків, особливо важливим стає визначення завдання класифікації цього типу питань.

Завдання класифікації, що є притаманні класифікації новинних текстів, можна поділити на чотири основні типи: бінарна класифікація, мультикласова класифікація, класифікації з декількома мітками та класифікація незбалансованої вибірки [12].

Бінарна класифікація – це поширений тип класифікації у машинному навчанні, необхідно класифікувати два взаємовиключні класи.

Зазвичай, така класифікація розглядає один клас, що належить до нормального стану, та інший клас, що належить до аномального стану. Наприклад, «новини справжні» – це нормальний стан завдань, а «новини фальшиві» – аномальний стан. Категорії в нормальному стані надається мітка категорії 0, а категорії в аномальному стані – мітка категорії 1.

Для вирішення задач бінарної класифікації застосовуються насамперед метод k найближчих сусідів, метод дерева рішень, метод опорних векторів та наївний баєсів класифікатор.

Мультикласова класифікація – це тип завдання класифікації з більш ніж двома класами. Кожен зразок можна помітити лише як один клас.

Мультикласова класифікація передбачає, що кожному зразку надається лише одна мітка.

Класифікація з декількома мітками належить до завдань класифікації, які мають дві або більше мітки класу, прицьому одна або кілька міток класу можуть бути передбачені для кожного прикладу.

Така класифікація відрізняється від бінарної та мультикласової класифікацій, де для кожного прикладу передбачається одна мітка класу.

Зазвичай завдання класифікації з кількома мітками моделюють за допомогою моделі, яка передбачає кілька вихідних даних, причому кожен вихідний результат прогнозується як розподіл ймовірностей Бернуллі. Можна сказати, що така модель здійснює кілька прогнозів бінарної класифікації для кожного прикладу.

Незбалансована класифікація – це тип завдання класифікації, де кількість прикладів у кожній категорії розподіляється нерівномірно. Це часто ілюструється завданням бінарної класифікації, де більшість прикладів відносяться до класу 0 і лише кілька прикладів відносяться до класу 1. Розподіл може варіюватися за ступенем незбалансованості (від 1:2 до 1:1000).

1.3 Методи та програмні засоби нейромережевої класифікації новинних тестів

На сьогодні існують методи та системи, які дозволяють вирішувати завдання класифікації новинних текстів таким чи іншим чином, але найбільшої популярності через свої можливості здобули нейронні мережі [13], [14]. У нейронних мережах за основу взято ідею роботи клітини-нейрона: через дендрити у ядро поступають електричні сигнали (інформація), яка з часом накопичується, що призводить до збудження та відсилання електричного сигналу вже до аксону. Також можна відмітити, що клітини можуть бути більш, або менш чуттєвими до певних дендритів, що призводить до збудження із меншим сингалом.

Архітектура нейронної мережі зазвичай залежить від складності задачі, а також безпосередньо її мети, оскільки кожна архітектура має кращі та гірші сторони.

Найпопулярнішими штучними нейронними мережами, які застосовуються у задачах NLP (зокрема, для побудови класифікаторів текстів) є згорткові нейронні мережі (англ. Convolutional Neural Network, CNN), рекурентні нейронні мережі (англ. Recurrent Neural Networks, RNN), мережі довгої короткочасної пам'яті (LTSM) [15], [16], [17].

Ці нейронні мережі для роботи з мовними даними використовують, зазвичай, алгоритми машинного навчання з вчителем, які намагаються вивести патерни і закономірності використання з множини попередньо анотованих пар вхідних і вихідних текстів. Розглянемо, наприклад, задачу класифікації новинного текста документа за однією з чотирьох категорій: Спорт, Політика, Наука та Економіка. Очевидно, що слова, які містяться в документі, допомагають віднести документ до теми, а потім, спираючись на зразки анотованих таким чином документів в кожній категорії, алгоритм машинного навчання з вчителем може вивести патерни використання слів та класифікувати нові документи. Методи машинного навчання ефективно

працюють в задачах, де визначити набір правил важко, а анотувати вхідні приклади відповідними мітками достатньо просто. Крім проблем, пов'язаних з обробкою неструктурованих вхідних даних в системі класифікації з неточними наборами правил, природній мові притаманні ще є і додаткові властивості, які ускладнюють розробку та практичну реалізацію обчислювальних методів на основі машинного навчання, а саме дискретність, композиційність і розрідженість.

Сенс фрази є більш складним, ніж сенсу окремих слів цієї фрази, і визначається набором різноманітних правил. Щоб інтерпретувати текст, доводиться розглядати довгі послідовності слів (наприклад, речення або більш масштабні текстові фрагменти). Дискретність та композиційність тексту призводить до розрідженості даних та неприпустимому збільшенню кількості комбінацій слів, що мають бути проаналізовані під час класифікації.

Нейронні мережі надають ефективний механізм навчання, ефективний для використання в задачах класифікації текстів. Головний компонент мовної нейронної мережі – шар занурення, тобто відображення дискретних символів на безперервні вектори в просторі порівняно невеликої розмірності. Якщо за міру відстані між словами аналізованого тексту прийняти відстань між векторами, то буде набагато простіше узагальнити взаємозв’язок слів. В процесі навчання, від шару до шару, мережа навчається комбінувати вектори слів корисним для прогнозування чином. Ця можливість дещо компенсує дискретність і розрідженість даних. Розглянемо два основних види архітектури нейронних мереж, які можна по-різному комбінувати: мережі прямого поширення і рекурентні мережі. Мережі прямого поширення (в тому числі і багатошарові персептрони), дозволяють працювати з вхідними даними фіксованого розміру або з даними змінного розміру, якщо не звертати уваги на порядок елементів. Багатошарові персептрони можна використовувати в тих випадках, де

необхідно брати до уваги нелінійність мережі, а також можливість інтегрувати в неї раніше навчені занурення слів.

Згорткові нейронні мережі (ЗНМ) – це спеціалізовані архітектури, що відрізняються здатністю виділяти локальні патерни в даних. Вони ефективно справляються з ідентифікацією фраз заздалегідь обмеженої довжини в довгих реченнях або документах.

Вихідні вектори ЗНМ об'єднуються в матрицю, в якій в кожен стовпець матиме не більше однієї одиниці. Кожен рядок отриманої матриці використовується як окрема карта ознак. На вхід згорткової нейронної мережі подається *m* карт ознак. Архітектуру ЗНМ можна вибирати згідно з конкретним завданням класифікації.

Рекурентні нейронні мережі (RNN) – це спеціалізовані моделі для послідовних даних. Вони приймають вхідну послідовність об'єктів і породжують вектор фіксованої довжини, який підсумовує її. Вихід рекурентної мережі можна подати на вхід мережі прямого поширення, яка спробує передбачити деяке значення. Рекурентні мережі є досить для побудови класифікаторів текстів. Вони дозволяють проектувати моделі, в яких умовами можуть бути цілі речення. При цьому вони можуть при необхідності враховувати порядок слів і не схильні до проблем статистичного оцінювання, що виникає з розрідженістю даних.

На відміну від традиційних нейронних мереж прямого зв’язку, RNN мають послідовності зі зворотним зв’язком, які дозволяють зберігати та передавати інформацію від одного кроку до наступного.

Основним будівельним блоком RNN є рекурентний нейрон, який отримує вхідні дані не тільки від поточного кроку, але й від власних результатів на попередньому кроці. Ця петля зворотного зв’язку дозволяє мережі обробляти послідовні дані та зберігати інформацію з часом.

Під час навчання RNN використовують алгоритм зворотнього поширення у часі (ВРТТ), який дозволяє оновлювати вагові показники мережі та оптимізувати її продуктивність. ВРТТ розширює і покращує

алгоритм зворотного поширення, який використовується в нейронних мережах прямого зв’язку, щоб врахувати послідовний характер RNN.

RNN мають стек нелінійних одиниць, де принаймні одне з’єднання між одиницями утворює спрямований цикл. Добре навчена RNN може моделювати будь-яку динамічну систему; однак навчання RNN є складним через проблеми із вивченням довгострокових залежностей [18].

Мережі RNN допомагають у різних сферах глибокого навчання та показують значний успіх у широкому діапазоні застосувань. Однак вони можуть зіткнутися з труднощами в охопленні довгострокових залежностей, оскільки може виникати проблема градієнта. Для усунення цього недоліку і досягнення кращої продуктивності були розроблені вдосконалені варіанти, такі як блоки рекурентного типу і трансформери.

Машинне навчання нейронних мереж комбінує велику кількість можливих використань (зокрема, і завдання класифікації текстів).

Багато задач в природній мові структуровані, тобто потребують породження складних вихідних структур типу послідовностей або дерев.

Класифікація передбачає визначення попередньо навченою моделлю приналежності вхідних даних до певного класу. Для вирішення задачі класифікації текстів необхідно пройти процес його попередньої обробки, що включає видалення шумів, стоп-слів, приведення до нижнього регістру, токенізацію та нормалізацію. Деталі цих процесів було описано у цьому розділі. Після цього етапу можна приступати безпосередньо до класифікації. Для цього існує велика кількість алгоритмів, і вибір правильного залежить від самої задачі класифікації, вхідних даних, вимог до точності і швидкості отримання результатів, доступності ресурсів для роботи моделі тощо. Загалом, цей крок є дуже важливим у процесі проектування. Вибір моделі визначає архітектуру та підхід, які використовуватимуться для навчання та прогнозування на основі текстових даних.

Крім систем класифікації, основаних на безпосередньому використанні архітектур CNN, RNN та LTSM, на практиці використовуються нейромережеві класифікатори текстів, що базуються, наприклад, на методах SVM (машини опорних векторів), Random Forest (випадковий ліс). Ці класифікатори мають свої певні переваги та недоліки.

Машини опорних векторів (SVM) – це відносно нова техніка, придатна, зокрема, для завдань бінарної класифікації текстів, яка містить елементи непараметричної прикладної статистики, нейронних мереж і машинного навчання. SVM – це контрольовані моделі навчання з пов'язаними алгоритмами навчання, які аналізують дані для класифікації та регресійного аналізу.

На сьогодні в откритому доступі є низка систем, що здійснюють класифікацію текстів різних тематик в реальному часі. Звісно, що найчастіше це API, які хоч і мають демо версію, проте є платними. Серед класифікаційних систем новинної тематики можна відзначити online класифікатор Dandelion API.

Дана система класифікує лише невеличкі частини текстової інформації і має досить точну класифікацію. Речення новинних текстів можуть містити різні ключові слова, що можуть відноситися до деяких категорій відразу і в такому випадку, система відображає відношення певного речення до декількох категорій одночасно у вигляді зафарбованих кольором областей на павутинні. Проте, незважаючи на цікавість даної системи, вона працює лише з певним невеликим параграфом новостної статті, великі в об'ємі текстові статті вона нажаль не оброблює. А саме така обробка має сенс в епоху великих масивів даних.

Система ParallelDots також дає можливість класифікувати новинні тексти. В результаті класифікації система видає відсоток належності введеного речення до категорії. Наприклад, для речення «Tennis, along with golf and angling, has been cited as a sport that can be played safely, while keeping

two metres apart from anyone else.» зі статті, що належить до категорії Спорт, система видає 73,9% приналежності категорії Спорт та значний відсоток категорії Розваги. І дійсно, певні слова, такі як «angling», «played» можно віднести до категорії Розваг.

Якщо провести аналіз стану сучасних нейромережевих технологій, то можна сформулювати висновок про те, що доцільність застосування конкретного типу НМ необхідно визначати на основі співставлення характеристик мережі з умовами поставленної задачі. До таких характеристик та умов відносяться:

– вимоги до обчислювальних потужностей;

– параметри навчальних даних;

– обмеження технічної реалізації НМ;

– загальні обмеження процесу навчання;

– вимоги до вихідної інформації, сфера застосування.

Розглянемо найпопулярніші програми, веб-додатки та бібліотеки, які дозволяють виконувати аналіз та класифікацію текстових даних [19].

Apache OpenNLP – це Java-бібліотека з відкритим кодом, яка використовується для обробки текстів природною мовою. OpenNLP надає такі можливості, як токенізація, сегментація речення, маркування мовлення, добування іменованого об'єкта, OpenNLP також включає максимальну ентропію і машинне навчання на основі перцептрону.

uClassify Веб-додатаки можуть бути використані для створення спам-фільтра, категоризації веб-сторінок, автоматизації підтримки електронної пошти, визначення мов, категоризації повідомлень блогів тощо.

Carrot2 – це механізм кластеризації результатів пошуку з відкритим кодом. Він може автоматично кластерувати невеликі колекції документів, наприклад, результати пошуку або реферати документів, в тематичні категорії. Carrot² написана на Java і розповсюджується під ліцензією BSD.

LibShortText – це програма з відкритим кодом для класифікації та аналізу короткого тексту. LibshortText може обробляти класифікацію імен,

питань, речень і коротких повідомлень. Він ефективніший, ніж пакети для загального текстового аналізу, та включає в себе інтерактивний інструмент для аналізу помилок. Виходячи з властивості, що кожен короткий текст містить декілька слів, LibshortText надає детальну інформацію для передбачення кожного тексту.

Text2data – це програма, що отримує важливу інформацію з текстових документів та створює докладні та гнучкі звіти про неструктуровані дані.

TextRazor – це програмне забезпечення, що призначене для класифікації конкретної інформації з текстових баз даних. Здатне створювати нові класифікації та класифікувати існуючі на їх основі набори даних. TextRazor може класифікувати інформацію з власних текстових баз даних клієнтів. Це досягається за рахунок попереднього обстеження загальнодоступних баз знань, таких як DBpedia (похідна від Вікіпедії), Freebase і Wikidata. Програмне забезпечення також може видобувати певні види інформації на основі запитаного виду інформації за допомогою нетипового класифікатора.

1.4 Постановка задачі

Мета даної кваліфікаційної роботи – дослідження методів нейромережевої класифікації новинних текстів.

Для цього в пропонованому дослідженні вирішуються такі завдання:

– аналіз існуючих типів новинних текстів та їх особливостей;

– аналіз існуючих методів попередньої обробки новинних текстів;

– дослідження засобів класифікації новинних текстів з використанням машинного навчання;

– розроблення узагальненої моделі нейромережевої класифікації новинних текстів;

– програмна реалізація та тестування розробленої моделі на прикладах бінарної та багатокласової класифікації новинних текстів.

2 УЗАГАЛЬНЕНА МОДЕЛЬ НЕЙРОМЕРЕЖЕВОЇ КЛАСИФІКАЦІЇ НОВИННИХ ТЕКСТІВ

2.1 Структура узагальненої моделі нейромережевої класифікації новинних текстів

Для реалізації завдань різних типів класифікації новинних текстів (НТ) з використанням сучасних архітектур нейронних мереж, в даній кваліфікаційній роботі запропоновано узагальнену модель, структуру якої наведено на рисунку 2.1.

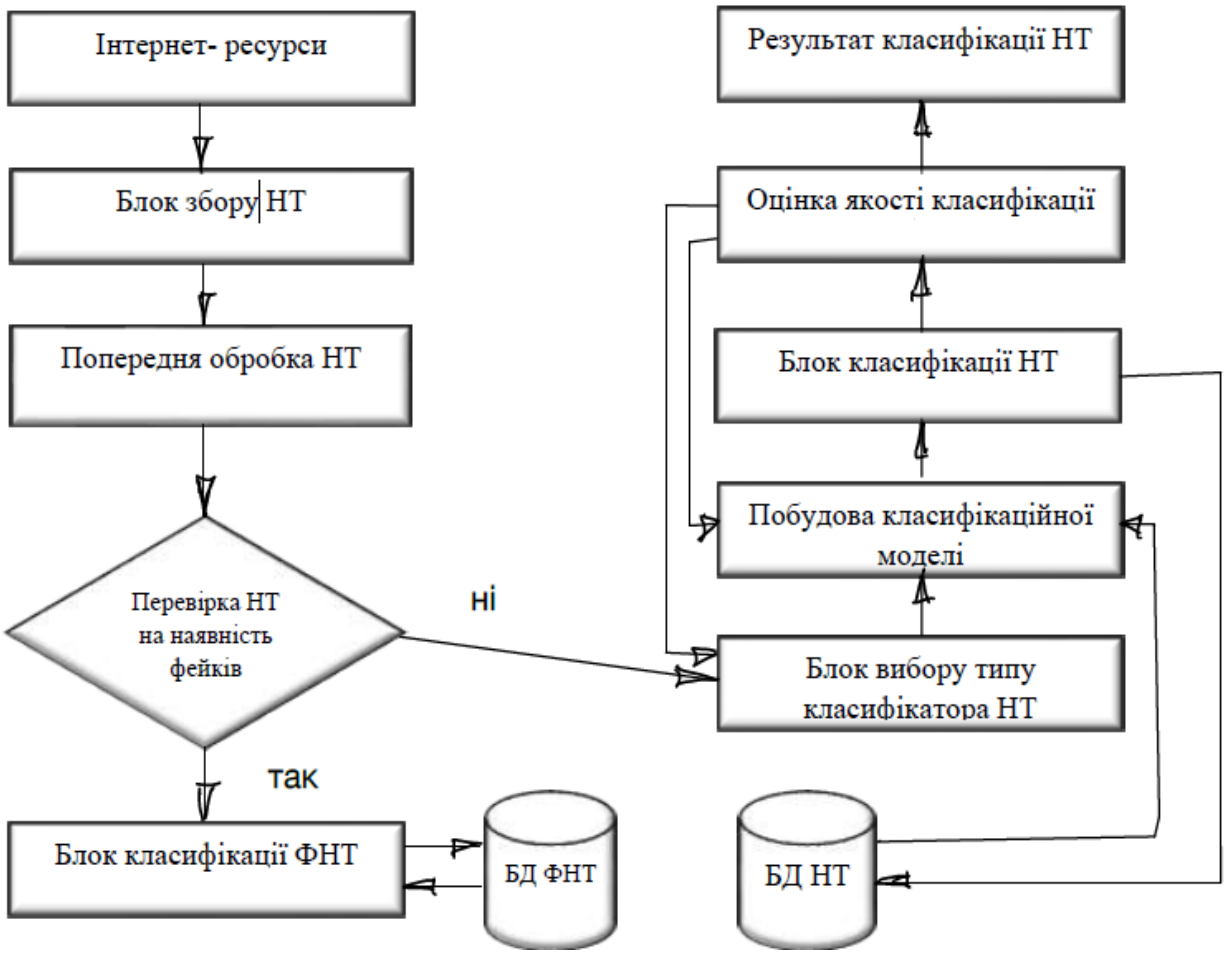

Рисунок 2.1 – Структура узагальненої моделі нейромережевої класифікації новинних текстів

Запропонована модель дозволяє побудувати та дослідити систему класифікації новинних текстів, що надходять з мережі інтернет. Така система може також здійснювати контент-моніторинг новинних Інтернет-ресурсів та дозволяє зберігати (в разі необхідності) інформацію щодо новинних текстів для подальшого її використання в аналітичній роботі.

Функціями моделі є:

– збір новинних текстів (НТ), що надходять з Інтернет-ресурсів (блок збору НТ);

– попередня обробка НТ (з використанням операцій очищення, токенизації, нормалізації, вбудовування слів);

– перевірка НТ на наявність в них фейкової складової (з використанням бінарного класифікатора);

– класифікація фейків у блоці класифікації ФНТ (в разі віднесення НТ до класу фейкових новинних текстів (ФНТ));

– перевірка НТ на наявність в них фейкової складової (з використанням бінарного класифікатора);

– вибір типу нейромережевого класифікатора НТ з використанням сукупності базових архітектур нейронних мереж (блок вибору типу класифікатора НТ);

– побудова класифікаційної моделі (з використанням правил генерації нейромережі обраного типу);

– класифікація поточних НТ з використанням побудованої класифікаційної моделі (блок класифікації НТ);

– оцінка якості класифікації НТ (з використанням оціночних критеріїв точності та повноти);

– формування результату класифікації НТ.

Відзначимо, що процес вибору типу класифікатора та подальшої побудови класифікаційної моделі може бути багатостадійним (в разі отримання незадовільної оцінки класифікації модель передбачає зміни типу класифікатора з подальшим його настроюванням).

В узагальненій моделі передбачено також використання баз даних БД НТ та БД ФНТ для формування навчальних виборок.

Розглянемо далі детальніше функції окремих блоків запропонованої узагальненої моделі нейромережевої класифікації НТ.

2.2 Збір та попередня обробка новинних текстів (НТ), що надходять з Інтернет-ресурсів

Збір та форматування новинних текстів, що надходять з Інтернет-ресурсів, здійснюється в оперативному режимі з визначених заздалегідь новинних ресурсів глобальної мережі (веб-сайтів, інтернет-порталів тощо).

Попередня обробка неструктурованих новинних текстів здійснює очищення текстових даних НТ та їх підготовку його до подальшого аналізу. Текстові дані містять шум у різних формах (емоції, пунктуація, орфографічні помилки тощо). Для того, щоб привести текст до зручної стандартної форми, необхідна попередня обробка тексту, яка складається з операцій очищення, токенізації та нормалізації.

2.2.1 Очищення тексту

Очищення тексту спрямоване на видалення небажаних або нерелевантних елементів із текстових даних. Для цього застосовуються операції усунення шуму, непотрібних символів або форматування, які можуть перешкоджати процесу класифікації [20]. Процес очищення тексту, що пропонується для використання в універсальній моделі класифікації ТН, передбачає виконання таких операцій:

– видалення спеціальних символів (наприклад, хештегів або смайлів), а також знаків пунктуації, які не впливають на значення тексту;

– видалення тегів HTML (якщо вони присутні після зчитування тексту ТН з HTML-сторінки Інтернет-ресурсу;

– обробка скорочень (скорочення можна розгорнути до повної форми, щоб стандартизувати текст);

– видалення числових символів (цифри або цифрові послідовності, які не стосуються завдання класифікації, можна видалити; втім, якщо числа мають смислове навантаження для розуміння тексту, їх необхідно зберегти);

– робота з великими літерами (приведення тексту до малого регістру, з метою уникнення проблем, спричинених розглядом слів, написаних різними літерами (великими або малими), як різних);

– видалення стоп-слів (стоп-слова, які не мають семантичного навантаження в тексті, можна видалити, щоб зосередитися на більш важливих ключових словах; втім ця операція не є обов'язковою, а її використання залежить від контексту завдання класифікації);

– орфографічна корекція (виправлення типових орфографічних помилок для покращення якості текстових даних);

– робота зі скороченнями (абревіатури в НТ можна розширити або стандартизувати до повної форми, щоб забезпечити узгодженість у тексті).

Метою очищення тексту є зменшення шуму та підготовка текстових даних до подальших етапів попередньої обробки, таких як токенізація, нормалізація та виділення спільних ознак. Завдяки видаленню нерелевантних елементів та стандартизації тексту, операції очищення допомагають покращити якість і надійність даних для завдань класифікації.

2.2.2 Токенізація тексту

Токенізація передбачає розбиття тексту на менші одиниці, які називаються токенами. Ці маркери можуть бути окремими словами, фразами, реченнями або навіть символами, залежно від рівня деталізації, необхідного для конкретного завдання. Токенізація є важливим кроком у попередній обробці тексту і часто виконується перед подальшим аналізом або моделюванням.

Найпоширенішою формою токенізації є токенізація слів, коли текст розбивається на окремі слова. Зазвичай це робиться шляхом поділу тексту на пробіли або знаки пунктуації.

Проте, у деяких випадках може знадобитися розбити текст на фрази або повні речення, а не на окремі слова.

Також інколи може знадобитися токенізація на рівні символів. Цей підхід розглядає кожен символ як окремий маркер. Така токенізація може бути корисною для завдань класифікації текстів.

В цілому, токенізація тексту перед класифікацією має на меті вирішення кількох завдань: розбиття тексту на токени полегшує обробку й аналіз даних (кожен токен стає одиницею аналізу або введенням для подальших кроків); визначення основних значущих одиниць в тексті, що дозволяє проводити точний аналіз і моделювання; поділ тексту на лексеми, що дає змогу застосовувати певні прийоми для кожної лексеми окремо.

2.2.3 Нормалізація тексту

Нормалізація в контексті попередньої обробки тексту відноситься до процесу перетворення слів або тексту в стандартне послідовне представлення. Вона спрямована на зменшення варіацій у формах слів і приведення їх до загальної бази або словникової форми, що дозволяє краще аналізувати та порівнювати. Техніки нормалізації допомагають гарантувати, що слова з однаковими або подібними значеннями розглядаються як рівні під час завдань обробки тексту. Серед поширених технік нормалізації текстів для їх подальшої обробки можна виділити такі, як стемінг і лематизація [21].

Стемінг – це техніка, яка скорочує слова до їх основи або кореня шляхом видалення суфіксів або префіксів. Отримане слово, яке називається основою, не завжди може бути правильним словом, але воно представляє основне значення. Наприклад, слова «біг», «бігає» і «бігти» мають корінь

«біг». Загалом, це підхід, який застосовує попередньо прописані правила визначення і видалення частин слова, що вносить зміну у значення кореня кожного слова, і він не завжди може давати точні основи.

Методи стеммінгу можна поділити на три групи: методи відсікання афіксів, статистичні методи та змішані методи. Кожна з цих груп має свою особливість формування основи слова.

Алгоритми першого типу відрізають префікси та суфікси слова, які є різновидами афікса. Найпростішим алгоритмом цього типу є Truncate (*n*) stemmer, який усікає слово, що починається з *n*-го символу. Ще однією простою реалізацією такого підходу є S-stemmer, який перетворює англійські слова з множини в однину, відсікаючи суфікси за певними правилами.

Статистичні алгоритми використовують лінгвістичний корпус, щоб прийти до набору правил, що будуються на основі аналізу слів, які мають однакову стему.

Для реалізації процедури стемінгу в узагальненій моделі класифікації текстів будемо використовувати стемер YASS (Yet another Suffix Stripper), запропонований П. Мадмуджером.

В цьому стемері для навчального корпусу створюються кластери з використанням евристичного підходу і функцій відстані між словами. Функція відстані між двома рядками визначає дійсне число, яке відповідає ступені подібності між цими двома рядками (чим менше її значення, тим більш близькими є рядки). З погляду стемінгу при малому значенні функції відстані аналізовані слова схожі морфологічно, а при великому значенні ці слова не пов'язані один з одним. Алгоритм орієнтований на мови, де словотворення здійснюється з використанням суфіксів і закінчень. При цьому слова вважаються спорідненими, якщо вони мають однакову першу частину слова (префікс і корінь). За алгоритмом стемера YASS для кожного рядка визначається міра відстані *D*.

Визначимо функцію p_i для двох рядків $X=x_0, x_1, ..., x_n$ та $Y=y_1, y_2, ...,$

y_n, яка знижує оцінку, коли слова не є близькими на початку роботи алгоритму:

$$p_i = \begin{cases} 0 \; if \; x_i = y_i \\ 1 \; otherwise \end{cases} \quad 0 \le i \le \min(n, n'). \tag{2.1}$$

Тоді відстань між рядками X і Y з довжиною n+1 буде дорівнювати:

$$D_1(X,Y) = \sum_{i=0}^{n} \frac{1}{2^i} p_i, \tag{2.2}$$

де p_i – допоміжна функція типу (2.1).

Позначимо перший неспівпадаючий символ між рядками X та Y як m та визначимо відстані D_2, D_3, D_4:

$$(x_1 = y_1, x_2 = y_2, \dots, x_{m-1} = y_{m-1}, x_m \neq y_m, \dots) \tag{2.3}$$

$$D_2(X,Y) = \begin{cases} \frac{1}{m} \sum_{i=m}^{n} \frac{1}{2^{i-m}} \; if \; m > 0 \\ \infty \; otherwise \end{cases}, \tag{2.4}$$

$$D_3(X,Y) = \begin{cases} \frac{n-m+1}{m} \sum_{i=m}^{n} \frac{1}{2^{i-m}} \; if \; m > 0 \\ \infty \; otherwise \end{cases}, \tag{2.5}$$

$$D_4(X,Y) = \frac{n-m+1}{n+1} \sum_{i=m}^{n} \frac{1}{2^{i-m}}, \tag{2.6}$$

Можна бачити, що ці відстані формуються множенням штрафу за невідповідність на коефіцієнт, спрямований на зменшення цього штрафу, якщо сусідня частина слова становить більшу частину слова (і навпаки, збільшення, якщо довжина всього слова набагато більше довжини частини, яка співпала).

Таким чином, функції відстані, запропоновані в даному алгоритмі, більше підходять для пошуку слів із загальною стемою більше, ніж традиційні. Певні функції відстані використовуються для об'єднання слів в однорідні групи. Передбачається, що кожна група являє собою набір однокореневих слів. До переваг алгоритму можна віднести те, що його можна використовувати для будь-якої мови, не знаючи її морфологічних правил. До недоліків – відносно висока обчислювальна складність.

Лематизація є більш просунутою технікою, яка також зводить слова до їх основної форми, але вона гарантує, що отримане слово є існує у мові. Лематизація враховує контекст і частину мови поточного слова для створення точних лем.

Наприклад, слово «краще» було б лематизовано до «добре», що є основною формою. Лематизація зазвичай потребує лінгвістичного аналізу та покладається на словник або лексичний ресурс для відображення слів у їхніх базових формах. Лематизація на відміну від стеммінгу, скорочує слова до їхньої основи, гарантуючи належність кореневого слова до мови. Зазвичай вона складніша, ніж стеммінг, оскільки лематизація працює над окремими словами без знання контексту. При лематизації кореневе слово називається лемою.

Вибір між стемінгом та лематизацією залежить від конкретних вимог завдання аналізу тексту та вимог по точності отриманих результатів. Загалом процес виділення коренів зі слів є швидшим і простішим, але він не завжди ефективний і може призвести до правильних слів, що згодом значно впливатиме на результати класифікації. Лематизація, з іншого боку,

продукує дійсні слова, але може бути дорожчою з точки зору обчислень і довшою, потребувати більше ресурсів.

Нормалізація допомагає вирішити проблему варіації слів у природній мові, дозволяючи точніше порівнювати, класифікувати або аналізувати текстові дані. Це допомагає зменшити надмірність і гарантувати, що подібні слова розглядаються як еквівалентні, покращуючи продуктивність подальшої обробки тексту та завдань машинного навчання.

2.2.4 Векторизація тексту

Під час вирішення задачі автоматичної класифікації текстів використовується їх представлення у векторному вигляді. Існує чимало методів векторизації тексту (зокрема, Word2Vec, TF-IDF, Skip-gram, Continuous Bag of Words, MUSE тощо). Але всі ці методи не дозволяють вирішити задачу багатомовної векторизації текстів, адже їх використання передбачає необхідність попереднього визначення мови, якою написано текст, перед опрацюванням тексту з застосуванням відповідною до мови моделі. Однак такий спосіб потребує написання багатої кількості коду для підтримки цієї логіки, а також наявності моделі для векторизації та для обробки векторів під кожну мову з якою потрібно працювати.

Більш прийнятними для векторизації багатомовних текстів є моделі mBERT та XLM-RoBerta, що передбачають використання замаскованих токенів [4]. На вхід цих моделей подається текст, в якому деякі токени замінені на спеціальний токен-маску, після чого необхідно, користуючись контекстом незамаскованих токенів, визначити які токени знаходяться під маскою. Для багатомовних реалізацій таких моделей можна використовувати навчальний набір даних, де присутні фрагменти текстів кількома мовами. Втім архітектурно багатомовні варіанти моделей mBERT та XLM-RoBerta не відрізняються від їх одномовної реалізації. Для багатомовної моделі необхідно мати датасет, який враховує велику кількість

мов. При цьому виникають суттєві проблеми з мовами, що недостатньо представлені в навчальному корпусі. Песпективним є комбіноване використання можливостей моделей типу XLM-RoBerta та згорткових нейронних мереж для побудови ефективних багатомовних класифікаторів текстів. Для реалізації процедури векторизації в узагальненій моделі класифікації текстів будемо використовувати два методи: Word2Vec (для одномовних текстів) та XLM-RoBerta (для багатомовних текстів). Метод Word2Vec пропонує векторизацію текста на рівні ембедингів слів. Метод є досить популярним завдяки легкій інтеграції до основних фреймворків машинного навчання та наявності векторів ембедингів слів для багатьох мов. Цей метод реалізується з використанням неглибокої нейронної мережі, яка навчена відновлювати лінгвістичний контекст слів. На вхід мережі подається великий корпус слів, для якого створюється створює векторний простір. Цей простір може мати декілька сотень вимірів, причому кожному слову в цьому вимірі призначено відповідний унікальний вектор. Вектори слів розміщуються у векторному просторі мережі таким чином, щоб слова з близьким контекстом мали малу векторну відстань, та були розміщенні поруч [22].

Для метода Word2Vec зазвичай використовуються дві алгоритмічно подібні моделі: Continuous-Bag-of-Word (CBOW) та Skip-gram (рисунок 2.2).

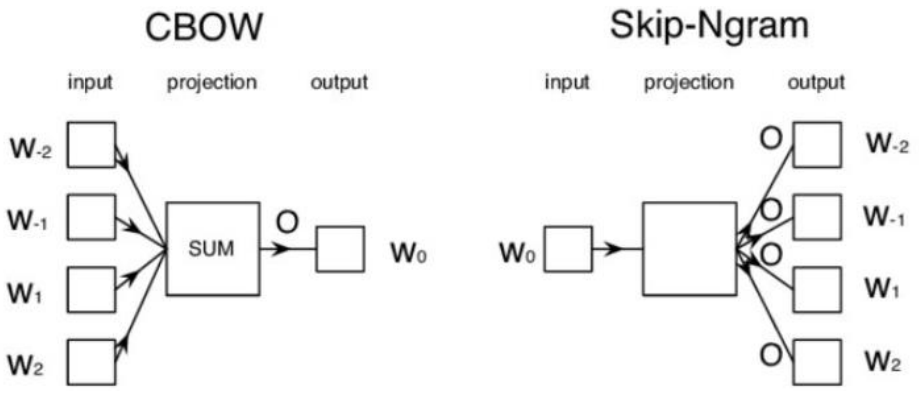

Рисунок 2.2 – Моделі векторизації текстів CBOW та Skip-gram

Алгоритм роботи моделі CBOW передбачає слово за контекстом сусідніх слів. Це призводить до того, що модель CBOW згладжує більшу частину інформації про розподіл та розглядає весь контекст як одне спостереження, що є доцільним при невеликому розмірі вхідного корпусу.

Модель Skip-gram вирішує на зворотню задачу, передбачаючи передбачає контекст слів за цільовими словами. Ця модель розглядає кожну пару «контекст-ціль» як нове спостереження, тому вона краще враховує контекст на великих наборах даних.

Базова архітектура Word2Vec (рисунок 2.3) подібна архітектурі автоенкодера: на вхід подається великий вектор тестових даних, який компресується за допомогою прихованого шару, а потім подається до вихідного шару, який має той самий розмір та функцію активації (Softmax), що і вхідний. Результатом роботи методу є ймовірність того, що вектор на виході прихованого шару є векторною репрезентацією слова, закодованого у вхідному векторі.

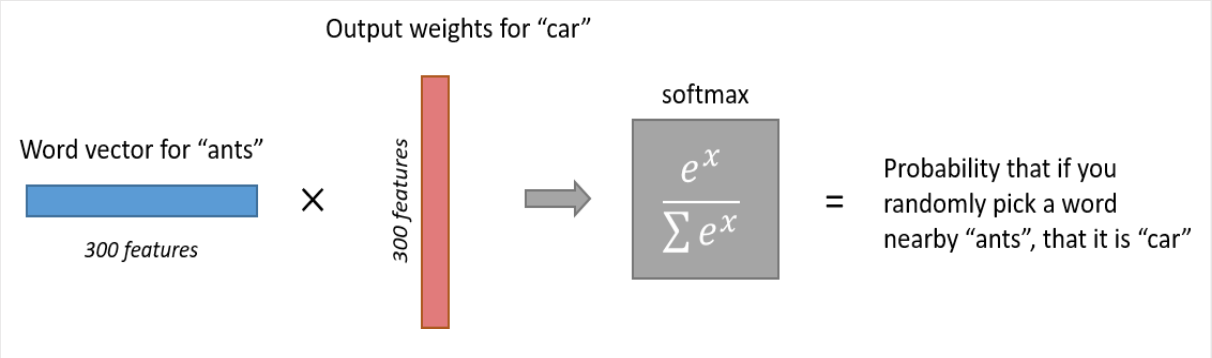

Рисунок 2.3 – Ілюстрація роботи методу Word2Vec

Розглянемо переваги використання для векторизації текстів за допомогою згорткової нейронної мережі на основі моделі XLM-RoBerta, що підтримує 100 мов.

На відміну від традиційних моделей ця модель має вищу точність на великому спектрі задач обробки природно мовних текстів (NLP – Natural-language processing).

Однією з особливостей моделі XLM-RoBerta є використання в механізмі внутрішньої уваги (self-attention) технології multi-head attention (багатоголової уваги) [3]. Ця технологія дозволяє замість використання однієї голови уваги з 512 розмірними векторами ключів, запитів та значень лінійно проеціювати ключі, значення та запити *h* разів з різними вивченими проекціями на відповідні вектори. Якщо значення *h* дорівнює 8, то кожна голова уваги має розмірність вихідної послідовності 64, що в сумі дозволяє обробляти послідовності, що не перевищують розмірність в 512 токенів. На кожну з цих проекцій ключів, запитів та значень паралельно застосовується функція внутрішньої уваги та отримуються відповідні вектори.

Після цього вектори об'єднуються та знову проеціюються, в результаті чого формуються остаточні значення (рисунок 2.4).

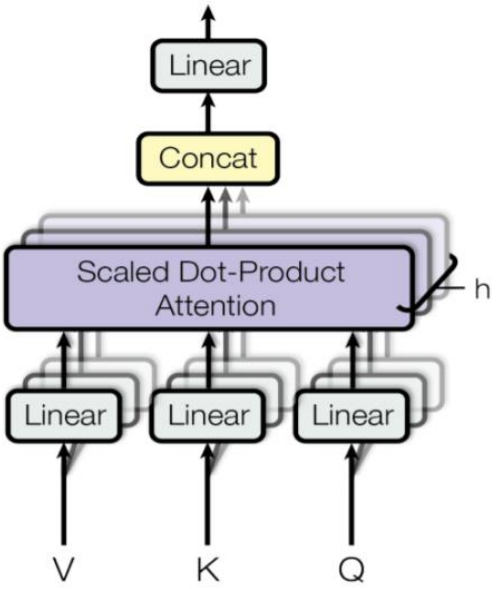

Рисунок 2.4 – Ілюстрація механізму багатоголової уваги для моделі XLM-RoBerta

Багатоголова увага дозволяє моделі XLM-RoBerta здійснювати доступ паралельно до різних частин вхідного речення. Формула для обчислення багатоголової уваги має наступний вигляд:

$$MultiHead(Q,K,V) = Concat(head_1,\dots,head_h)W^O\ , \qquad (2.7)$$

де $head_1 = Attention\left(QW_i^Q, KW_i^K, VW_i^V\right)$;

W_i^Q – матриця ваг для матриці запитів i;

W_i^K – матриця ваг для матриці ключів i;

W_i^V – матриця ваг для матриці значень i .

Результати порівняння складності операцій в різних шарах нейронної мережі з механізмом внутрішньої уваги, що передбачається далі використовувати у багатомовному класифікаторі, наведено в таблиці 2.1.

Таблиця 2.1 – Порівняння складності операцій в шарах мережі для моделі векторизаціїї XLM-RoBerta

Тип шару	Складність шару	Послідовність операцій	Максимальна довжина
Внутрішня увага	$O(n^2 * d)$	$O(1)$	$O(1)$
Рекурентний	$O(n * d^2)$	$O(n)$	$O(n)$
Згортковий	$O(k * n * d^2)$	$O(1)$	$O(log_k(n))$
Обмежена внутрішня увага	$O(r * n * d)$	$O(1)$	$O(n/r)$

У таблиці 2.1 прийнято такі позначення: n – кількість елементів у вхідній послідовності; d – розмір простору репрезентації вхідних даних (розмірність словнику моделі); k – розмір вікна операції згортки, r – кількість сусідів в шарі обмеженої складності внутрішньої уваги (Self-Attention restricted).

Результати порівняння показують, що шар внутрішньої уваги з'єднує усі позиції з постійною кількістю послідовно виконаних операцій, в той час

рекурентна мережа потребує $O(n)$ послідовних операцій. Шар власної уваги працює швидше, ніж рекурентна мережа, коли вхідна послідовність n менша ніж розмір простору репрезентації d, що є досить поширеною ситуацією.

Якщо замість рекурентних мереж використовувати згорткову мережу з розміром вікна $k < n$, то виникає проблема з'єднання вхідних та вихідних позицій, адже декодер буде втрачати частину інформації, що надійшла з енкодеру. Щоб вирішити цю проблему, потрібно використання $O(n/k)$ згорткових мереж. При цьому необхідним є врахування позиції кожного слова у вхідних даних моделі.

2.3 Перевірка новинних текстів на наявність фейків та класифікація ФНТ

Пропонована узагальнена модель нейромережевої класифікації передбачає необхідість перевірки вхідних новинних текстів (після їх попередньої обробки) на наявність фейкової складової.

На сьогоднішній день існує багато методів виявлення фейкових новин, але більшість з них базуються на ручному аналізі тексту, що уповільнює процес виявлення та не дає можливості його автоматизувати. Одним з найбільш перспективних методів є використання моделі машинного навчання, яка може бути впроваджена в системи моніторингу новин та інформаційні портали для автоматичного виявлення фейкових новин у текстах. Така модель може суттєво зменшити поширення фейкових новин, підвищити якість інформації для потенційних користувачів та зміцнити довіру до засобів масової інформації [23], [24], [25].

В запропонованій узагальненій моделі класифікації НТ для виявлення фейків пропонується використання бінарного класифікатора (два класи: НТ без фейків та НТ з фейковою складовою (або ФНТ)), що реалізується на базі неглибокої нейронної мережі (наприклад, на базі багатошарового персептрону (БШП)). В цьому класифікаторі використовується алгоритм

класифікації текстів на основі нейронних мереж зі зворотнім поширенням помилки. Для побудови нейронної мережі тут використовується набір даних, який містить реальні та фейкові новинні тексти. Нейромережева архітектура пропонованої моделі містить вхідний, прихований та вихідний шари. Для прихованого шару моделі використовується функція активації *ReLU*, а для вихідного шару – функція активації *sigmoid*. Як функції втрат використовується бінарна перехресна ентропія:

$$h_j = \mathrm{ReLU}\left(\sum_{i=1}^{n} w_{ij}\, x_i + b_j\right). \tag{2.8}$$

Навчання моделі бінарного класифікатора ФНТ з використанням алгоритму зворотного поширення помилки передбачає її тренування на навчальному наборі даних та оцінювання її ефективності на тестовому наборі даних. Ефективність моделі оцінювалася на основі метрики точності (accuracy):

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN}, \tag{2.9}$$

де *TP* – кількість правильно класифікованих позитивних зразків;

TN – кількість правильно класифікованих негативних зразків;

FP – кількість неправильно класифікованих позитивних зразків;

FN – кількість неправильно класифікованих негативних зразків.

Тексти, в яких бінарний класифікатор узагальненої моделі виявив фейкову складову (тобто ФНТ) можуть бути додатково проаналізовані у блоці класифікації ФНТ з використанням багатокласового нейромережевого класифікатора НТ, що розглядається далі (підрозділ 2.4). Перелік класів ФНТ та екземпляри ФНТ для формування навчальних виборок зберігаються у базі даних ФНТ (БД ФНТ).

2.4 Вибір типу нейромережевого класифікатора новинних текстів

Вибір типу нейромережевого класифікатора новинних текстів, для яких було підтверджено відсутність фейкової складової, здійснюється в узагальненій моделі з використанням сукупності базових архітектур нейронних мереж (блок вибору типу класифікатора НТ).

До сукупності таких мереж входять:

– багатошаровий персептрон (MLP), базова версія;

– нейронна мережа довготривалої короткочасної пам'яті (LSTM), базова версія;

– рекурентна нейронна мережа (RNN), базова версія;

– глибока нейронна мережа прямого поширення (DNN), базова версія;

– згорткова нейронна мережа (CNN), базова версія;

– нейронна мережа опорних векторів (SVM).

Розглянемо особливості базових архітектур нейронних мереж, що входять до блоку вибору типу класифікатора НТ універсальної моделі.

Багатошаровий персептрон (англ. Multi-layer Perceptron, MLP) є типом штучної нейронної мережі з прямим зв’язком. Типова мережа MLP – це повнозв’язна мережа, яка складається з вхідного рівня, який отримує вхідні дані, вихідного рівня, який приймає рішення або передбачення щодо вхідного сигналу, і одного або кількох прихованих шарів між цими двома, які вважаються обчислювальним механізмом мережі. Типову архітектуру MLP з одним прихованим шаром наведено на рисунку 2.5.

Вихід мережі MLP визначається за допомогою різноманітних функцій активації, також відомих як функції передачі, таких як ReLU (Rectified Linear Unit), Tanh, Sigmoid і Softmax.

Розглянемо детальніше архітектуру нейронної мережі LSTM (Long Short Term Memory), що використовується в моделі для реалізації завдань багатокласової класифікації НТ [26].

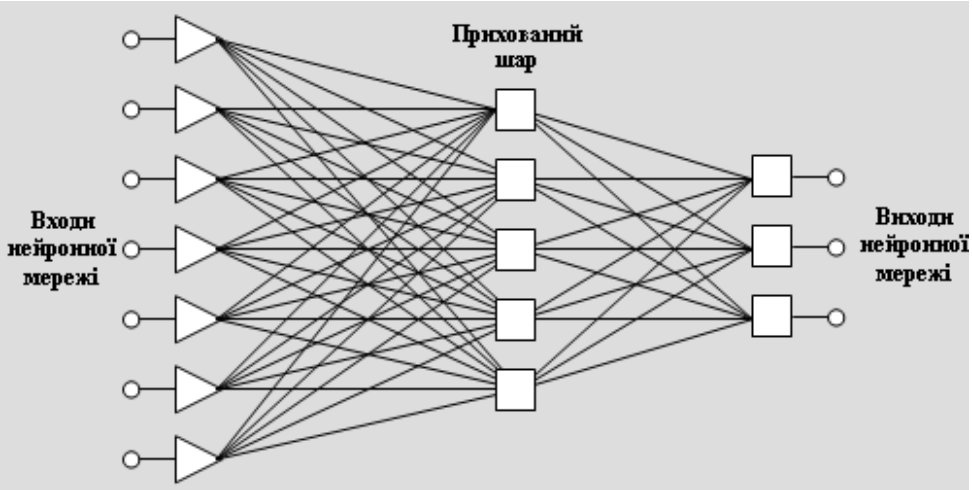

Рисунок 2.5 – Загальна структура мережі MLP з одним прихованим шаром

Ця мережа є модифікацією архітектури рекурентних нейронних мереж RNN, що були розроблені для вирішення задач, в яких вхідна послідовність представлена у вигляді ряду. Але якщо вхідні послідовності мають велику довжину, то базова архітектура RNN працює незадовільно через проблему затухаючого градієнту: градієнт під час проходження від кінця мережі до її початку становиться занадто малим, внаслідок чого не змінюються ваги шарів в мережі. Вирішенню даної проблеми сприяє архітектура LSTM (рисунок 2.6).

В LSTM використовуються у кожній комірці вхідний фільтр, вихідний фільтр та фільтр-відсіювач.

Фільтр-відсіювач необхідний для відсіювання нерелевантних даних з попередньої комірки, що допомагає використовувати лише релевантну для нейронної мережі інформацію в поточний час.

Вихід даного фільтра формується таким чином:

$$f_t = \sigma(W_f * [h_{t-1}, x_t] + b_f \ , \tag{2.10}$$

де W_f – ваги для шару фільтру;

h_{t-1}– вихід попереднього шару;

x – вхідні дані;

b_f – зсув.

Фільтр вхідних даних потрібен для того, щоб визначати, яку інформацію з вхідної послідовності потрібно зберегти у стані даної комірки. Вихід даного фільтра формується таким чином:

$$i_t = \sigma(W_i * [h_{t-1}, x] + b_i), \tag{2.11}$$

де W_i – ваги для шару фільтру;

h_{t-1}– вихід попереднього шару;

x – вхідні дані;

b_i – зсув.

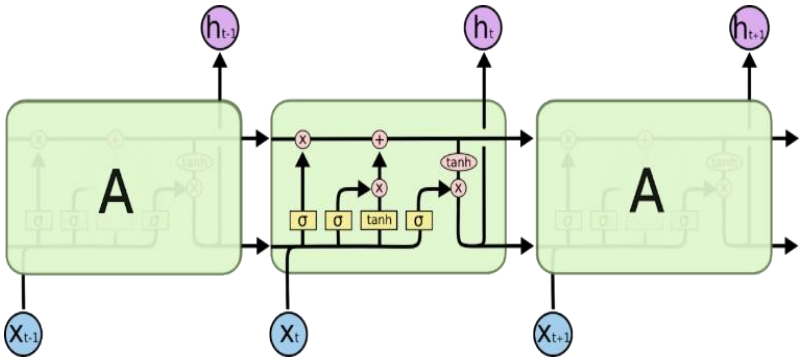

Рисунок 2.6 – Архітектура нейронної мережі LSTM

Фільтр вихідних даних потрібен для коригування інформації, яку ми хочемо передати у наступну комірку. Вихід даного фільтра формується таким чином:

$$o_t = \sigma(W_o * [h_{t-1}, x] + b_o), \tag{2.12}$$

де W_o – ваги для шару фільтру;

h_{t-1}– вихід попереднього шару;

x – вхідні дані;

b_o – зсув.

LSTM здатна обробляти великі послідовності текстових даних без виникнення проблем з вагами мережі, які не змінюються.

Слід відзначити, що існує можливість комбінового використання LSTM з засобами платформи Tensorflow Serving для побудови багато класових класифікаторів НТ. Платформа Tensorflow Serving характеризується наявністю високорівневого API Keras, що дозволяє зменшити кількість часу, необхідного для створення прототипів моделей та зменшує складність коду, необхідного для створення програмного продукту.

Розглянемо принцип дії базової версії рекурентної нейронної мережі, що що використовується в моделі для реалізації завдань багатокласової класифікації НТ малої довжини (рисунок 2.7).

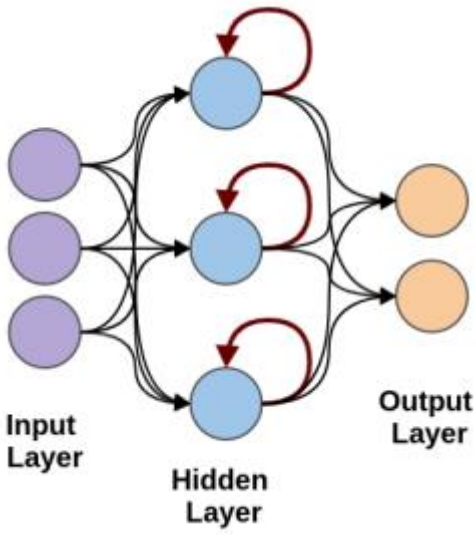

Рисунок 2.7 – Принцип дії нейронної мережі RNN

В базовому варіанті мережі RNN вихід певного шару зберігається і подається назад на вхід, що дозволяє передбачити результат роботи цього шару. Формування першого відбувається з урахуванням добутку суми ваг

та ознак, а наступних шарах починається повторюваний (рекурентний) процес роботи нейронної мережі.

Протягом поточного часового дискретного інтервалу роботи RNN кожен вузол запам'ятовує інформацію з попереднього часового інтервалу. Таким чином, кожен вузол виконує функцію регістру пам'яті під час обчислення та виконання операцій. Нейронна мережа починається з переднього розповсюдження, як зазвичай, але запам'ятовує і використовує в подальшому необхідну для остаточної класифікації НТ інформацію. RNN самостійно навчається та покращує якість прогнозуванням під час роботи.

В глибинних нейронних мережах DNN реалізується принцип глибинного навчання (Deep Learning), згідно з яким за допомогою математичної абстракції нейронної мережі можна симулювати процес отримання інформації та прийняття рішень (аналогічно роботі людського мозоку). На рисунку 2.8 наведено приклад структури DNN.

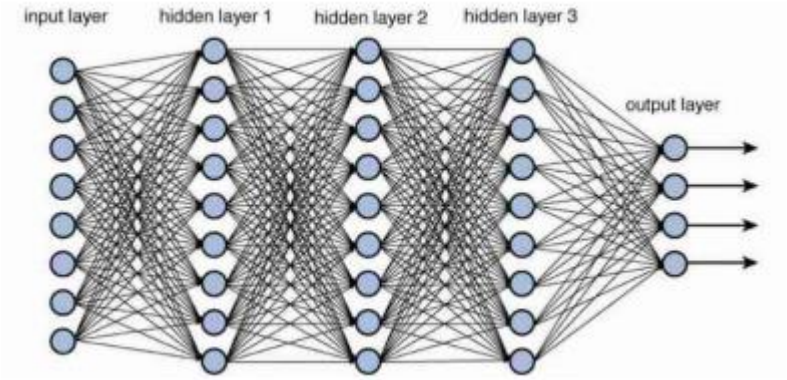

Рисунок 2.8– Принцип структури мережі DNN

Нейронна мережа DNN є по суті зваженим графом, де кожне ребро має різну вагу. В DNN кожен вузол моделює нейрон, ребра виконують роль дендритів та аксонів, а сигнал, що передається між нейронами є результатом множення активаційних функцій нейронів на вагу відповідного ребра [34]. Відзначимо, що алгоритми глибинного навчання досить

складні і вимагають значної обчислювальної потужності для ефективного навчання. Втім, DNN часто показають найкращі результати серед моделей під час вирішення завдань класифікації. Вони можуть мають досить високий поріг навчання, завдяки чому датасет можна нарощувати й покращувати показники. З цих причин нейронні моделі DNN на сьогодні є популярні для вирішення класифікаційних завдань, що потребують високої точності та класифікують велику кількість класів.

Згорткові нейронні мережі (CNN, Convolutional Neural Network) дозволяють ефективно реалізувати алгоритми глибокого навчання для вирішення завдань різного призначення (зокрема, для завдань класифікації зображень та текстів).

Мережа CNN використовує варіацію багатошарових перцептронів та містить один або декілька згорткових шарів. Ці шари можуть бути повністю пов'язані між собою або об'єднані. На рисунку 2.9 наведено приклад структури CNN, призначеної для обробки текстів.

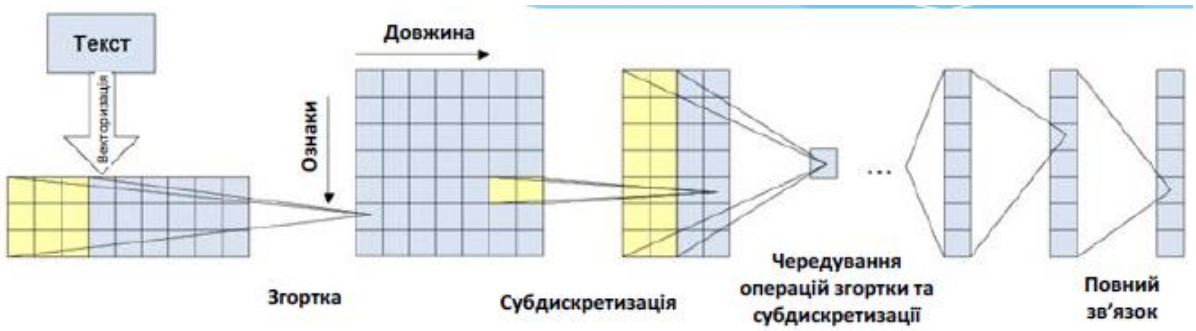

Рисунок 2.9 – Приклад структури мережі CNN, призначеної для обробки текстів

Згорткові нейронні мережі є різновидом багатошарових нейронних мереж прямого поширення, що характеризується меншими вимогами до попередньої обробки вхідних даних.

Основою згорткових нейронних мереж є згорткові шари, які складаються з фільтрів. Операція згортки передбачає обчислення

скалярного добутку даних фільтру та вхідних даних, що формує карту збудження фільтру.

Перш ніж передати результат наступному шару, згортковий шар використовує згорнуту операцію на вході. Завдяки цій згортковій роботі мережа може бути набагато глибшою, але зі значно меншими параметрами.

Окрім згорткових шарів, CNN мають також шари субдискретизації, що зменшуть розмірність вхідного сигналу шляхом операції усереднення або знаходження максимума (пулінговий шар), а також звичайні повнозв'язні шари багатошарового перцептрону (MLP) для кінцевої класифікації. Їх перевагою є менша кількість параметрів, необхідних для обробки вхідного сигналу певного розміру, що допомагає полегшити процес навчання мережі та позбутися проблеми перенавчання.

Пулінговий шар вибирає із вже існуючих ознак, які були обрані минулим шаром, найбільш значущі за допомогою деякої функції активації (наприклад, max-функції, що з набору значень обирає максимальне).

Повнозв'язний шар дозволяє вивчити нелінійні комбінації високорівневих ознак, які представляються у результаті минулих операцій. На вхід цього шару у нейрони подаються значення, ініціалізуються ваги, та за допомогою одних з методів навчання (зазвичай це метод оберненого розповсюдження помилки, англ. Backpropagation) налаштовуються параметри моделі, які передаються у вихідний шар.

Нейронні мережі опорних векторів, відомі також як машини опорних векторів (Support Vector Machines – SVM) є одним типом ШНМ, що використовують комбіноване навчання. Ці нейромережі відносяться до нейромережі з прямою передачею інформації, де як активаційні функції використовуються ядерні конструкції і вони являються узагальненням таких популярних конструкцій, як багатошарові персептрони, радіально базисні і поліноміальні мережі. Ці ШНМ реалізують метод мінімізації емпіричного

ризику і знаходять застосування при вирішенні задач ідентифікації, розпізнавання та класифікації образів (зокрема, класифікації текстів) тощо.

За кількістю класів досліджувані в узагальненій моделі типи класифікаторів діляться на бінарні та мультикласові.

За використанням мови та алфавіту досліджувані в узагальненій моделі типи класифікаторів діляться на одномовні (україномовні, англомовні тощо) та мультимовні.

Розглянемо далі деякі специфічні методи класифікації, реалізація яких може здійснюватися нейромережевими класифікаторами новинних текстів з використанням нейронних мереж удосконаленої моделі, а саме: метод опорних векторів, метод «випадкового лісу», метод найближчого сусіда.

Метод опорних векторів (SVM) – це відносно нова техніка, придатна для завдань бінарної класифікації, яка пов’язана та містить елементи непараметричної прикладної статистики, нейронних мереж і машинного навчання. Цей метод реалізується на основі використання нейронної мережі SVM та контрольованої моделі навчання з пов’язаними алгоритмами навчання, які аналізують дані для класифікації та регресійного аналізу. Метою машинного алгоритму опорних векторів є знаходження гіперплощини в N-вимірному просторі (N – кількість ознак), яка чітко класифікує точки даних (стосовно обраних ознак аналізованих НТ). Існує чимало можливих гіперплощин для розділення двох класів точок даних. Метод SVM дозволяє знайти площину, яка має максимальну відстань між точками даних обох класів. Збільшення такої відстані забезпечує можливість того, щоб майбутні точки даних можна було класифікувати з більшою впевненістю. На рисунку 2.10 наведено ілюстрацію принципа роботи методу опорних векторів.

Перевагами методу SVM (стосовно завдань класифікації НТ) є його алгоритмічна простота та можливість класифікації різних типів даних НТ. Метод SVM може працювати як з короткими так і довгими текстами, забезпечуючи високу надійність і прийнятну точність класифікації.

Недоліком цього методу є те, що він здійснює тільки бінарну класифікацію, тобто підтримує розподіл лише між двома класами.

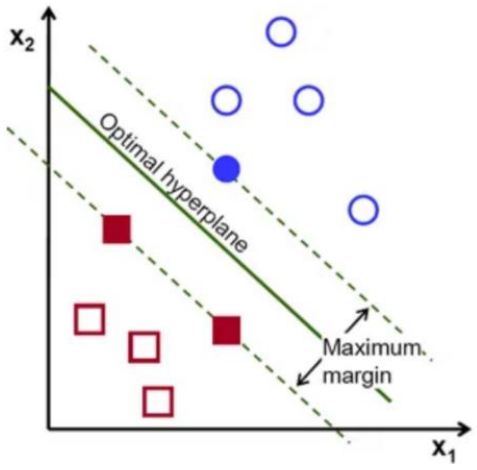

Рисунок 2.10 – Ілюстрація до принципу роботи методу опорних векторів

Метод «випадкового лісу» (Random Forest) може використовуватися в моделі як для бінарної, так і для мультикласової класифікації НТ.

«Випадковий ліс» складається з великої кількості окремих дерев рішень, які працюють як ансамбль. Кожне окреме дерево у випадковому лісі видає прогнозований клас, а клас, що набрав найбільшу кількість голосів, стає прогнозом моделі [27].

Припустимо, що набір даних в НТ складається з N прикладів, розмірність простору ознак дорівнює M і задано параметр $m < M$. Дерева будуються незалежно одне від одного за таким алгоритмом:

– з початкового набору даних генерується випадкова вибірка з повторенням записів так, що деякі з них потраплять в неї кілька разів, а деякі (приблизно $N/3$) з прикладів не ввійдуть у неї взагалі;

– формується дерево рішень для класифікації кожного запису вибірки, згенерованої на попередньому кроці. При цьому в ході створення кожного вузла дерева вибирається ознака, на основі якої проводиться розбиття (з випадково вибраних m ознак з можини M початкових ознак;

– дерево формується доки не вичерпаються усі записи вибірки.

Результат класифікації об'єктів визначається шляхом голосування. Кожне дерево комітету відносить об'єкт, який класифікується, до одного з класів, і перемагає клас, який отримав найбільше число голосів.

Оптимальна кількість дерев вибирається так, щоб мінімізувати похибку класифікатора в тестовій вибірці. Випадкові ліси, отримані в результаті застосування описаного методу, можуть використовуватися для оцінки важливості змінних у задачах класифікації НТ.

Ілюстрацію роботи методу Random Forest наведено на рисунку 2.11.

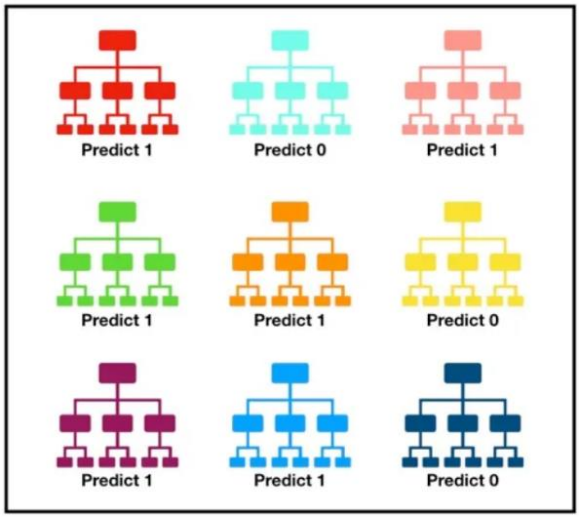

Рисунок 2.11 – Ілюстрація до принципу роботи методу Random Forest

Ефективність моделі Random Forest обумовлюється низькою кореляцією між її компонентами (деревами). Некорельовані моделі створюють складні прогнози, які є більш точними, ніж будь-який окремий прогноз.

Метод найближчого сусіда (KNN – K nearest neighbor) відносить об'єкт, що класифікується, до класу, якому належить найближчий навчальний об'єкт. Робота методу ґрунтується на зберіганні даних в пам'яті для порівняння з новими елементами. При появі нового запису для прогнозування визначається відхилення між цим записом і подібними

наборами даних, і найбільш подібний запис (ближній сусід) ідентифікується. Для змешення обчислювальних затрат іноді зберігається лише множина «типових» випадків. В такому випадку модифікований метод найближчого сусіда називають методом прецедента за аналогією (CBR). Цей метод відноситься до категорії методів навчання без вчителя, завдяки чому робочі характеристики кожної бази прецедентів з часом і поліпшуються. Метод не створює будь-яких моделей або правил, узагальнюючих попередній досвід, у виборі рішення вони ґрунтуються на усьому масиві доступних історичних даних, тому неможливо сказати, на якій підставі будуються відповіді. При цьому існує складність вибору міри «близькості» (метрики). Від цієї міри залежить обсяг множини записів, які потрібно зберігати в пам'яті для досягнення задовільної класифікації. Крім того, існує висока залежність результатів класифікації від обраної метрики. Метод не підходить для вирішення задач великої розмірності за кількістю класів і документів. Однак, використовуючи цей метод, не потрібно будувати класифікуючу функцію, що дає можливість оновлювати навчальну вибірку без перенавчання класифікатора. Алгоритм є стійким до аномальних викидів у вихідних даних. Програмна реалізація алгоритму відносно проста і результати роботи алгоритму легко піддаються інтерпретації.

Крім зазначених специфічних методів класифікації в моделі для класифікації НТ можуть безпосередньо використовуватися нейронні мережі (як базові варіанти, так і їх гібридні архітектури). Як вхідні дані при цьому обирається вектор параметрів аналізованого НТ. Результатом роботи мережі буде код класу, до якого належить поданий на вході об'єкт (НТ). Відзначимо, що якість класифікації може суттєво залежати від вдалого вибору комбінації типу мережі та обраних методів попередньої обробки (нормалізації, векторизації тощо).

2.5 Побудова класифікаційної моделі та класифікація поточних НТ (блок класифікації НТ)

Після вибору загальної структури класифікаційної моделі та методу класифікації необхідно визначити параметри нейронної мережі. Наприклад, для мереж, подібних MLP, такими параметрами є число шарів, число нейронів у прихованих шарах, наявність або відсутність обхідних з'єднань, передатні функції нейронів. При визначенні кількості шарів і нейронів слід брати до уваги, що здатність мережі до узагальнення завдання класифікації підвищується з підвищенням сумарної кількості зв'язків між нейронами. При цьому кількість таких зв'язків обмежується наявних навчальних даних. Етап вибору параметрів особливо важливий для ШНМ, шо навчаються з учителем. Від їх правильного вибору безпосередньо залежить збіжність роботи текстових класифікаторів. Зокрема, вибір низької швидкості навчання збільшує час збіжності, однак сприяє уникненню зупинки мережі внаслідок її потрапляння до локального оптимуму. Збільшення швидкості навчання може привести як до збільшення, так і до зменшення часу збіжності (в залежності від форми поверхні похибки). Зазвичай, значення параметрів необхідно вибирати експериментально згідно з обраним критерієм завершення навчання (наприклад, мінімізацією похибки або обмеженням за часом навчання).

У процесі навчання мережа конкретного обраного типу за порядком, притаманним правилам побудови цієї мережі, аналізує навчальну вибірку. Порядок такого аналізу може бути послідовним, паралельним,випадковим тощо. При навчанні з учителем ШНМ аналізує вибірку кілька разів, при цьому один повний прохід по вибірці називається епохою навчання. Набір навчальних даних поділяють на дві частини: власне навчальну вибірку і тестові (перевірочні) дані. Дані навчальною вибірки подаються на вхід мережі для навчання, а перевірочні дані дозволяють оцінити похибки мережі. При цьому, якщо на перевірочних даних похибка буде

зменшуватися, то мережа відповідає вимогам узагальнення. Якщо ж похибка на навчальних даних зменшується, а на тестових даних збільшується, то мережа перестає відповідати цим вимогам. Це свідчить про перенавчання (оверфіттінг) мережі. У таких випадках навчання зазвичай припиняють. Для запобігання перенавчання застосовується між шарами ШНМ метод виключення (англ. dropout).

Згідно з цим методом в процесі навчання випадково із загальної мережі багаторазово видаляється деяка підмережа, після чого оновлення ваг виконується тільки в її межах. Нейрони потрапляють в підмережу з ймовірністю p (коефіцієнт дропаута). На наступному етапі на даних тренується лише скорочена мережа, але після тренування видалені вузли повторно вставляються до ШНМ з первинними вагами (рисунок 2.12).

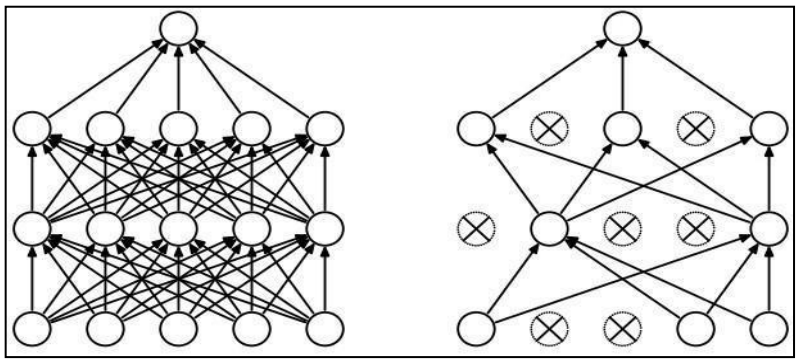

Рисунок 2.12 – Ілюстрація до застосування техніки Dropout на етапі навчання класифікатора

Таким чином, процедура виключення дозволяє уникати необхідності тренування всіх вузлів ШНМ класифікатора текстів на всій сукупності тренувальних даних, що компенсує негативний вплив ефекту перенавчання мережі. Крім того, метод виключення суттєво покращує швидкість тренування. Він широко застосовується, зокрема, в нейромережевих класифікаторах на основі згорткових нейронних мереж (CNN). Розглянемо докладніше деякі особливості побудови класифікаторів текстів цього типу.

Згідно з рисунком 2.9 в цій ШНМ таких класифікаторів присутні дві основні частини: виділення ознак НТ (feature extraxion) та класифікація НТ (classification). Виділення ознак є інструментом згортки, який розділяє та ідентифікує різні характеристики НТ для їх аналізу та подальшого використання на етапі класифікації.

Блок виділення ознак CNN класифікаторів текстів складається з пар згорткових (convolution) та об’єднувальних (pooling) шарів. Ця частина мережа має зменшувати кількість ознак НТ, присутніх у наборі даних. При цьому можуть створюватися нові функції, які узагальнюють існуючі функції, що містяться в оригінальному наборі функцій НТ.

Згортковий шар є першим шаром, який використовується для виділення різних функцій із вхідних НТ. У цьому шарі виконується математична операція згортки між вхідним НТ і фільтром (ядром). Пересування ядра по вхідному масиву НТ дозволяє сформувати скалярний добуток між ядром і частинами вхідного НТ. Якщо застосувати згортку до кожного фрагменту НТ, то буде отримано результат, що залежить від обраного ядра згортки.

Ядро переміщується по всій області НТ і знаходить певні ознаки, що притаманні конкретному новинному тексту. Наприклад, якщо мережу навчали на множині новин з області актуарної математики, то одне з ядер в процесі навчання може видавати найбільший сигнал в області прогнозу економічних показників трейдингової системи, а інше ядро може виявляти інші ознаки. Якщо розмір ядра недостатній, то воно не зможе виділити потрібні ознаки, якщо ж занадто великий, то збільшується кількість зв'язків між нейронами. Ядро реалізує систему поділюваних ваг, що є однією з головних особливостей CNN. У звичайній багатошаровій ШНМ існує занадто багато синаптичних зв'язків між нейронами, що суттєво уповільнює процес виділення ознак та подальшої класифікації. В мережах CNN узагальнення ваг дозволяє зменшувати число таких зв'язків і знаходити одну і ту саму ознаку по всьому аналізованому НТ.

В CNN результат передається на наступний рівень (рівень шарів об'єднання) після застосування операції згортки для вхідних НТ.

Головною метою шарів об'єднання або субдискретизації (Pooling Layers) є зменшення зв’язків між шарами та отримання можливості їх незалежної роботи на всій карті НТ. Шари об'єднання розміщають одразу після згорткових шарів (виходи згорткових шарів є входами для шарів об'єднання мережі). Операції об'єднання скорочують розміри карт характеристик НТ за рахунок використання деяких функцій для узагальнення частин НТ. Шари об'єднання дозволяють поступово зменшувати розмірності даних та додаткове скорочувати кількість параметрів і складність моделі класифікації НТ. Це, зокрема, зменшує ймовірність виникнення ефекту перенавчання під час налаштування мережі коасифікатора НТ. В узагальненій моделі класифікатора текстів передбачується використання різних методів об'єднання (максимальне об'єднання, середнє об'єднання, стохастичне об'єднання, спектральне об'єднання, об'єднання просторової піраміди, об'єднання по нормі $L2$ тощо). Залежно від методу, який використовується, реалізуються кілька типів операцій об’єднання. Зокрема, операція Max Pooling полягає у визначенні найбільшого елементу з карти функцій. Об'єднувальний шар сприяє поступовому скороченню просторового розміру представлення для зменшення кількості параметрів та об'єму обчислень у мережі, і відтак також для контролю перенавчання. Операція Average Pooling обчислює середнє значення фраментів для вхідного НТ попередньо визначеного розміру. Загальна сума елементів у попередньо визначеному фрагменті визначається як об’єднання сум елементів. Рівень об’єднання зазвичай служить мостом між згортковим рівнем і рівнем класифікації. Ця модель CNN узагальнює ознаки, витягнуті шаром згортки, і допомагає мережам розпізнавати функції незалежно. За допомогою цього також зменшуються обчислення в мережі.

Об'єднувальний шар, як і згортковий також має карти. Одним з важливих завдань цього шару є зменшення розмірності карт попереднього

шару. Якщо на попередній операції згортки вже були виявлені деякі ознаки, то для операції об’єднання повний варіант НТ вже не потрібен (тобто він суттєво ущільнюється).

Повнозв’язний рівень Fully Connected (FC) CNN класифікаторів текстів використовує вихідні дані процесу згортання та передбачає класифікацію НТ на основі ознак, виділених на попередніх етапах. Цей рівень складається з ваг і зміщень разом із нейронами та використовується для з’єднання нейронів між двома різними шарами. Ці шари зазвичай розміщуються перед вихідним шаром і утворюють кілька останніх шарів архітектури CNN.

У цьому випадку дані вхідного НТ з попередніх шарів вирівнюються та подаються на повнозв’язний шар FC. Потім сплощений вектор проходить ще кілька шарів FC, після чого починається процес класифікації НТ. Зазвичай два повністю з’єднані шари працюють краще, ніж один з’єднаний шар.

Повнозв’язний шар моделює складну нелінійну функцію, оптимізація якої сприяє підвищенню якості класифікації аналізованих НТ.

Нейрони кожної карти попереднього об’єднувального шару пов'язані з одним нейроном повнозв’язного шару. Таким чином, число нейронів повнозв’язного шару дорівнює числу карт об’єднувального шару, але при цьому тільки частина нейронів будь-якої з карт об’єднувального шару може бути пов'язана з першим нейроном прихованого шару. Частина, що залишилася, може бути пов'язана з другим нейроном прихованого шару.

Визначення кількісті нейронів у шарі є важливим етапом побудови CNN. На сьогодні не існує універсальних рекомендацій до вибору кількості прихованих шарів та кількості нейронів в них. Їх недостатня кількість не дозволяє мережі ефективно навчатися, а занадто велика кількість значно збільшує час навчання CNN або призводить до її перенавчання. Як було вже відзачено вище, в разі виникнення перенавчання створена модель класифікації починає пояснювати лише приклади з навчальної вибірки,

адаптуючись до них. Разом з тим вона не вчитися класифікувати приклади, які не залучалися до участі в навчанні, що зменшує здатність мережі до узагальнення можливих ситуацій класифікації НТ.

До найбільш важливих завдань побудови ШНМ класифікатора текстів слід віднести вибір функцій активації. Їх вибір залежить від типу ШНМ та особливостей завдання класифікації.

Вони використовуються для вивчення та апроксимації будь-якого виду безперервних і складних зв'язків між змінними мережі.

Зазвичай, в об'єднувальномі шарі застосовується функція активації *ReLU* (англ. Rectified Linear Units – зрізані лінійні вузли). Цей шар застосовує ненасичувальну передавальну функцію $f(x) = \max(0, x)$. Він посилює нелінійні властивості функції ухвалення рішення і мережі в цілому, не зачіпаючи рецептивних полів згорткового шару. Для посилення нелінійності застосовуються й інші функції, наприклад, насичувальні гіперболічний тангенс $f(x) = tanh(x)$, $f(x) = |tanh(x)|$, сигмоїдна функція $f(x) = (1 + e^{-x})^{-1}$ та *Softmax* (нормалізована сігмоїда). Кожна з цих функцій має певні рекомендації до використання: зрізаному лінійному вузлові *ReLU* часто віддають перевагу перед іншими функціями, оскільки він тренує нейронну мережу доволі швидко без втрати точності узагальнення; сигмоїдна функція використовується, зазвичай, для бінарної класифікації, а *Softmax* – для багатокласової класифікації.

Крім стандартних архітектур CNN останнім часом набули поширення нові способи побудови згорткових шарів, щоб підвищити ефективність навчання. Ці архітектури надають загальні архітектурні рекомендації для практиків машинного навчання, які можна адаптувати для вирішення проблем нейромережевої класифікаціх текстів. Ці архітектури можна використовувати як екстрактори функцій для класифікації НТ та ввести їх до склалу узагальненої моделі класифікації НТ.

Для оцінки ефективності фази навчання в ШНМ нейромережевої класифікації НТ в узагальненій моделі використовується функція втрат (loss function).

Оцінювання рівеня втрат (англ. loss layer) мережі за результатами касифікації використовується для оновлення параметрів при її тренуванні. За допомогою попередньо визначеної функції втрат визначається різниця між значеннями тренувальних даних та результатами роботи мережі (прогнозом), після чого при наступному етапі тренування мережі ці дані використовуються для застосування алгоритмів тренування.

Вибір функції втрат є одним з найбільш важливих кроків побудови ШНМ класифікатора, адже ця функція впливає на швидкість та якість тренування мережі, а також на її здатність мережі оцінювати точність отримуваних результатів. Функція втрат є складовою частиною критерія оцінки ефективності роботи ШНМ, адже за її допомогою всі переваги та недоліки мережі мають характеризувати нейромережеву модель таким чином, щоб результати роботи для різних варіантів моделі можливо було оцінити за точністю та повнотою. Зазначимо, що функція втрат має враховувати особливості конкретної задачі класифікації НТ.

Двома головними типами функцій втрат для ШНМ є функції перехресної ентропії (англ. cross-entropy functions) та функції середньоквадратичної похибки (англ. mean-squared error functions).

Проте, для CNN в мультикласових задачах класифікації для аналізу результатів роботи мережі можуть використовуватися імовірності того, що вхідні дані належать кожному з визначених класів. В цьому разі знаходження параметрів відбувається за принципом максимальної правдоподібності і нейромережева модель шляхом наближення до оптильних зачень параметрів намагається наблизити розподіл результатів до розподілу справжніх даних. В цьому випадку критерієм помилки роботи ШНМ буде різниця між імовірностями належності вхідних даних до всіх класів, аналізованих мережею та заданих у навчальній вибірці. Для задач

класифікації зі згортковими мережами використовуються переважно функції перехресної ентропії. Можна показати, що функція середньоквадратичної похибки для задачі класифікації є різновидом функції перехресної ентропії, якщо дані розподілені за нормальним законом.

Функція перехресної ентропії, яка використовується як функція втрат для задачі класифікації, має ще назву функції логарифмічних втрат (англ. log-loss function) та зазвичай застосовується в комбінації з використанням функцій активації Softmax (для багатокласової класифікації) та сігмоїди (для бінарної класифікації) на останньому рівні роботи класифікаційної номережі (для обчислення прогнозів).

Функції активації або активаційні функції (англ. activation functions) використовуються як нелінійна складова частина після згорткових, об'єднувальних (а інколи й інших шарів) нейронних мереж, сприяючи формуванню остаточних вихідних даних.

Вибір та використання функцій активації суттєво впливає на роботу ШНМ, оскільки без їх застосування мережа не мала б змоги навчатися розпізнавати нелінійні закономірності та розподіли. Якщо шари мережі не поєднуються між собою функціями активації, то буде формуватися лінійна залежність між початковим та кінцевим рівнями мережі. За допомогою обгрунтованого вибору активаційної функції можна контролювати вплив на результати окремих нейронів. Зазвичай функції активації зменшують реальні значення даних, щоб запобігти ризику нескінченного зростання параметрів.

Крім того, вони мають бути диференційовані, щоб реалізувати процедури зворотнього поширення помилки або градієнтні методи навчання ШНМ. Правильний вибір функції активації суттєво впливає на задання початкових параметрів мережі перед початком її навчання (наприклад, якщо функція активації не наближує адекватне перетворення поблизу нульової точки, то при заданні випадкових невеликих значень мережа може незадовільно навчатися.

Випрямлена лінійна одиниця (англ. rectified linear unit, ReLU) є функцією активації, що часто застосовується в сучасних ШНМ. Це зумовлено тим, що вона оброблює дані та навчає мережу набагато швидше за інші функції активації, при цьому враховуючи нелінійність. Крім пришвидшення навчання мережі, це й знижує можливість її перенавчання.

Функція ReLU усуває негативні значення з мережі, надаючи їм нульові значення, виконуючи при цьому тотожне перетворення для інших значень. Швидкість роботи цієї функції з апаратної точки зору покращується ще й тим, що вона є звичайним порівнянням значення з нулем (похідна функції ReLU дорівнює 0 для від'ємних значень і 1 для всіх інших, що спрощує реалізацію алгоритму зворотнього поширення помилки). В той же час функція ReLU вимагає наявності хоча б одного додатного початкового значення для ефективного навчання. Відзначимо втім, що активація не всіх початкових значень зазвичай позитивно впливає на роботу ШНМ через ефект розрідження.

Ще двома функціями, які традиційно використовуються для функцій активації в різних архітектурах ШНМ, є гіперболічний тангенс $f(x) = tanh(x)$ та сигмоїдна функція $f(x) = (1 + e^{-x})^{-1}$, яка крім того застосовується як функція активації на останньому рівні мереж в задачах бінарної класифікації, оскільки її область значень знаходиться в інтервалі [0;1], що дозволяє апроксимувати імовірності належності вхідних даних до двох класів:

В задачах багатокласової класифікації на останньому рівні мережі зазвичай використовується функція Softmax, яка враховує всі значення з попереднього рівня мережі та може обрахувати відповідну імовірність для кожного класу. Фактично, функція Softmax нормалізує екпоненційні значення даних з попереднього рівня, тим самим переводячи вхідний масив довільних значень в область [0;1], що використовується для обчислення імовірностей.

Функція Softmax може бути модифікована шляхом заміни бази степені з *e* на інше число, тим самим збільшуючи вагу великих значень на попередньому рівні (при збільшенні бази), або навпаки зрівнюючи вплив різних значень (при зменшенні бази). Крім того, ця функція стійка до збільшення всіх показників з попереднього рівня на постійне значення, що збільшує стійкість результуючих імовірностей належності до класів. Крім настроювання згаданих вище параметрів та функцій ШНМ класифікатора корисним є застосування так званого оптимізатора, який здійснює оптимізацію моделі, координуючи прямий і зворотний градієнти мережі для формування оновлень параметрів, які щоб поліпшити функцію втрат. Для моніторинга тренування і тестування ШНМ класифікації НТ різних типів необхідним є застосування критеріїв оцінювання, що базуються на використанні певних метрик (metrics).

Таким чином, до більшості архітектур нейронних мереж, що враховуються в узагальненій моделі нейромежевої класифікації НТ, належать:

– шари (layers), де, власне, відбувається навчання (обов’язковими є вхідний, прихований і вихідний шари;

– функція втрат (loss function), що використовується для оцінки ефективності фази навчання; оптимізатор (oprimizer), який покращує навчання за допомогою оновлення знань у мережі (з використанням вхідних даних і функції втрат);

– метрики (metrics), що використовуються для моніторинга тренування і тестування нейромережевих моделей класифікаторів НТ.

Характерний фрагмент типової схеми навчання ШНМ, що містить ці частини мережі, наведено на рисунку 2.13.

Побудована нейромережева модель використовується безпосередньо для класифікації поточних НТ (в блоці класифікації НТ) згідно з обраним методом.

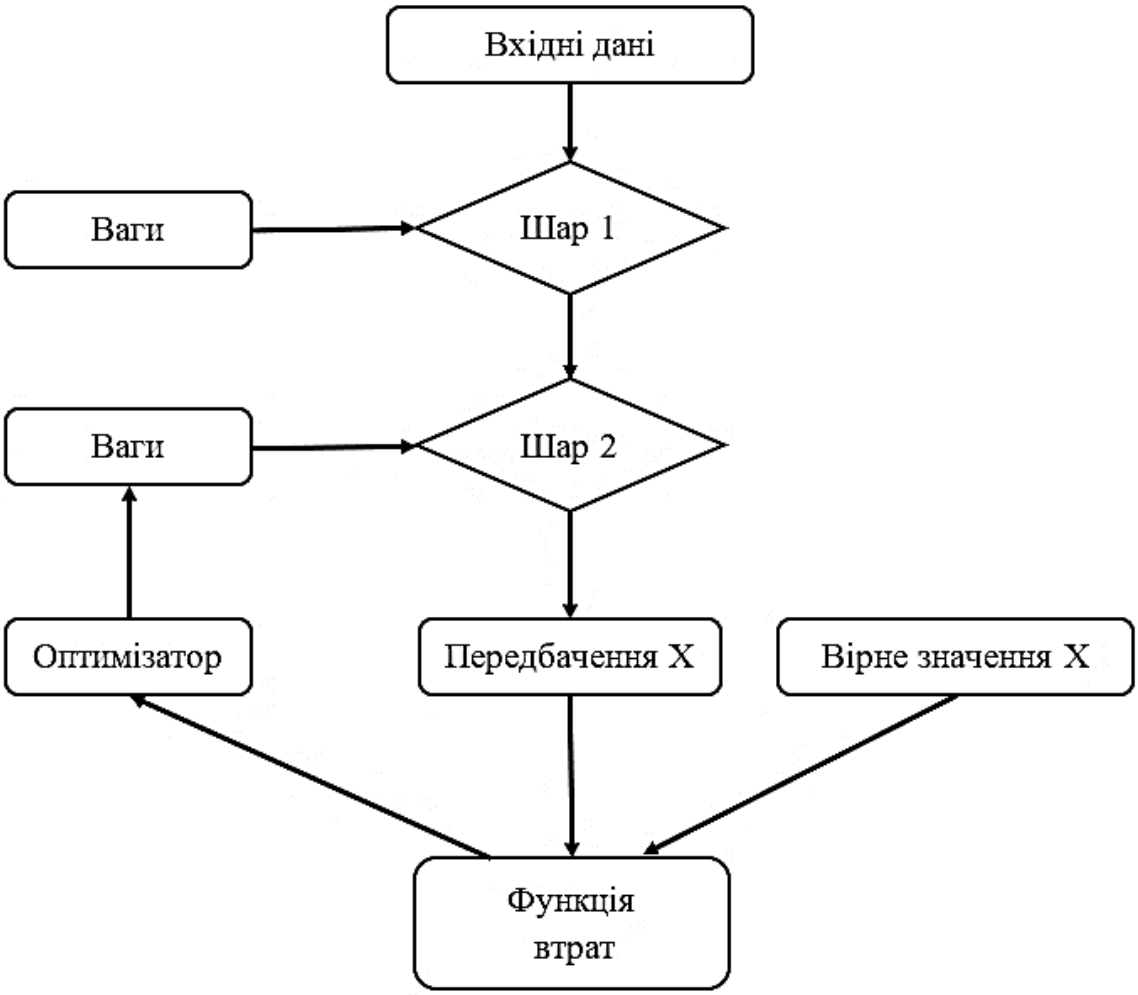

Рисунок 2.13 – Фрагмент типової схеми навчання ШНМ класифікатора

2.6 Оцінка якості та формування результатів класифікації НТ

Оцінка моделі класифікації обєктів різного типу (зокрема, і класифікації НТ) основана на використанні різних метрик оцінки для визначення ефективності роботи класифікаторів. Така оцінка особливо важлива для аналізу можливостей моделі на початкових етапах її дослідження з використанням навчальних, валідаційних та тестових наборів даних (за наявності необхідних датасетів).

Навчальна виборка – це підмножина набору даних, що використовується для побудови прогностичних моделей.

Валідаційний набір – це підмножина набору даних, що використовується для оцінки ефективності моделі, побудованої на етапі навчання. Цей набір забезпечує тестову платформу для точного настроювання параметрів моделі та вибору найбільш ефективної її конфігурації.

Слід зазначити, що інколи алгоритми моделювання класифікаційних мереж не потребують використання валідаційного набору. Тестовий набір – це підмножина набору даних для оцінки продуктивності моделі в процесі її подальшого прогнозування на довільних даних.

Результати класифікації можна визначити наступним чином:

– істинно позитивний (True Positive – TP), якщо класифікатор правильно відніс об'єкт до класу, що розглядається;

– істинно негативний (True Negative – TN), якщо класифікатор вірно стверджує, що об'єкт не належить до класу, який розглядається;

– хибно позитивний (False Positive – FP), якщо класифікатор неправильно відніс об'єкт до класу, що розглядається;

– хибно негативний (False Negative – FN), якщо класифікатор невірно стверджує, що об'єкт не належить до класу, що розглядається.

Ці можливі результати (матриця помилок класифікації) відображені у таблиці 2.2. Наприклад, для бінарної класифікації така матиця формується після тестування моделі на тестових даних з подальшим визначенням для кожного передбачення одного з чотирьох можливих результатів.

Таблиця 2.2 – Матриця помилок класифікації

		Прогнозування класів	
		Negative 0	Positive 1
Типи класів	Negative 0	TN	FP
	Positive 1	FN	TP

Матриця помилок класифікації є таблицею, де рядки відповідають фактичним класам, а стовпці – передбаченим класифікатором класам. За цієї матриці неточностей можна розрахувати різні вживані метрики оцінки класифікаційних моделей.

Розглянемо загальні метрики, що використовуються для оцінки моделей [9].

Однією з найбільш вживаних метрик для оцінки ефективності роботи класифікатора є правильність (accuracy), значення якої визначається наступним відношенням:

$$Accuracy = \frac{P}{N}, \tag{2.12}$$

де $Accuracy$ – правильність;

P– кількість об’єктів, для яких класифікатор сформував правильне рішення;

N – розмір навчальної вибірки.

Недоліком цієї метрики є те, що всі об'єкти вважаються рівноцінними, а це може бути некоректним у зміщених вибірках, де більшість об'єктів належить до одного або декількох класів. В цьому разі класифікатор має більше інформації про класи з великою кількістю об'єктів і приймає для цих класів більш адекватні рішення. При цьому оцінка $Accuracy$ може виявитися високою за результатами тестування, але класифікатор працюватиме незадовільно під час роботи класифікатора з незміщеними класами.

Іншими поширеними оцінками якості класифікатора є точність (precision) і повнота (recall).

Точність (precision) в межах класу визначається відношенням частки об'єктів, що дійсно належать даному класу, до всіх обєктів, які класифікатор відніс до цього класу:

$$Precision = \frac{TP}{TP+FP}, \quad (2.13)$$

де $Precision$ – точність класифікатора;

TP – істинно-позитивні рішення;

FP – хибно-позитивні рішення.

Повнота (recall) класифікації визначається відношенням частки знайдених класифікатором об'єктів, що належать деякому класу, до всіх об'єктів цього класу в тестовій вибірці:

$$Recall = \frac{TP}{TP+FN}, \quad (2.14)$$

де $Recall$ – повнота;

TP – істинно-позитивні рішення;

FN – хибно-негативні рішення.

Використання матриці помилок дозволяє наглядно оцінити роботу класифікатора та отримати інформацію про його точність та повноту для кожного класу у випадку бінарної класифікації або відносно невеликої кількості класів ьакатокдасової класифікації. В цьому разі елементи матриці помилок (точність і повнота) розраховуються таким чином:

$$Precision_c = \frac{A_{c,c}}{\sum_{i=1}^{n} A_{c,i}}, \quad (2.15)$$

$$Recall_c = \frac{A_{c,c}}{\sum_{i=1}^{n} A_{i,c}}. \quad (2.16)$$

де $Precision_c$ – точність для класу c;

A– матриця помилок;

$Recall_c$– повнота для класу c.

Результуючі точність і повнота розраховуються як арифметичне

середнє точності і повноти по всіх класах.

Метрики Precision і Recall не залежать (на відміну від Accuracy) від співвідношення класів і тому використовуються в умовах незбалансованих вибірок. Часто у реальній практиці виникає проблема знаходження оптимального балансу між метриками повноти та точності.

Поширеним підходом до поєднання цих показників є застосування *F*-міри, яка визначається як середнє гармонійне значення точності та повноти моделі:

$$F_\beta = \frac{(1+\beta^2)precision*recall}{(\beta^2*precision)+recall}, \quad (2.17)$$

де F– значення *F*-міри;

β– ваговий коефіцієнт;

$Precision$– розрахована точність класифікатора;

$Recall$– розрахована повнота класифікатора.

Відзначимо, що *F*-міра прагне до нуля, якщо точність або повнота будуть прагнути до нуля.

Коефіцієнт β приймає значення в діапазоні $0 < \beta < 1$, якщо ми надаємо пріоритет точності, або $\beta > 1$, якщо пріоритет надається повноті.

Використання показника F_β зі значеннями $\beta = 2$ та $\beta = 3$ дозволяє в практичних задач класифікації врахувати те, що ціна помилки класифікації може бути різною. Наприклад, показник *F*2 вважає, що значення *recall* є більш важливим за значення *precision*.

Для забезпечення рівнозначності точності і повноти при розрахунку *F*-міри ваговий коефіцієнт β слід прирівняти до одиниці:

$$F = \frac{2*precision*recall}{precision+recall}. \quad (2.18)$$

При β=1 отримуємо збалансовану *F*-міру (її також називають мірою

*F*1 або *F*1-score). Показник *F*1 набуває значення від 0 до 1, де 1 відповідає ідеальному балансу між *precision* і *recall*, а 0 свідчить про найгіршу ефективність класифікатора.

F-міра дає зрозумілу оцінку того, чи відбулися покращення в алгоритмі класифікації і якою є його здатність правильно класифікувати дані. Результуюче значення *F*-міри (формування результатів класифікації) розраховується як середнє арифметичне цієї міри по всіх класах.

Таким чином, *F*-міра є зручним інструментом для порівняння та визначення ефективності класифікаторів.

3 ПРОГРАМНА РЕАЛІЗАЦІЯ ТА РЕЗУЛЬТАТИ МОДЕЛЮВАННЯ БІНАРНИХ ТА БАГАТОКЛАСОВИХ КЛАСИФІКАТОРІВ НОВИННИХ ТЕКСТІВ

3.1 Вибір програмних засобів узагальненої моделі

Для програмної реалізації переважної більшості архітектур ШНМ класифікаторів НТ узагальненої моделі, розглянутих в попередньому розділі, доцільним є застосування мови програмування Python, що містить містить значну кількість корисних бібліотек для роботи з нейронними мережами (зокрема, бібліотеки Keras, TensorFlow та NumPy).

Вхідною базою даних для навчання та тестування класифікаційних мереж текстів можуть бути як загальнодоступні бази текстів (наприклад, imdb, яка вже вбудовані в бібліотеці Keras, так і будь-які новинні ресурси з мережі Інтернет.

Вхідною базою даних для навчання та тестування класифікаційних мереж текстів можуть бути як загальнодоступні бази текстів (наприклад, imdb, яка вже вбудовані в бібліотеці Keras, так і будь-які новинні ресурси з мережі Інтернет.

Практичну реалізацію потрібної нейромережевої системи та обраного типу класифікатора можна розбити на три частини: попередня обробка даних; створення моделі та навчання ШНМ класифікатора; проведення експериментів.

Для створення та тестування відповідних нейромережевих класифікаторів узагальненої моделі будемо використовувати хмарну платформу Google Colab, що дозволяє створювати коди побудови нейромереж та моделі машинного навчання мовами Python або R. Платформа Google Colab надає розробникам готове до використання програмне середовище у веб-браузері [28], [29].

Використання Google Colab є доцільним ще й тому, що воно звільняє розробника від необхідності виконувати створені програми з використанням можливостей його комп'ютера. Це дозволяє виконувати коди/програми, для виконання яких комп'ютеру розробника може не вистачити ресурсів (наприклад, потужності процесора або об'єму пам'яті).

Для використання цієї платформи необхідно її підключити до особистого або бізнес-аккаунту Google, після чого до нього можна отримати доступ з будь-якого браузера (наприклад, Chrome, Explorer або Safari тощо).

Слід також відзначити, що Google Colab не потребує налаштування, а створені користувачем блокноти можуть одночасно редагувати також і члени його команди. Безперечною перевагою є й те, що Google Colab підтримує більшість бібліотек машинного навчання, які можна завантажити у блокнот для подальшого використання.

Розробник (або команда розробників) може виконувати такі дії за допомогою платформи Google Colab:

– писати та виконувати код мовою Python;
– створювати/завантажувати/обмінюватися блокнотами;
– імпортувати/зберігати блокноти з/на Google Drive;
– імпортувати/публікувати блокноти із GitHub;
– імпортувати зовнішні набори даних;
– використовувати можливості бібліотек PyTorch, TensorFlow, Keras, OpenCV;
– користуватися безкоштовним хмарним сервісом із безкоштовним GPU.

Python – мова програмування, яка часто використовується для створення веб-сайтів і програмного забезпечення, автоматизації завдань і аналізу даних. Python є мовою загального призначення, що дозволяє аналітикам даних та іншим фахівцям створювати та досліджувати програми проведення складних статистичних розрахунків, візуалізації даних,

побудови алгоритмів машинного навчання, аналізу даних та виконання інших завдань, пов'язаних з обробкою даних.

Python дозволяє створювати широкий спектр різних візуалізацій даних, таких як лінійні та стовпчасті діаграми, кругові діаграми, гістограми та тривимірні графіки. У Python вбудовано ряд потужних бібліотек, які допомагають програмістам швидше та ефективніше програмувати завдання аналізу даних та машинного навчання, зокрема, Keras та TensorFlow.

Keras є бібліотекою API з відкритим кодом для ШНМ, створеною на базі мови Python. Вона працює поверх таких фреймворків, як Theano і TensorFlow. На сьогодні Keras є однією з найбільш широко використовуваних бібліотек API для розробки і тестування багатошарових ШНМ, що дозволяє створювати шари для нейронних мереж та конфігурувати і досліджувати складні нейромережеві архітектури.

Модель побудови ШНМ з використанням Keras складається з послідовності налаштованих модулів, які можуть бути об'єднані для створення нових моделей. Перевагою цієї модульності є здатність додавати нові функціональні можливості у вигляді окремих модулів. При цьому можливим є об'єднання модулів нейронних шарів, оптимізаторів, схем ініціалізації, функцій активації або схем регуляризації для створення нових модулів тощо. Шаблони визначаються в коді Python, а не в окремих файлах налаштування шаблонів. Бібліотека Keras суттєво полегшує роботу з обчисленнями, дозволяючи використовувати як CPU (центральний процесор), так і GPU (графічний процесор), в якому реалізована технологія Nvidia CUDA. Найчастіше розрахунки на GPU в завданнях машинного навчання гарантують високу продуктивність.

TensorFlow є це бібліотекою Python для швидких числових обчислень та підтримки традиційного машинного нвчання, що створена компанією Google. Вона також виявилася дуже корисною для розробки мереж глибокого навчання. TensorFlow приймає дані у вигляді багатовимірних масивів, іменованих тензорами. Такі масиви дуже зручні під час роботи з

великими обсягами даних. TensorFlow працює на основі графів потоків даних, які мають вузли та ребра. Оскільки механізм виконання має графову форму, доцільно виконувати код TensorFlow на розподіленому кластері комп'ютерів під час використання графічних процесорів.

NumPy (Numerical Python) є бібліотекою Python з відкритим кодом, яка є універсальним стандартом для числових даних при використанні мови Python, а також є ядром наукових екосистем Python і PyData. Бібліотека NumPy містить структури даних багатовимірних масивів і матриць. Вона може бути використана для виконання широкого спектру математичних операцій над масивами. Ця бібліотека додає структури даних в Python, які гарантують ефективність роботи з масивами і матрицями, а також дає доступ до бібліотеки високорівневих математичних функцій, які працюють з цими масивами та матрицями.

3.2 Реалізація бінарних класифікаторів НТ

В попередньому розділі було відзначено, що бінарні класифікатори в узагальненій моделі можуть застосовувались як для виявлення фейкової складової в НТ, так і для класифікації НТ, що не містять фейків.

В рамках кваліфікаційної роботи неможливо описати всі типи бінарних класифікаторів узагальненої моделі. Тому в подальшому розглянемо приклад та дослідження побудови багатошарової нейронної мережі бінарного класифікатора НТ.

Як датасет для формування навчальних наборів використаємо набір даних IMDB Dataset of 50K Movie Reviews [30], який містить 50 тисяч новинних текстів, що є відгуками на фільми. Цей датасет охоплює 25000 відгуків для навчання та тестування (50% позитивних і 50% негативних відгуків). Завданням класифікатора є визначення за текстом відгуку, чи є він позитивним.

Для програмування та реалізації такого класифікатора були обрані мова Python та хмарна бібліотека Google Colaboratory.

Підключимо Google диск до Colab та завантажимо набір даних IMDB Dataset (рисунок 3.1).

```
[ ] import pandas as pd
    import numpy as np

    df = pd.read_csv('drive/MyDrive/Collab/movie_data.csv', engine='python', quoting = 1, sep=',')
    df.head()
```

	review	sentiment
0	In 1974, the teenager Martha Moxley (Maggie Gr...	1
1	OK... so... I really like Kris Kristofferson a...	0
2	***SPOILER*** Do not read this, if you think a...	0
3	hi for all the people who have seen this wonde...	1
4	I recently bought the DVD, forgetting just how...	0

Рисунок 3.1 – Фрагмент відгуків з набору даних IMDB Dataset

Імпортуємо необхідні модулі для роботи із нейронними мережами, в тому числі:

– модуль array з бібліотеки NumPy (використовується для конвертації датасету в масиви типу NumPy);

– модуль one_hot з бібліотеки Keras (для кодування слів масивами цілих чисел);

– модуль pad_sequences (для вирівнювання векторів речень до однакової довжини);

– модуль pandas (для завантаження файлу із .csv);

– модуль Sequential (для побудови послідовної моделі ШНМ);

– модуль Dense (для додавання повнозв'язних шарів);

– модуль Flatten (для зміни розмірностей масиву);

– модуль Embedding (для реалізації шару word embedding).

Створюємо змінні для зберігання оглядів фільмів та міток

Використаємо вбудовану функцію зі sklearn для розбиття датасету на навчальний (80%) та тестовий набори (20%) (рисунок 3.2).

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test=train_test_split(docs,labels,test_size=0.20)
```

Рисунок 3.2 – Розбиття датасету на навчальний та тестовий набори

Для реалізації процедур машинного навчання нейромережевого класифікатора перетворимо категорійні дані в числові та конвертуємо кожне речення в набір чисел із використанням функції one_hot. Ця функція має наступні аргументи: текст для конвертації; розмір словника; фільтр, який вилучає знаки пунктуації; булеве значення, що вказує, чи потрібно трансформувати великі літери в малі; split (розділовий знак між словами).

Традиційно ШНМ будуються таким чином, що всі вхідні вектори (а також вектори, які потім будуть використовуватися для класифікації) мають однакову довжину. Функція pad_sequences дозволяє вирівняти всі чисельні вектори речень в тренувальному наборі до однакової довжини.

Побудуємо модель бінарного класифікатора на базі багатошарової мережі прямого розповсюдження (з використанням моделі Sequential бібліотеки) з додатковим шар вбудовування (Embedding0).

Перший параметр у шарі вбудовування задає розмір словника або загальну кількість унікальних слів у корпусі. Другий та третій параметри є кількістю вимірів кожного вектора слів та довжиною вхідного речення відповідно. Результатом вбудовування слів є двовимірний вектор, у якому слова представлені рядками, а відповідні їм розміри стовпцями. Отриманий

вектор embedding визначає унікальність кожного текстового речення та близькість порівнюваних речень.

Для з'єднання шару вбудовування слів зі щільно пов'язаним шаром, потрібно двовимірний вихід з шару Embeddings трансформувати в одновимірний масив (за допомогою Flatten) та передати результат на два послідовні повнозв'язні шари Dense із відповідними функціями активації Relu та Sigmoid. Розташуємо між ними шар відкидання частини нейронів (Dropout) для запобігання перенавчанню.

Далі модель компілюється (compile) з методом оптимізації (обрано метод Adam), функцією втрат (binary_crossentropy) та метрикою оцінки якості (metrics). В наступному рядку вказано метод summary, який друкує структуру всієї моделі. В рядку запуску навчання (fit) вказується навчальна вибірка з двох частин X_train, y_train, кількість ітерацій, а також коефіцієнт розбиття між навчальною та валідаційною вибіркою (рисунок 3.3).

```
model=Sequential()
model.add(Embedding(vocab_size,32,input_length=max_length))
model.add(Flatten())
model.add(Dense(32,activation='relu'))
model.add(Dense(20,activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(1,activation='sigmoid'))

model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['acc'])

print(model.summary())

history=model.fit(X_train,y_train,epochs=10,verbose=1,validation_split=0.4, batch_size=500)
```

Рисунок 3.3 – Задання параметрів ШНМ бінарного класифікатора

Після завершення процедури навчання за допомогою функції evaluate можна оцінити якість моделі на навчальній та тестовій вибірці (рисунок 3.4, рисунок 3.5).

```
[99] loss,accuracy=model.evaluate(X_train,y_train,verbose=1)
     print('Training accuracy is {}'.format(accuracy*100))

     1250/1250 [==============================] - 12s 9ms/step - loss: 0.2113 - acc: 0.9495
     Training accuracy is 94.94749903678894
```

Рисунок 3.4 – Оцінка якості моделі на навчальній вибірці

```
[100] loss,accuracy=model.evaluate(X_test,y_test)
      print('Testing accuracy is {}'.format(accuracy*100))

      313/313 [==============================] - 3s 9ms/step - loss: 0.5075 - acc: 0.8778
      Testing accuracy is 87.77999877929688
```

Рисунок 3.5 – Оцінка якості моделі на тестовій вибірці

Точність (accuracy) навчальної вибірки склала майже 95%, а як ми бачимо точність на тестовій вибірці є дещо гіршою, ніж на навчальній, та складає близько 88%.

Порівняємо результати класифікації побудованої нейронної мережі з результатами досліджень, наведеними в [31] для аналогічного набору даних IMDB Dataset. Точність класифікації там склала: для наївного байєсова класифікатора – 81%, для класифікатора SVM – 82,9%, для класифікатора CNN – 87,7%, для класифікатора MLP – 86,74%, для класифікатора LSTM – 86,64% відповідно. Таким чином, точність бінарної класифікації з використанням розробленої нейронної мережі виявилась вищим за точність класифікації текстових даних з з використанням інших мереж.

На рисунку 3.6 наведено графіки навчання для аналізу та візуалізації процесу навчання розробленої мережі бінарної класифікації.

На графіку праворуч відображені значення помилки навчання, а на графіку ліворуч – точність навчання для окремих ітерацій навчальної та валідаційної виборок.

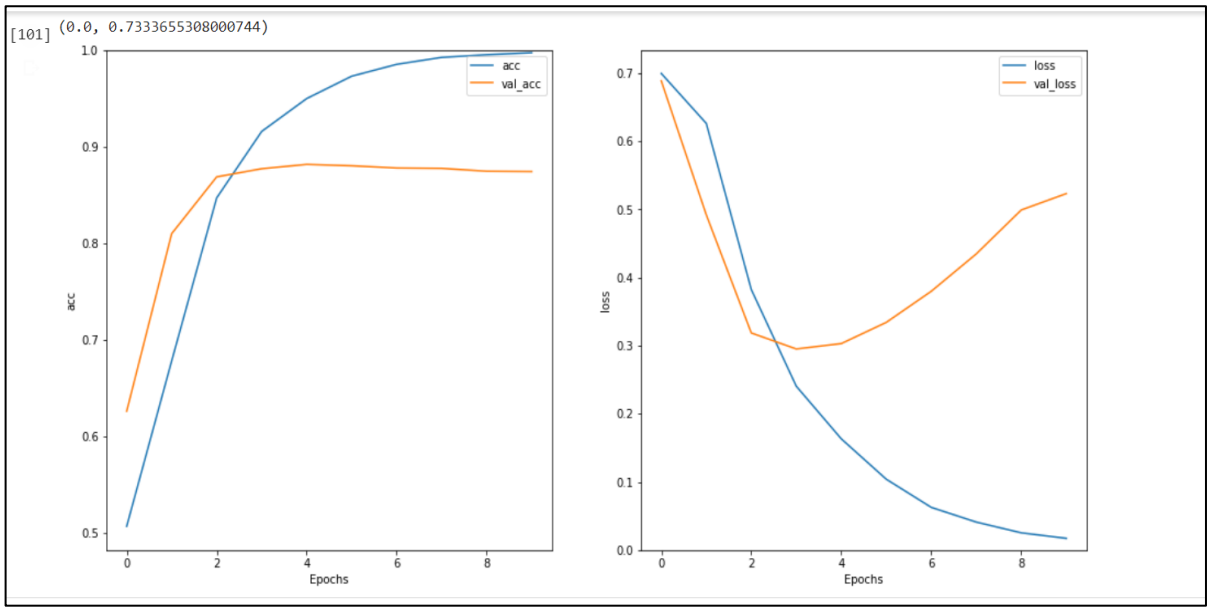

Рисунок 3.6 – Графіки навчання бінарного класифікатора

Відхилення тренувального набору зменшується з кожною ітерацією (епохою), а точність навчання збільшується з кожною епохою. Це очікувано при використанні оптимізації градієнтного спуску — вона повинна мінімізувати значення помилки на кожній ітерації. Але це не стосується похибки і точності для валідаційної. Це є прикладом перенавчання (overfitting): модель працює набагато краще з навчальними даними, ніж з даними, яких вона ніколи не бачила. В процесі такого навчання з певного моменту (зростання сумарної помилки val_loss) модель починає надмірно оптимізувати навчальний набір, але не узагальнюється для довільних векторів. У нашому випадку щоб запобігти такому перенавчанню, ми зупинимо навчання вчасно, а також спробуємо додати ще один шар відкидання певної частини нейронів (Dropout) після першого повнозв'язного Dense шару.

Була також досліджена процедура побудови ШНМ для бінарної класифікації україномовних заголовків статей. Як датасет використовувалися статті з сайту tsn.ua.

Для розробки алгоритму класифікації статей по категоріям (політика, спорт, економіка тощо) було здійснено парсінг сайту за допомогою мови Python з використанням бібліотек beautifulsoup та requests.

Завданням бінарної класифікації визначимо віднесення (за аналізом заголовку) належності нової статті нашій рубриці (класу) «Політичні новини» чи ні.

Загальна кількість новин початкового з набору даних складала 24900 (з них 4220 – політичні новини).

Після завантаження необхідних бібліотеки та розбиття датасет на навчальний (80%) та тестовий набори (20%) була побудована та навчена (за аналогією з попереднім прикладом) багатошарова мережа бінарної класифікації НТ (рисунок 3.7).

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding (Embedding)       (None, 33, 32)            1436160

 flatten (Flatten)           (None, 1056)              0

 dense (Dense)               (None, 32)                33824

 dense_1 (Dense)             (None, 20)                660

 dropout (Dropout)           (None, 20)                0

 dense_2 (Dense)             (None, 1)                 21

=================================================================
Total params: 1,470,665
Trainable params: 1,470,665
Non-trainable params: 0
_________________________________________________________________
None
Epoch 1/50
372/372 [==============================] - 7s 17ms/step - loss: 0.4826 - acc: 0.8298 - val_loss: 0.4444 - val_acc: 0.8372
Epoch 2/50
372/372 [==============================] - 6s 16ms/step - loss: 0.3626 - acc: 0.8486 - val_loss: 0.5373 - val_acc: 0.7787
Epoch 3/50
372/372 [==============================] - 6s 17ms/step - loss: 0.0845 - acc: 0.9695 - val_loss: 0.8524 - val_acc: 0.7191
Epoch 4/50
372/372 [==============================] - 6s 16ms/step - loss: 0.0264 - acc: 0.9865 - val_loss: 1.1161 - val_acc: 0.7685
Epoch 5/50
```

Рисунок 3.7 – Побудова та навчання моделі бінарної класифікації україномовних новин

Результати оцінки якості моделі на навчальній та тестовій вибірках наведено на рисунок 3.8.

```
620/620 [==============================] - 1s 2ms/step - loss: 0.4384 - acc: 0.8339
Training accuracy is 83.38885307312012
155/155 [==============================] - 0s 2ms/step - loss: 0.4571 - acc: 0.8294
Testing accuracy is 82.93963074684143
```

Рисунок 3.8 – Оцінка якості моделі на навчальній та тестовій вибірках

Точність побудованої моделі є прийнятною для класу подібних класифікаторів.

3.3 Багатокласова класифікація новинних текстів

Розглянемо приклад побудови багатошарової нейронної мережі багатокласового класифікатора НТ та її навчання для вирішення задачі багатокласової класифікації новинних даних.

Як датасет для формування навчальних наборів використаємо набір набір статей BBC news [32], який містить близько тисячі новинних текстів. Завданням класифікатора є визначення за НТ, до якого з 5 класів (розваги, бізнес, політика, спорт, техніка) відноситься новина.

Для програмування та реалізації такого класифікатора були обрані мова Python та хмарна бібліотека Google Colaboratory.

Підключимо Google диск до Colab та завантажимо набір даних BBC news Dataset (рисунок 3.9).

Імпортуємо бібліотеку ntlk і функцію stopwords (визначимо список кажемо стоп-слів для англійської мови). Імпортуємо необхідні модулі для роботи з ШНМ (рисунок 3.10).

```
[ ] from google.colab import drive
    drive.mount('/content/drive')

    Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).
```

```
import pandas as pd
import numpy as np

df = pd.read_csv('drive/MyDrive/Collab/BBC News Train - BBC News Train.csv', engine='python', quoting = 1, sep=',')
df.head()
```

	Category	Text
0	business	worldcom ex-boss launches defence lawyers defe...
1	business	german business confidence slides german busin...
2	business	bbc poll indicates economic gloom citizens in ...
3	tech	lifestyle governs mobile choice faster bett...
4	business	enron bosses in $168m payout eighteen former e...

Рисунок 3.9 – Фрагмент з набору даних IMDB Dataset

```
import tensorflow as tf
from keras.preprocessing.text import Tokenizer
from keras.preprocessing.sequence import pad_sequences
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import Flatten
from keras.layers import Dropout
from keras.layers import Activation
from keras.layers import Embedding
from keras.layers import Bidirectional
from keras.layers.embeddings import Embedding
```

Рисунок 3.10 – Імпорт модулів для роботи з ШНМ класифікатора

Задамо гіперпараметри, необхідні для побудови і тренування моделі (рисунок 3.11).

```
[ ] vocab_size = 5000
    embedding_dim = 64
    max_length = 200
    trunc_type = 'post'
    padding_type = 'post'
    oov_tok = '<OOV>' # OOV = Out of Vocabulary
    training_portion = .8
```

Рисунок 3.11 – Задання гіперпараметрів моделі

Розбиваємо датасет на навчальну та валідаційну вибірки (80% НТ для навчальної вибірки і 20% для перевірки). Здійснюємо токенизацію НТ і перетворюємо речення на послідовність слів.

На рисунку 3.12 та 3.13 наведено процедури побудови, компіляції, та навчання ШНМ багатокласового класифікатора НТ.

```
[ ] model = Sequential()

    model.add(Embedding(vocab_size, embedding_dim))
    model.add(Dropout(0.5))
    model.add(Bidirectional(LSTM(embedding_dim)))
    model.add(Dense(6, activation='softmax'))

    model.summary()
```

```
Model: "sequential_4"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding_4 (Embedding)     (None, None, 64)          320000

 dropout_4 (Dropout)         (None, None, 64)          0

 bidirectional_4 (Bidirectio  (None, 128)              66048
 nal)

 dense_4 (Dense)             (None, 6)                 774

=================================================================
Total params: 386,822
Trainable params: 386,822
Non-trainable params: 0
_________________________________________________________________
```

Рисунок 3.12 – Побудова моделі ШНМ багатокласового класифікатора НТ

```
[ ] num_epochs = 10
    history = model.fit(train_padded, training_label_seq, epochs=num_epochs, validation_data=(validation_padded, validation_label_seq), verbose=2)

Epoch 1/10
38/38 - 14s - loss: 1.6632 - accuracy: 0.2559 - val_loss: 1.5515 - val_accuracy: 0.3054 - 14s/epoch - 373ms/step
Epoch 2/10
38/38 - 9s - loss: 1.3949 - accuracy: 0.4404 - val_loss: 1.5262 - val_accuracy: 0.4899 - 9s/epoch - 233ms/step
Epoch 3/10
38/38 - 9s - loss: 0.9753 - accuracy: 0.7114 - val_loss: 0.6367 - val_accuracy: 0.8188 - 9s/epoch - 248ms/step
Epoch 4/10
38/38 - 9s - loss: 0.5126 - accuracy: 0.8406 - val_loss: 0.5819 - val_accuracy: 0.8289 - 9s/epoch - 229ms/step
Epoch 5/10
38/38 - 9s - loss: 0.2860 - accuracy: 0.9161 - val_loss: 0.3767 - val_accuracy: 0.9060 - 9s/epoch - 233ms/step
Epoch 6/10
38/38 - 9s - loss: 0.1893 - accuracy: 0.9480 - val_loss: 0.3583 - val_accuracy: 0.8926 - 9s/epoch - 230ms/step
Epoch 7/10
38/38 - 9s - loss: 0.1551 - accuracy: 0.9673 - val_loss: 0.2598 - val_accuracy: 0.9161 - 9s/epoch - 233ms/step
Epoch 8/10
38/38 - 9s - loss: 0.0712 - accuracy: 0.9924 - val_loss: 0.2127 - val_accuracy: 0.9262 - 9s/epoch - 234ms/step
Epoch 9/10
38/38 - 9s - loss: 0.0520 - accuracy: 0.9908 - val_loss: 0.1901 - val_accuracy: 0.9463 - 9s/epoch - 233ms/step
Epoch 10/10
38/38 - 9s - loss: 0.0308 - accuracy: 0.9975 - val_loss: 0.1743 - val_accuracy: 0.9362 - 9s/epoch - 234ms/step
```

Рисунок 3.13 – Тренування моделі

На рисунках 3.14 та 3.15 наведено приклади класифікації новинних статей (перша стаття відноситься до класу «бізнес», а друга – до класу «розваги»).

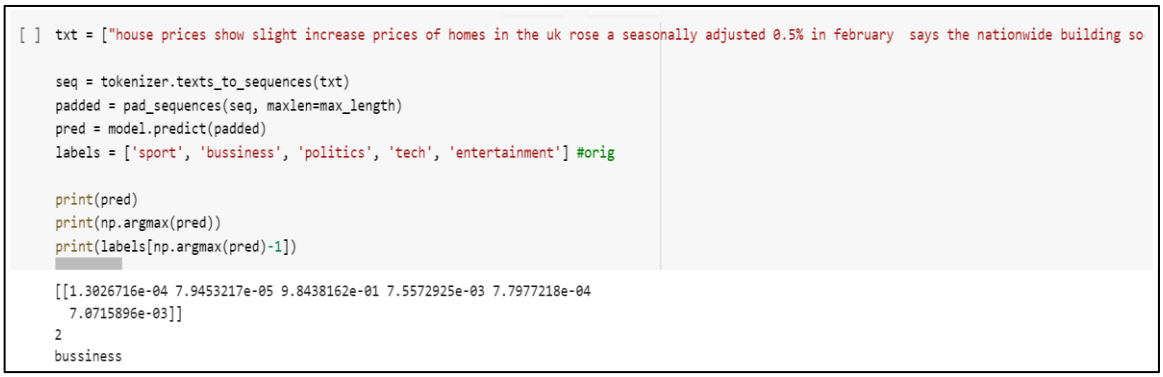

Рисунок 3.14 – Приклад класифікації НТ (1)

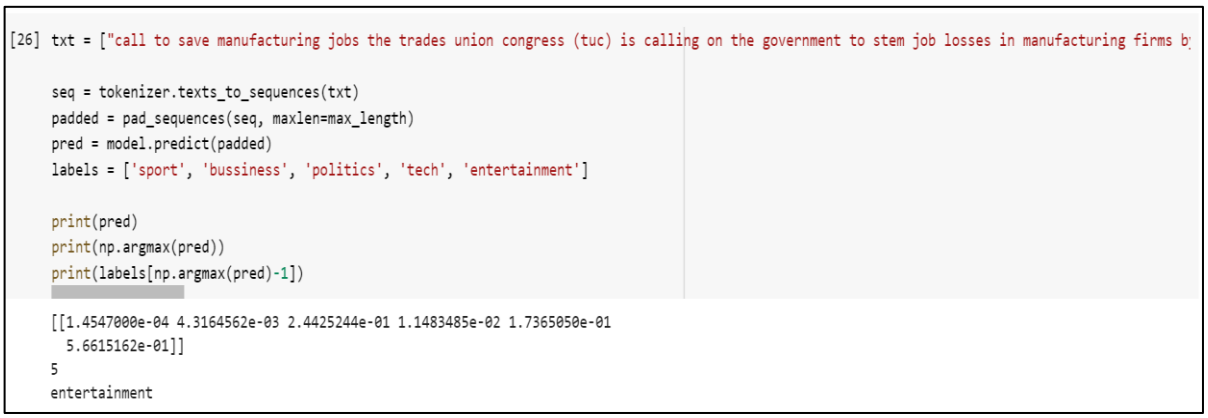

Рисунок 3.15 – Приклад класифікації НТ (2)

Також була розглянута задача багатокласової класифікації текстових новин українською мовою. Датасет містив близько 50000 описів НТ, що включають назву новини, текст новини та ідентифікатор джерела (одного з 7 новинних сайтів: BBC News Україна, НВ, Українська правда, Економічна правда, Європейська правда, Українська правда Життя та Уніан). 20% описів було використано як тестову вибірку, 20% – як валідаційну, відповідно, розмір тренувальної вибірки склав 60%.

Побудуємо багатошарову нейронну мережу для класифікації джерел новин (7 класів) за повним текстом новини. В якості словника було використано повний набір унікальних слів, які зустрічаються в тренувальному наборі (майже мільйон слів). Кожен вхідний текст було закодовано векторами довжиною 1000 значень, які подавалися на вхід першого шару (Embeddings) ШНМ класифікатора. Цей шар ініціалізується випадковими вагами, вивчає вбудовування для всіх слів у навчальному наборі даних та повертає навчений вектор вбудовувань фіксованої довжини (в нашому випадку було використано 128 значень).

Після цього було застосовано повнозв'язний шар, який містить 32 нейрони, шар відкидання нейронів (Dropout) та шар із виходами, який містить 7 нейронів по кількості класів. Всі модель загалом містить близько 132 мільйони параметрів, більшість з яких залежить від розміру словника.

Для реалізації багатокласового класифікатора джерел новин було використано алгоритм оптимізації Adam (adaptive moment estimation). Крім того, з бібліотеки Keras були імпортовані такі компоненти: шари LSTM, Dense, Embedding, активаційна функція та модуль sequence для роботи з даними.

Результати навчання представлені на рисунку 3.16. В результаті навчання середня точність моделі на тестовій вибірці становила близько 84%. Ця мережа схильна до перенавчання, тож навчання зупиняється при збільшенні помилки на валідаційній вибірці.

Підвищення точності досягнути (для цього ж датасету) з використанням інших конфігурацій нейронної мережі багатошарового класифікатора (зокрема, з використанням інших засобів попередньої обробки НТ).

Можна зробити висновок, що перший тренувальний шар Embedding перенасичує мережу параметрами, кількість яких залежить від розміру словника, максимальної довжини вхідного вектору та вектору вбудовувань, що призводить до перенавчання. Можливо, застосування інших методів та

моделей вбудовування слів (замість використання шару Embedding) буде більш ефективним. Також треба врахувати, що набір даних є частково незбалансованим, що також створює додаткові труднощі.

```
Epoch 1/8
2023-10-13 17:56:16.000269: I tensorflow/compiler/xla/service/service.cc:168] XLA service 0x56c616f10af0 initialized for platf
2023-10-13 17:56:16.000321: I tensorflow/compiler/xla/service/service.cc:176]   StreamExecutor device (0): Tesla T4, Compute C
2023-10-13 17:56:16.134806: I tensorflow/compiler/mlir/tensorflow/utils/dump_mlir_util.cc:255] disabling MLIR crash reproducer
2023-10-13 17:56:16.654137: I tensorflow/compiler/xla/stream_executor/cuda/cuda_dnn.cc:432] Loaded cuDNN version 8900
2023-10-13 17:56:17.011790: I ./tensorflow/compiler/jit/device_compiler.h:186] Compiled cluster using XLA!  This line is logge
248/248 [==============================] - ETA: 0s - loss: 1.7433 - acc: 0.3097/usr/local/lib/python3.10/dist-packages/keras/s
  saving_api.save_model(
248/248 [==============================] - 75s 281ms/step - loss: 1.7433 - acc: 0.3097 - val_loss: 1.4206 - val_acc: 0.4237
Epoch 2/8
248/248 [==============================] - 65s 262ms/step - loss: 1.0263 - acc: 0.6215 - val_loss: 0.5619 - val_acc: 0.8105
Epoch 3/8
248/248 [==============================] - 59s 236ms/step - loss: 0.3586 - acc: 0.8833 - val_loss: 0.5005 - val_acc: 0.8412
Epoch 4/8
248/248 [==============================] - 63s 255ms/step - loss: 0.1502 - acc: 0.9546 - val_loss: 0.4980 - val_acc: 0.8524
Epoch 5/8
248/248 [==============================] - 47s 191ms/step - loss: 0.0862 - acc: 0.9742 - val_loss: 0.5458 - val_acc: 0.8567
Epoch 6/8
248/248 [==============================] - 47s 189ms/step - loss: 0.0637 - acc: 0.9802 - val_loss: 0.6248 - val_acc: 0.8489
Epoch 7/8
248/248 [==============================] - 44s 177ms/step - loss: 0.0560 - acc: 0.9821 - val_loss: 0.6458 - val_acc: 0.8471
Epoch 8/8
248/248 [==============================] - 41s 164ms/step - loss: 0.0506 - acc: 0.9844 - val_loss: 0.6867 - val_acc: 0.8544
1239/1239 [==============================] - 3s 2ms/step - loss: 0.1390 - acc: 0.9706
Training Accuracy is 97.05748558044434
310/310 [==============================] - 1s 2ms/step - loss: 0.6945 - acc: 0.8482
Testing Accuracy is 84.81881618499756
```

Рисунок 3.16 – Результати навчання нейромережевого класифікатора новинних текстів за джерелами новин

Програмний код побудови нейронної мережі для багатокласової класифікації джерел новин наведено в Додатку А.

Разом із тим, підібрати гіперпараметри для мережі, яка буде здатна класифікувати новини точніше, набагато важче.

Під час додаткових експериментів була зроблена спроба покращити якість навчання мережі багатокласового класифікатора НТ шляхом зміни таких її параметрів як кількість епох навчання та кількість нейронів в LSTM шарі. Замість стандартної кількості епох навчання (7 епох для 7 класів) аналізувався процес навчання мережі протягом 6, 12 та 16 епох, щоб визначити, коли починається процес перенавчання.

Результати цього експерименту наведено в таблиці 3.1

Таблиця 3.1 – Вплив кількості епох навчання на якість класифікації

Кількість епох навчання	Точність класифікації
6	83, 8%
12	84, 2%
16	82, 9%

Згідно з отриманими результатами, точність класифікації була найвищою десь на 12-й епосі, а далі починала зменшуватися внаслідок ефекту перенавчання.

Крім того, було проаналізовано вплив кількості нейронів в шарі LSTM на якість навчання. Результати цього експерименту наведено в таблиці 3.2.

Таблиця 3.2 – Вплив кількості нейронів в шарі LSTM на якість класифікації

Кількість нейронів в шарі LSTM	Точність класифікації
30	84, 3%
50	84, 8%
100	85, 1%
120	84%

Згідно з отриманими результатами, точність класифікації несуттєво зростає зі збільшенням кількості нейронів в шарі LSTM, досягає максимального значення десь для 100 нейронів, а потім починає зменшуватися.

Таким чином, оптимальні значення гіперпараметрів класифікатора новин доцільно визначати шляхом експериментальних досліджень.

ВИСНОВКИ

До найбільш важливих завдань обробки текстової інформації слід віднести класифікацію текстів, їх попередню обробку, семантичне анотування текстових документів, пошук та порівняння тематичних текстів, автоматичний переклад тощо. Всі ці завдання у певній мірі пов’язані з класифікацією тексту.

Створюючи текстові класифікатори, можна автоматично структурувати існуючі різновиди текстових новинних документів.

На сьогодні існує багато напрямів практичного застосування методів обробки та класифікації новинних текстів з використанням засобів штучного інтелекту (зокрема, спроба відрізняти надійні новини від фейкових новин).

В кваліфікаційній роботі досліджені важливі аспекти аналізу цієї проблеми, розробки перспективних методів обробки текстів та їх практичного застосування у деяких предметних галузях.

Для цього в пропонованому дослідженні вирішені такі завдання:

– аналіз існуючих типів новинних текстів та їх особливостей;

– аналіз існуючих методів попередньої обробки новинних текстів;

– дослідження засобів класифікації новинних текстів з використанням машинного навчання;

– розроблення узагальненої моделі нейромережевої класифікації новинних текстів;

– програмна реалізація та тестування розробленої моделі на прикладах бінарної та багатокласової класифікації новинних текстів.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Вавіленкова А. Теоретичні основи аналізу електронних текстів: монографія. Київ : ТОВ «СІК ГРУП УКРАЇНА», 2016. 192 с. URL: https://dspace.nau.edu.ua/bitstream/NAU/42695/10/All.pdf (дата звернення: 10.05.2024).

2. Доманецька І.М., Федусенко О.В., Хроленко В.М. Нейромережеві технології опрацювання природномовних текстів в адаптивних системах навчання. *Штучний інтелект*. 2017. № 3–4. С. 24–31.

3. Дудник М.П., Удовенко С.Г., Чала Л.Е., Соколовська М.М. Нейромережева технологія багатомовної класифікації електронних текстів. *Біоніка интелекту*. 2021. Вип. 2 (97). С. 3–12.

4. Феній Н.С., Грицюк Ю І.. Автоматизація процесу класифікації текстових новин з інтернет-сайтів методами нейронної мережі. *Науковий вісник НЛТУ України*. 2020. т. 30, № 4. С. 123–132.

5. Берко А.Ю. Система контент-моніторингу новинних інтернет-ресурсів. *Вісник Національного університету "Львівська політехніка"*. 2011. № 699: Інформаційні системи та мережі. С. 13–21

6. Ротон Н. Структурно-функціональні особливості текстів засобів масової комунікації. *Гуманітарна освіта в технічних вищих навчальних закладах*. 2017. № 36. С. 56–61

7. Conneau A., Khandelwal K., Goyal N., Chaudhary V., Wenzek G., Guzmán F., Grave E., Ott M., Zettlemoyer L., Stoyanov V. Unsupervised Cross-lingual Representation Learning at Scale, 8 Apr, 2020, URL: https://arxiv.org/pdf/1911.02116.pdf (last accessed: 15.05.2024).

8. Shmatkov D., Gorokhovatskyi O., Vnukova N. Elaborative Trademark Similarity Evaluation Using Goods and Services Automated Comparison. *In CEUR Workshop Proceedings: Computational Linguistics and Intelligent Systems 2023* (COLINS 2023). Vol. 3403. P. 293–308.

9. Udovenko S., Chala L., Dudnyk M. Research of neural network technologies for automated classification of unstructured texts. *Proceedings of the XI International Scientific and Practical Conference «Innovative solutions to modern scientific challenges»* (Februari 21 – 23, 2024). Zagreb, Croatia. 2024. P. 88–91.

10. Devlin J., Chang M.-W., Lee K., Toutanova K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, 11 Oct, 2018, URL: https://arxiv.org/pdf/1810.04805.pdf (last accessed: 15.05.2024).

11. Гущин І., Сич Д. Аналіз впливу попередньої обробки тексту на результати текстової класифікації. *Молодий вчений.* 2018. Т. 10, № 26. С. 264–266. URL: http://molodyvcheny.in.ua/files/journal/2018/10/63.pdf (last accessed: 15.05.2024).

12. Read and send messages. Manage drafts and attachments. Set up push notifications and manage settings. *Gmail for Developers*: вебсайт. URL: https://developers.google.com/gmail/api (last accessed: 15.05.2024).

13. International Workshop In Applications of Semantic Web technologies for E-Learning (SW-EL). URL: http://www.win.tue.nl/SW-EL (last accessed: 15.05.2024).

14. Yang Y., Cer D., Amin A., Guo M., Law J., Constant N., Hernandez Abrego G., Yuan S., Tar C., Yun-Hsuan S., Strope B., Kurzweil R. Multilingual Universal Sentence Encoder for Semantic Retrieval, 9 Jul, 2019, URL: https://arxiv.org/pdf/1907.04307.pdf (last accessed: 15.05.2024).

15. A Guide on Word Embeddings in NLP. URL: https://www.turing.com/kb/guide-on-word-embeddings-in-nlp (last accessed: 15.05.2024).

16. A Deep Learning based Approach to Reduced Order Modeling for Turbulent Flow Control using LSTM Neural Networks. URL: https://arxiv.org/abs/1804.09269 (last accessed: 15.05.2024).

17. Understanding LSTM Networks. *Colah`s Blog*: вебсайт. URL: http://colah.github.io/posts/2015-08-Understanding-LSTMs (last accessed: 15.05.2024).

18. Deep Learning | Introduction to Long Short Term Memory. URL: https://www.geeksforgeeks.org/deep-learning-introduction-to-long-short-term-memory/ (last accessed: 15.05.2024).

19. Rahul Dey, Fathi M. Salem. Gate-Variants of Gated Recurrent Unit Neural Networks : East Lansing : Michigan State University, 2017. 5 с.

20. Stoyanov V., Necip F. A., Under the hood: Multilingual embeddings, January 24, 2018, URL: https://ai.facebook.com/blog/under-the-hood-multilingual-embeddings/ (last accessed: 15.05.2024).

21. Інформаційні технології та системи: монографія / Удовенко С.Г. Розділ 8. Класифікація електронних науково-технічних текстів в інформаційно-пошукових системах // С.Г. Удовенко, Л.Е. Чала. Х.: ФОП Бровін О.В., 2019. С.108–123.

22. Vaswani A., Shazeer N., Parmar N., Uszkoreit J., Jones L., Gomez A. N., Kaiser L., Polosukhin I., Attention Is All You Need, 12 Jun, 2017, URL: https://arxiv.org/pdf/1706.03762.pdf (last accessed: 15.05.2024).

23. Ahmed H., Traore I., Saad S. Detecting opinion spams and fake news using text classification. Security and Privacy. 2017. Т. 1, № 1. С. e9. URL: https://doi.org/10.1002/spy2.9 (last accessed: 15.05.2024).

24. Kai Shu, Limeng Cui, Suhang Wang, Dongwon Lee, Huan Liu. Defend: Explainable fake news detection. International Conference on Knowledge Discovery & Data Mining, 2019, 395-405 с.

25. Staff E. Fake News: True or False Quiz Book. Egmont Books, Limited, 2020. 96 с.

26. Olah C. LSTM. 21 червень, 2017, URL: https://habr.com/ru/company/wunderfund/blog/331310/ (last accessed: 15.05.2024).

27. Understand Random Forest Algorithms With Examples. URL: https://www.analyticsvidhya.com/blog/2021/06/understanding-random-forest/ (last accessed: 15.05.2024).

28. Machine Learning in Python. *Scikit-learn*: вебсайт. URL: https://scikit-learn.org/stable (last accessed: 15.05.2024).

29. Getting Started. PythonTM: вебсайт. URL: https://www.python.org/about (last accessed: 15.05.2024).

30. Misra R. News Category Dataset, 2 Dec, 2018, URL: https://www.kaggle.com/rmisra/news-category-dataset (last accessed: 15.05.2024).

31. Nautiyal D. ML | Underfitting and Overfitting. GeeksforGeeks. URL: https://www.geeksforgeeks.org/underfitting-and-overfitting-in-machine-learning/ (last accessed: 15.05.2024).

32. Dataset 20 Newsgroups. URL: https://www.kaggle.com/datasets/crawford/20-newsgroups (last accessed: 15.05.2024).