Вінницький національний технічний університет
Факультет інформаційних технологій та комп’ютерної інженерії
Кафедра комп’ютерних наук

Пояснювальна записка

до магістерської кваліфікаційної роботи

на тему «Інформаційна технологія класифікації тональності речень»

Виконав: студент 2 курсу,
групи 2КН-18 м
спеціальності 122 «Комп'ютерні науки»
Барановський В. С.

Керівник: к.т.н., доц. Колесницький О.К.

Рецензент: к. т. н., доц. Войтко В. В.

Вінниця
2019

ЗАТВЕРДЖУЮ
Завідувач кафедри____КН____
д.т.н., проф.. Яровий А.А.

_____
(підпис)
“____” _____2019 року

ЗАВДАННЯ

на магістерську кваліфікаційну роботу на здобуття кваліфікації магістра наук зі спеціальності: 122 – «Комп'ютерні науки»

08-22.МКР.024.18.000.ПЗ

Магістранта групи 2КН-18м Барановського Владислава Сергійовича

Тема магістерської кваліфікаційної роботи: «Інформаційна технологія класифікації тональності речень»

Вхідні дані: 1) використання нейронної мережі; 2) вхідна інформація – речення, введене з клавіатури у полі вікна програми, 3) максимальний розмір речення – 50000 знаків, 4) кількість рівнів тональності речення – 3 (позитивне, негативне, нейтральне), 5) обсяг навчальної вибірки - не менше 100, 6) об’єктно – орієнтована мова програмування; 7) операційна система сімейства Windows;.

Короткий зміст частин магістерської кваліфікаційної роботи:

1. Графічна: Граф-схема алгоритму роботи програмного забезпечення класифікації тональності речень, структура нейронної мережі, структура інформаційної технології, стартове вікно програми класифікації тональності речень, робочі вікна програми класифікації тональності речень.
2. Текстова (пояснювальна записка): вступ, аналіз предметної області класифікації тональності речень, розробка інформаційної технології класифікації тональності речень на основі нейронної мережі, програмна реалізація інформаційної технології класифікації тональності речень на основі нейронної мережі, економічна частина, висновки, перелік використаних джерел, додатки.

КАЛЕНДАРНИЙ ПЛАН ВИКОНАННЯ МКР

№ етапу	Назва етапу	Термін виконання		Очікувані результати
		початок	кінець	
1	Аналіз сучасного рівня інформаційних технологій класифікації тональності речень. Постановка задач дослідження			Аналітичний огляд літературних джерел, задачі досліджень, розділ 1 ПЗ
2	Побудова математичних моделей функціонування нейронної мережі			Математичні моделі, розділ 2
3	Практичне застосування та оцінка ефективності розроблених моделей			розділ 3
4	Підготовка економічної частини			розділ 4
5	Апробація та/або впровадження результатів дослідження			тези доповідей/акт впровадження
6	Оформлення пояснювальної записки, графічного матеріалу та презентації			Пояснювальна записка, графічний матеріал, презентація

Консультанти з окремих розділів магістерської кваліфікаційної роботи

1. Науковий керівник _____ канд. техн. наук, доц., доц. кафедри КН
(підпис) наук. ступінь, вчене звання (посада)
"____" _____20___ р. _____ О. К. Колесницький _____
ініціали та прізвище

2. Економічна частина _____ канд. екон. наук, доц. доц. каф. ЕПВМ
(підпис) наук. ступінь, вчене звання (посада)
___М. В. Бальзан _____
ініціали та прізвище
"____" _____20___ р.

Дата попереднього захисту роботи "____" _____20___ р.

Рецензент _____ канд. техн. наук, доц., доц. кафедри ПЗ
(підпис) наук. ступінь, вчене звання (посада)
_____ В. В. Войтко _____
ініціали та прізвище

Завдання видав
науковий керівник _____ канд. техн. наук, доц., доц. кафедри КН
(підпис) наук. ступінь, вчене звання (посада)
_____О. К. Колесницький _____
ініціали та прізвище
"____" _____20___ р.

Завдання отримав магістрант _____ _____В.С. Барановський_____
(підпис) ініціали та прізвище
"____" _____20___ р.

АНОТАЦІЯ

Дана магістерська кваліфікаційна робота присвячена розробці програмного забезпечення для класифікації тональності речень за допомогою згорткової нейронної мережі. Були розглянуті основні методи класифікації тональності речень, визначені їх переваги і недоліки, та на основі цього був зроблений вибір на користь згорткової нейронної мережі. Було досліджено структуру, математичну модель та алгоритм роботи обраної мережі, який слугував базисом для розробки алгоритму роботи програмного забезпечення. Було визначено оптимальне для потенційного користувача програмне середовище, алгоритм роботи програми та в кінцевому підсумку на основі проведених досліджень був побудований програмний засіб класифікації тональності речень. Створене програмне забезпечення написане на мові програмування Python та порівняно з аналогом має кращу на 4,5% достовірність класифікації тональності речень.

ABSTRACT

The master’s degree work is dedicated to the development of software for classifying sentence tone using a convolutional neural network. The basic methods of classifying the tone of sentences were considered, their advantages and disadvantages were determined, and on this basis a choice was made in favor of a convolutional neural network. The structure, mathematical model and algorithm of work of the selected network were investigated, which served as the basis for the development of the algorithm of software operation. The optimal software environment for the potential user was determined, the algorithm of the program operation and finally, on the basis of the conducted researches, a software tool was used to classify the tone of sentences. The created software is written in Python programming language and has a 4.5% better classification accuracy of sentence tone compared to its counterpart.

ЗМІСТ

ВСТУП

Актуальність. У сучасному світі, підприємства та організації завжди бажають дізнатися думки своїх споживачів або користувачів щодо своїх продуктів чи послуг. Самі споживачі також хочуть знати думки вже існуючих користувачів продукту до її покупки чи певної послуги перед її оплатою, думки інших людей щодо політичних кандидатів до прийняття рішення за кого голосувати на політичних виборах. У минулому, коли людині необхідно було отримати пораду перед купівлею товару щодо його якості, вона питала своїх друзів і родину. Коли певній організації або підприємству необхідно було дізнатися громадську думку про свій продукт чи послугу, то вони проводили анкетування або опитування серед своїх споживачів, або проводили фокусні групові співбесіди. Отримання і накопичення думок споживачів і громадян вже давно стали серйозним і великим бізнесом, а саме в сфері маркетингу, зв'язків з громадськістю та компаній, які займаються політичними кампаніями.

При стрімкому зростанні соціальних медіа (наприклад, огляди, форуми обговорення, блоги, мікро-блоги, Twitter, коментарі та повідомлення в соціальних мережах) в Інтернеті, приватні особи та організації все частіше використовують вміст у цих середовищах для прийняття рішень. Сьогодні, якщо хтось бажає купити певний продукт, то майбутній покупець вже не обмежується лише опитуванням своїх друзів і родини, тому що є багато відгуків користувачів та обговорення на різних форумах в Інтернеті про вибраний цією людиною продукт.

Для організації вже не потрібно проводити опитування, анкетування, і фокусні групові співбесіди, щоб отримати відгуки від своїх користувачів, тому що є велика кількість такої інформації у відкритому доступі. Тим не менш, постійний пошук і моніторинг сайтів в Інтернеті, відбір необхідної інформації, залишаються доволі складним завданням через значне поширення різноманітних таких сайтів. Кожен сайт, як правило, містить величезний обсяг тексту, який містить необхідну інформацію щодо певного продукту чи певної

послуги, але цю інформацію не завжди легко виявити в довгих блогах, на форумах чи у великих повідомленнях.

Середньостатичний користувач може зіткнутися з труднощами із визначенням відповідних сайтів, видобуванням та узагальненням потрібної інформації на них. І це зумовило розвиток галузі автоматичний аналіз тональності текстів.

Зв'язок роботи з науковими програмами, планами, темами. Магістерська робота виконана відповідно до напрямку наукових досліджень кафедри комп'ютерних наук Вінницького національного технічного університету 22 К1 «Моделі, методи, технології та пристрої інтелектуальних інформаційних систем управління, економіки, навчання та комунікацій» та плану наукової та навчально-методичної роботи кафедри.

Мета і завдання досліджень. Метою магістерської кваліфікаційної роботи є підвищення достовірності класифікації тональності речень програмними засобами за рахунок застосування штучних нейронних мереж.

Для досягнення мети розробки необхідно виконати такі задачі:

- провести аналіз проблеми розв'язання задачі класифікації тональності речень;
- розглянути існуючі методи вирішення задачі класифікації тональності речень та обрати й обґрунтувати вибір методу, який задовольняє мету даної магістерської кваліфікаційної роботи;
- розробити математичну модель класифікації тональності речень;
- сформулювати стадії інформаційної технології, розробити структуру та алгоритм роботи програмного засобу;
- виконати програмну реалізацію запропонованої інформаційної технології;
- провести тестування програмного продукту та виконати аналіз отриманих результатів.

Об'єкт дослідження – процес комп'ютеризованої класифікації тональності речень з використанням штучних нейронних мереж.

Предмет дослідження – інформаційна технологія та програмні засоби класифікації тональності речень з використанням штучних нейронних мереж та достовірність їх роботи.

Методи дослідження. У роботі використані наступні методи наукових досліджень: системного аналізу, інтелектуального аналізу даних, теорії штучних нейронних мереж для реалізації інформаційної технології класифікації тональності речень, методи математичної статистики для розробки процесу класифікації тональності речень та обрахунків результатів експериментів із програмним засобом, об'єктно-орієнтованого програмування.

Наукова новизна одержаних результатів.

1. Набула подальшого розвитку інформаційна технологія класифікації тональності речень, яка відрізняється використанням згорткової нейронної мережі, що дозволило підвищити достовірність класифікації тональності речень.

2. Удосконалено архітектуру згорткової нейронної мережі, яка відрізняється зменшенням згорткових шарів до 3 та викристанням повної зв'язності в останньому шарі, що дозволило підвищити швидкодію мережі без втрати достовірності класифікації тональності речень.

Практичне значення одержаних результатів полягає в тому, що на основі проведених досліджень розроблено програмне забезпечення класифікації тональності речень.

Запропонована інформаційна технологія сприяє підвищенню достовірності процесу класифікації тональності речень, зокрема:

– розроблено алгоритм роботи програмного забезпечення класифікації тональності речень на основі згорткової нейронної мережі;

– розроблено програмні засоби для класифікації тональності речень на основі згорткової нейронної мережі;

Достовірність теоретичних положень магістерської кваліфікаційної роботи підтверджується коректністю постановки завдання, коректністю використання математичного апарату методів дослідження, експериментальними дослідженнями тестування програмної реалізації інформаційної технології класифікації тональності речень. Адекватність

розроблених математичних моделей підтверджується результатами експериментальних досліджень.

Особистий внесок здобувача. Усі результати, наведені у магістерській кваліфікаційній роботі, отримані самостійно. У працях, написаних у співавторстві, здобувачу належать: аналіз процесу класифікації тональності речень на основі згорткової нейронної мережі [1] та методи підвищення достовірності [2].

Апробація результатів роботи. Результати досліджень апробовані на XLVII Науково-технічній конференції підрозділів Вінницького національного технічного університету (НТКП ВНТУ) 21–23 березня 2018 року [1] та на конференції «Молодь в науці: дослідження, проблеми, перспективи-2020», Вінниця, 2019, м. Вінниця [2].

Публікації. За результатами досліджень опубліковано двоє тез доповідей на науково-технічній конференції [1,2] та подано заявку на авторське свідоцтво на твір (програму).

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ КЛАСИФІКАЦІЇ ТОНАЛЬНОСТІ РЕЧЕНЬ

1.1 Аналіз технічної проблеми та постановка задачі класифікації тональності речень

Актуальність завдання класифікації тональності речень обумовлена тим, що обсяг даних цієї процедури такий, що її нормальне функціонування без використання засобів обчислювальної техніки та програмного забезпечення неможливо. Це обумовлено, насамперед, тим, що обсяг операцій, виконуваних операторами настільки великий, що їх виконання в ручному режимі на паперових носіях просто неможливо. Другою причиною є необхідність збільшення швидкодії та підвищення достовірності класифікації.

Використання розроблювальних програмних засобів надає зручність і швидкість при класифікації тональності речень, піднімає процес класифікації на принципово новий рівень.

Таким чином, можна виділити наступні основні переваги використання:

- підвищення достовірності класифікації тональності речень;
- підвищення зручності класифікації тональності речень;
- підвищення швидкодії класифікації тональності речень;

При виконанні роботи одразу була взята орієнтація на розробку власного програмного продукту, а не на використання одного з існуючих. Така установка була зроблена виходячи з того , що у власній розробці завжди можна реалізувати специфічні , необхідні і користувачам можливості. Не виключено , що вимоги до реалізації таких можливостей з'являться вже після розробки та впровадження системи , на етапі роботи , якщо у замовника з'являться нові погляди на необхідний функціонал системи , пропозиції щодо поліпшення або зміни логіки поведінки системи.

У даній роботі потрібно розробити інформаційну технологію та програмні засоби класифікації тональності речень, використовуючи сучасні

фреймворки та нейронну мережу. Функціонально розроблювана програма дозволить вивести всю необхідну інформацію по тональності тексту. Нейронна мережа допоможе системі швидко визначити тональність тексту. Безпосередньо при виконанні класифікації тональності кожен користувач зможе побачити який характер несе речення.

Вхідною інформацією є текст або речення, введене з клавіатури у вікні додатку. Вихідною інформацією є один із 3 рівнів тональності речень: негативне, позитивне, нейтральне.

У ході виконання роботи необхідно вивчити предметну область автоматичного аналізу речень, вивчити всі тонкощі процедури класифікації, скласти уявлення про порядок дій, можливих штатних і позаштатних ситуаціях, даних, які потрібні в процесі класифікації.

Далі необхідно ознайомитися з існуючим станом справ в області автоматизації процедури класифікації, вивчити чи є аналоги і зробити висновок про необхідність розробки нового продукту, розробити для нього технічне завдання, необхідно вибрати засоби розробки: операційну систему, мову і середовище програмування.

Для розробки програмних модулів використовується технологія програмування Pyton. Вибраною мовою програмування необхідно реалізувати додаток, що виконує всі модулі класифікації тональності речень, а саме, розробити всі форми, програмні модулі, схеми їх взаємодії, порядок звернення. Після цього необхідно провести всебічне тестування розробленого додатку.

1.2 Аналіз існуючих методів класифікації тональності речень

Існує багато підходів для вирішення проблеми розробки програмного забезпечення класифікації тональності речень, тому розглянемо деякі з них.

В сучасних системах автоматичного визначення емоційної оцінки тексту найчастіше використовується одномірний емотивний простір: позитив чи

негатив (добре або погано). Однак, відомі успішні випадки використання і багатовимірних просторів.

Основним завданням в аналізі тональності є класифікація полярності даного документа, тобто визначення, чи є виражена в документі думка або пропозиція позитивною, негативною або нейтральною. Більш розгорнуто, «поза полярності» класифікація тональності виражається, наприклад, такими емоційними станами, як «злий», «сумний» і «щасливий».

Існують такі методи класифікації тональності [3] як:

- метод опорних векторів,
- на основі теореми Баєса,
- нейромережеві.

Метод опорних векторів – це метод навчання з учителем, що використовується для бінарної класифікації. Даний алгоритм машинного навчання будує розділяючу поверхню у гіперпросторі з точок, що лежать під полярними підмножинами, тобто розмежовує класи. Точки побудованої поверхні називаються опорними векторами. Цей класифікатор може замінювати нейронні мережі, але має дуже повільний процес навчання.

В якості найпростішого методу для класифікації тональності тексту використовується наївний класифікатор Баєса. У даному класифікаторі використовується теорема Баєса для визначення ймовірності приналежності елемента вибірки до одного з класів при припущенні незалежності ознак.

Для автоматичного визначення тональності речення чи тексту можна використовувати штучні нейронні мережі. Перевагами застосування нейронних мереж є: здатність до навчання, стійкість до шумів вхідних даних. Алгоритми навчання штучних нейронних мереж поділяються на алгоритми навчання з учителем та без учителя. Для аналізу текстових даних, доцільно використовувати згорткову нейронну мережу.

Отже, аналіз існуючих способів класифікації тональності речень показав, що найбільш перспективними для виявлення у текстах та реченнях емоційної забарвленості є штучні нейронні мережі.

1.3 Обгрунтування вибору аналогу програми класифікації тональності речень

З розвитком потужностей обчислювальної техніки виростала кількість різних систем аналізу тональності речень. На сьогодні розроблено досить багато продуктів на допомогу класифікаторам. Розглянемо деякі з них.

Сервіс Twitter зараз дуже популярний. Користувачі сервісу, зокрема, пишуть в своїх повідомленнях відгуки про товари. Веб-сервіс Twitter Sentiment [4] дозволяє аналізувати інформацію про продукт, який згадують користувачі, за допомогою даних з веб-сервісу Twitter. Користувачеві Twitter Sentiment досить ввести слово (див. рис.1.1), і програма проаналізує всі останні 100 записів про це слово. При цьому буде побудований графік співвідношення позитивних і негативних відгуків. В сукупності, надається легкий спосіб проаналізувати думки користувачів про будь-які продукти. Twitter Sentiment використовує метод машинного навчання і також має API.

Рисунок 1.1 Аналіз тональності речень «Twitter Sentiment»

Аналіз тональності речень « I-Teco» [5]. Даний програмний продукт є веб-сервісом (див. рис. 1.2), який дозволяє визначати емоційне забарвлення тексту, введеного користувачем. У ньому використовуються метрики і спеціальні словники емоційно забарвленої лексики.

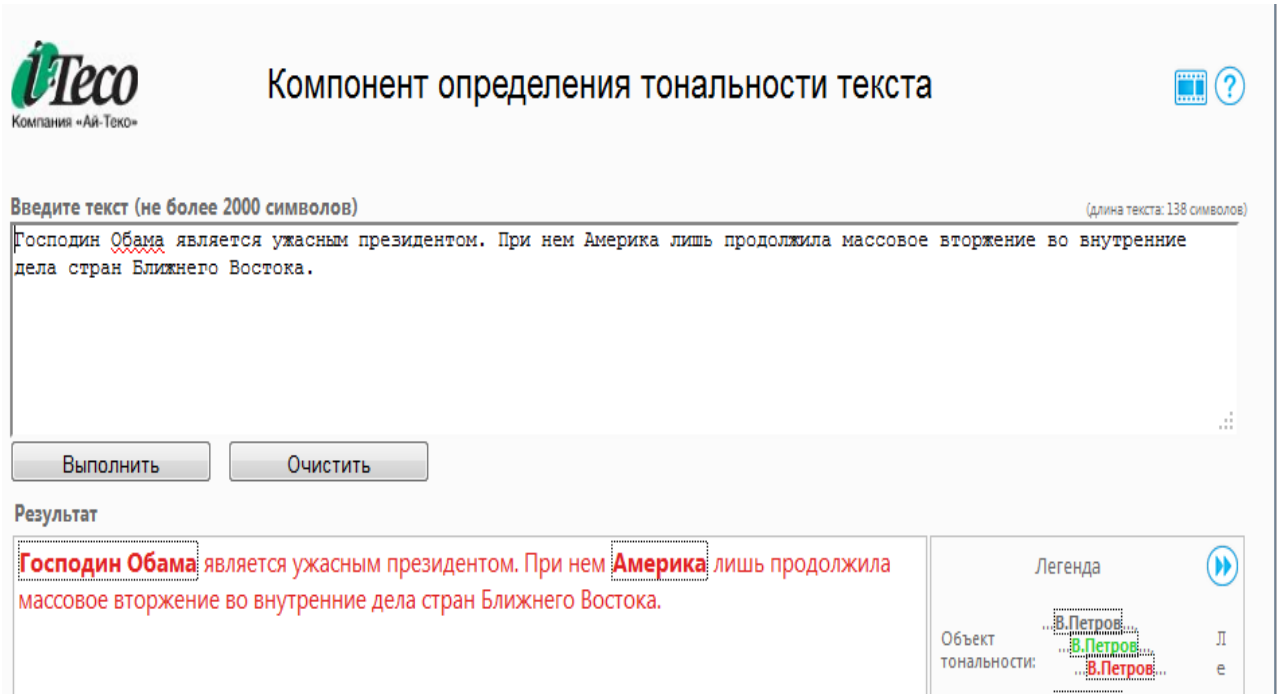

Рисунок 1.2 – Аналіз тональності речень «I-Teco»

Програма «Twitter Sentiment» має такі недоліки як аналіз даних тільки веб-сервісу Twitter та має достовірність ~85%, в свою чергу сервіс I-Teco» може аналізувати будь які речення але має досить низьку достовірність через навчання через словники, близько ~80%. Тому проаналізувавши ці програми була поставлена мета підвищення достовірності класифікації речень та додавання можливості введення тексту з клавіатури у вікні програми.

1.4 Висновок

У даному розділі був проведений аналіз суті технічної проблеми класифікації тональності речень, поставлене завдання та проаналізовані існуючі

способи вирішення проблеми. Для задачі класифікації тональності речень було обрано метод на основі штучних нейронних мереж, адже вони мають здатність до навчання та стійкість до шумів вхідних даних. Перспективним є використання саме згорткової нейронної мережі, адже за допомогою шарів згортки можна зменшити вхідні даніі, а відтак і зменшити складність обчислень. Також було проаналізовано існуючі програми-аналоги та зроблено постановку задачі. На основі вихідних даних, аналізу сучасних методів класифікації тональності речень та існуючих програм-аналогів було зроблено висновок, що метою роботи є підвищення достовірності.

2 РОЗРОБКА ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ КЛАСИФІКАЦІЇ ТОНАЛЬНОСТІ РЕЧЕНЬ НА ОСНОВІ НЕЙРОННОЇ МЕРЕЖІ

2.1 Обґрунтування вибору типу нейронної мережі

Відомо багато парадигм нейронних мереж, але нам потрібно обрати найбільш доцільну для задач класифікації тональності речень. Ця задача відноситься до задач загальної класифікації. Нам потрібно буде виконувати класифікацію речень на 3 класи (позитивне, негативне, нейтральне). Для таких задач можна використовувати такі нейронні мережі як багатошаровий парцептрон, мережі радіально-базисних функцій (РБФ), згорткові нейронні мережі. Розглянемо їх детальніше для вибору того типу нейронної мережі, який найкраще підійде для класифікації тональності речень.

2.1.1 Багатошаровий перцептрон.

Багатошарові мережі можна одержати каскадним з'єднанням одношарових мереж, де вихід одного шару є входом для наступного шару, причому така мережа може привести до збільшення обчислювальної потужності лише в тому випадку, якщо активаційна функція між шарами буде нелінійною.

Багатошаровий перцептрон – багатошарова нейронна мережа прямого поширення. Зазвичай складається з множини сенсорних елементів, які утворюють вхідний шар, одного чи декількох прихованих шарів нейронів і одного вихідного шару нейронів. Вхідний сигнал поширюється мережею в прямому напрямку, від шару до шару [6].

Для навчання багатошарового перцептрону використовують алгоритм зворотного поширення похибки, який передбачає два проходи по всіх шарах мережі: прямий і зворотний. Під час прямого проходу образ (вхідний вектор) подається на сенсорні вузли мережі, після чого поширюється від шару до шару. В результаті генерується набір вихідних сигналів, який є реакцією мережі на цей вхідний образ. Під час прямого проходу всі синаптичні ваги мережі

фіксовані. Під час зворотного проходу всі синаптичні ваги налаштовуються відповідно до правила корекції помилок, а саме: фактичний вихід мережі віднімається від цільового відгуку, в результаті чого формується сигнал помилки. Цей сигнал згодом поширюється мережею в напрямку, зворотному до напрямку синаптичних зв'язків. Синаптичні ваги налаштовуються з метою максимального наближення вихідного сигналу мережі до бажаного [6].

Багатошарові перцептрони мають три характерні ознаки:

1. Кожен нейрон мережі має нелінійну функцію активації. Важливо підкреслити, що така нелінійна функція є гладкою (тобто всюди диференційованою), на відміну від жорсткої порогової функції, використовуваної в перцептроні Розенблатта. Найпоширенішою формою функції, що задовольняє ці вимоги, є сигмоїдальна

$$y_i = \frac{1}{1+\exp(-v_j)},$$

де v_j – індуковане локальне поле (зважена сума всіх синаптичних входів плюс порогове значення) нейрона *j*;

y_j – вихід нейрона.

2. Мережа містить один чи декілька шарів прихованих нейронів. Ці нейрони, послідовно витягаючи найважливіші ознаки вхідного образу, дозволяють мережі навчатись розв'язувати складні задачі.

3. Мережа має високий ступень зв'язності, що реалізується завдяки синаптичним зв'язкам. Зміна рівня зв'язності вимагає зміни множини синаптичних зв'язків або їхніх вагових коефіцієнтів.

Комбінація цих властивостей поєднано із здатністю до навчання на власному досвіді забезпечує високу потужність багатошарового перцептрону. Однак, ті самі властивості є причиною неповноти знань про мережі такого типу.

Це пояснюється тим, що, по-перше, розподілена форма нелінійності та висока зв’язність мережі значно ускладнюють теоретичний аналіз багатошарового перцептрону. По-друге, наявність прихованих нейронів робить процес навчання складнішим для візуалізації. Саме в процесі навчання необхідно визначити, які ознаки вхідного сигналу необхідно подавати прихованими нейронами. Отже, процес навчання ускладнюється, оскільки пошук необхідно виконувати у дуже широкій області можливих функцій, а вибір – серед альтернативних представлень вхідних образів [6].

Якщо розглянути просту двошарову мережу із двома нейронами в першому шарі, з'єднаними з єдиним нейроном у другому шарі, то кожен нейрон першого шару розбиває площина на дві напівплощини, утворюючи в просторі образів область V-форми, а нейрон другого шару реалізує різні функції при підходящому виборі ваг і порога. Аналогічно в другому шарі може бути використано три нейрони з подальшою розбивкою площини й створенням області трикутної форми. Включенням достатнього числа нейронів у вхідний шар може бути утворений опуклий багатокутник будь-якої бажаної форми. Крапки, не складові опуклої області, не можуть бути відділені про інших крапок площини двошаровою мережею.

На рис. 2.1 зображено архітектурний граф багатошарового перцептрона з двома прихованими шарами та одним вихідним шаром. Ця мережа є повнозв’язною – кожен нейрон у будь-якому шарі мережі зв’язаний зі всіма нейронами попереднього шару. Сигнал передається мережею виключно в прямому напрямку, від шару до шару.

У мережах цього типу існують два типи сигналів:

Функціональний сигнал – вхідний сигнал, що надходить у мережу та проходить вперед від одного нейрона до іншого всією мережею. Такий сигнал досягає кінця мережі у вигляді вихідного сигналу.

Сигнал похибки – бере початок на виході мережі і поширюється в зворотному напрямку від шару до шару. Отримав свою назву внаслідок того, що він обчислюється кожним нейроном мережі на основі функції похибки.

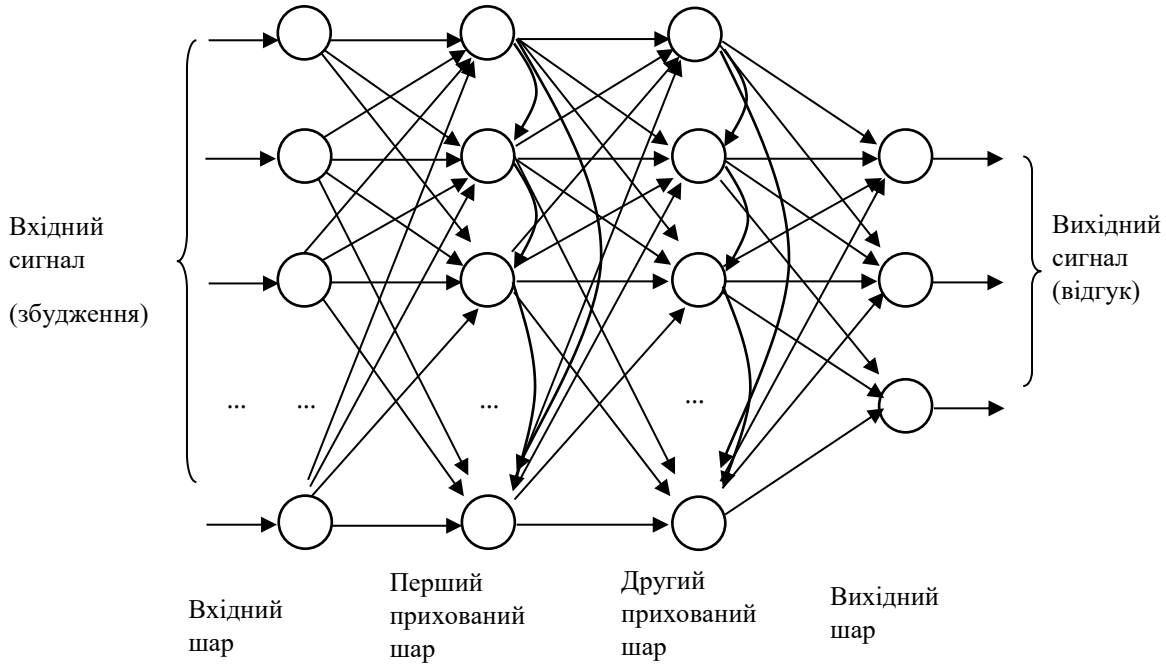

Рисунок 2.1 - Архітектурний граф багатошарового перцептрона з двома прихованими шарами

Тришарова мережа є більш загальною. Її можливості класифікації обмежені лише числом нейронів і ваг. Обмеження на опуклість відсутні. Тепер нейрон третього шару приймає як вхід набір опуклих багатокутників і їхня логічна комбінація може бути неопуклої форми. При додаванні нейронів і ваг число сторін багатокутника може необмежено зростати. Це дозволяє апроксимувати область будь-якої форми з будь-якою точністю. На додачу не всі вихідні області другого шару повинні перетинатися. Отже, можливо поєднувати різні області, опуклі та неопуклі, видаючи на виході одиницю щораз, коли вхідний вектор належить одній з них.

Будь-який прихований чи вихідний нейрон багатошарового перцептрона може виконувати обчислення двох типів:

1. Обчислення функціонального сигналу на виході нейрона, що реалізується у вигляді неперервної нелінійної функції від вхідного сигналу і синаптичних ваг, пов’язаних з цим вектором.

2. Обчислення оцінки вектора градієнта (тобто поверхні похибки за синаптичними вагами, пов’язаними зі входами цього нейрона), необхідного для зворотного проходу мережею.

З декількох причин дуже популярним є навчання багатошарового перцептрона за алгоритмом зворотного поширення похибки, а саме:

– локальність методу змін синаптичних ваг і порогів у багатошаровому перцептроні.

– ефективність методу обчислення всіх часткових похідних функцій вартості за вільними параметрами.

Однак, метод зворотного поширення похибки є ітераційним і вимагає щонайменше декількох проходів через мережу всієї навчальної вибірки, відповідно, застосування його для розв’язання задач інтелектуального аналізу даних є неможливим, оскільки через великий обсяг даних та їх багатовимірність процес триватиме занадто довго.

2.1.2 Нейронні мережі на основі радіальних базисних функцій

Побудову нейронної мережі, згідно з [7], можна звести до задачі апроксимації поверхні відгуку за деякими базовими точками в просторі великої вимірності. Тобто навчання еквівалентне побудові такої поверхні в багатовимірному просторі, яка б найточніше відповідала даним навчання, де критерій ”найкращої відповідності” обирають у деякому статистичному сенсі.

Базова архітектура RBF-мережі передбачає наявність трьох шарів нейронів, що виконують різні функції (рис. 2.2).

Вхідний шар складається із сенсорних елементів, які пов’язують мережу із зовнішнім середовищем. Другий шар є єдиним прихованим шаром, який переважно має значно більшу розмірність, ніж вхідний.

Кожен елемент прихованого шару використовує активаційну функцію Гауса (2.1).

$$h(\bar{x}) = \exp(-\frac{\|\bar{x} - \bar{c}\|^2}{r^2}) \quad (2.1)$$

де $\bar{x}$ – вхідний вектор (матриця розсіяння для задачі неруйнівного контролю),

$\bar{c}$ – центр функції Гаусса,

r – радіус функції Гаусса.

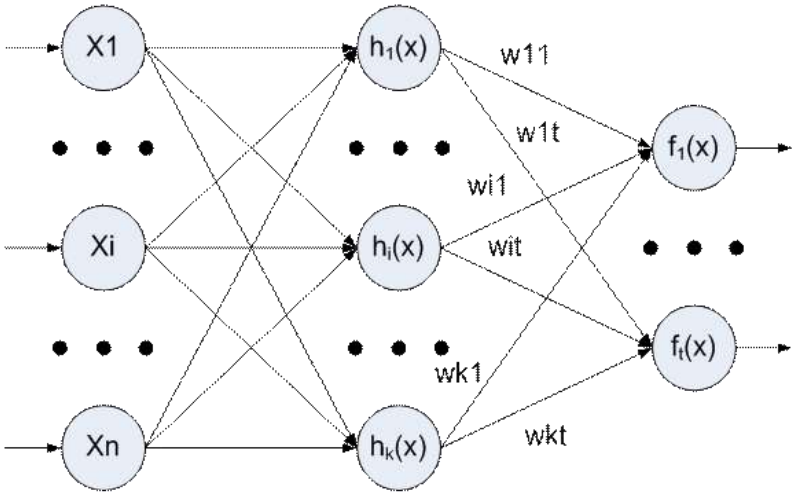

Рисунок 2.2 - Мережа на основі радіальних базисних функцій

У центрі радіальної базисної функції (функції ядра) точка, визначена ваговим вектором, що пов’язаний із нейроном. Позиція та ширина функції ядра мають бути навчені на тренувальній вибірці прикладів. Як правило, кількість ядер є набагато меншою, ніж кількість навчальних прикладів.

Кожен вихідний елемент обчислює лінійну комбінацію цих радіальних базисних функцій за формулою (2.2).

$$f(\bar{x}) = \sum_{j=1}^{m} w_j h_j(\bar{x}) \quad (2.2)$$

З погляду апроксимації приховані елементи формують сукупність функцій, які створюють базисну систему для представлення вхідних прикладів у побудованому на ній просторі.

Така архітектура ґрунтується на висновках з теореми Ковера про роздільність образів, згідно з якими нелінійне перетворення складної задачі класифікації образів на простір більшої вимірності підвищує ймовірність лінійної роздільність образів. Також розмірність прихованого шару безпосередньо пов’язана із можливістю мережі апроксимувати гладке відображення ”вхід–вихід”. Чим вища розмірність прихованого шару, тим вища точність апроксимації [7].

Існують різні алгоритми навчання RBF-мереж. Основний алгоритм використовує двокрокову стратегію навчання або змішане навчання. Він оцінює позицію та ширину ядра з використанням алгоритму кластеризації „без вчителя”, а потім алгоритм мінімізації середньоквадратичної похибки „з вчителем” для визначення ваг між прихованим і вихідним шарами. Оскільки вихідні елементи є лінійними, застосовується неітераційний алгоритм. Після отримання цього початкового наближення використовується градієнтний спуск для уточнення параметрів мережі [6,7].

Такий змішаний алгоритм навчання RBF-мережі збігається набагато швидше, ніж алгоритм зворотного поширення похибки для навчання багатошарових перцептронів, однак RBF-мережа часто містить занадто велику кількість прихованих елементів, а отже, функціонування RBF-мережі в такому випадку є повільнішим за функціонування багатошарового перцептрона. Тому ефективність (похибка залежно від розміру мережі) RBF-мереж та багатошарового перцептрона залежить від розв’язуваної задачі [7].

Якщо порівнювати між собою багатошаровий перцептрон та мережу на основі радіальних базисних функцій, то можна зазначити, що багатошаровий перцептрон забезпечує глобальну апроксимацію нелінійного відображення, а RBF-мережа за допомогою локалізованих нелінійностей, що експоненційно

зменшуються (функції Гауса) створює локальну апроксимацію нелінійного відображення.

2.1.3 Згорткові нейронні мережі

Згóрткові нейрóнні мерéжі (ЗНМ, англ. convolutional neural network, CNN, ConvNet) в машинному навчанні — це клас глибинних штучних нейронних мереж прямого поширення (див. рис. 2.3), який успішно застосовувався до аналізу, зокрема, візуальних зображень.

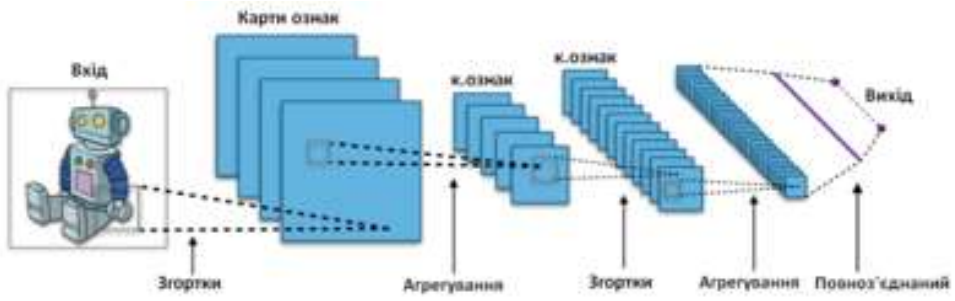

Рисунок 2.3 - Згорткова нейронна мережа

ЗНМ складається з шарів входу та виходу, а також із декількох прихованих шарів. Приховані шари ЗНМ зазвичай складаються зі згорткових шарів, агрегувальних шарів, повноз'єднаних шарів та шарів нормалізації.

Згорткові шари застосовують до входу операцію згортки, передаючи результат до наступного шару. Згортка імітує реакцію окремого нейрону на зоровий стимул [8]. Кожен згортковий нейрон обробляє дані лише для свого рецептивного поля.

Хоч повноз'єднані нейронні мережі прямого поширення й можливо застосовувати як для навчання ознак, так і для класифікування даних, застосування цієї архітектури до зображень є непрактичним. Було би необхідним дуже велике число нейронів, навіть у поверхневій (протилежній до глибинної) архітектурі, через дуже великі розміри входу, пов'язані з зображеннями, де кожен піксель є відповідною змінною. Наприклад,

повноз'єднаний шар для (маленького) зображення розміром 100 × 100 має 10 000 ваг. Операція згортки дає змогу розв'язати цю проблему, оскільки вона зменшує кількість вільних параметрів, дозволяючи мережі бути глибшою за меншої кількості параметрів [8]. Наприклад, незалежно від розміру зображення, області замощування розміру 5 × 5, кожна з одними й тими ж спільними вагами, вимагають лише 25 вільних параметрів. Таким чином, це розв'язує проблему зникання або вибуху градієнтів у тренуванні традиційних багатошарових нейронних мереж з багатьма шарами за допомогою зворотного поширення.

Агрегувальні шари. Згорткові мережі можуть включати шари локального або глобального агрегування, які об'єднують виходи кластерів нейронів одного шару до одного нейрону наступного шару [8]. Наприклад, максимізаційне агрегування (англ. max pooling) використовує максимальне значення з кожного з кластерів нейронів попереднього шару [8]. Іншим прикладом є усереднювальне агрегування (англ. average pooling), що використовує усереднене значення з кожного з кластерів нейронів попереднього шару.

Повноз'єднані шари з'єднують кожен нейрон одного шару з кожним нейроном наступного шару. Це, в принципі, є тим же, що й традиційна нейронна мережа багатошарового перцептрону (БШП).

У багатошарового парцептрона перевага в тому, що він може навчити всі шари нейронної мережі, і його легко прорахувати локально. Однак, цей метод є дуже довгим, до того ж, для його застосування потрібно, щоб передавальна функція нейронів була диференційовною.

Нейронні мережі на основі радіальних базисних функцій володіють поганими екстраполятивними властивостями і виходять вельми громіздкими при великій розмірності вектора входів.

В згорткових нейронних мережах немає таких недоліків як у багатошаровому парцептроні та РБФ мережах, а ще є така перевага, яка полягає у тому, що вони можуть надійно класифікувати дуже схожі між собою образи.

Тому, проаналізувавши ці нейронні мережі, був зроблений висновок, що згорткова мережа підходить найбільше для класифікації тональності речень.

2.2 Розробка архітектури згорткової нейронної мереж

Згорткові нейронні мережі (ЗНМ) в машинному навчанні — це клас глибинних штучних нейронних мереж прямого поширення, який успішно застосовувався до розпізнавання зображень та обробки текстів.

ЗНМ використовують різновид багатошарових перцептронів, розроблений так, щоб вимагати використання мінімального обсягу попередньої обробки. Вони відомі також як інваріантні відносно зсуву або просторово інваріантні штучні нейронні мережі, виходячи з їхньої архітектури, спільних ваг та характеристик інваріантності відносно паралельного перенесення.

Згорткові мережі було натхнено біологічними процесами, в яких схему з'єднання нейронів натхнено організацією зорової кори тварин. Окремі нейрони кори реагують на стимули лише в обмеженій області зорового поля, відомій як рецептивне поле. Рецептивні поля різних нейронів частково перекриваються таким чином, що вони покривають усе зорове поле.

ЗНМ використовують порівняно мало попередньої обробки в порівнянні з іншими алгоритмами класифікування зображень. Це означає, що мережа навчається фільтрів, що в традиційних алгоритмах розроблялися вручну. Ця незалежність у конструюванні ознак від апріорних знань та людських зусиль є великою перевагою.

Згорткова нейронна мережа складається з шарів входу та виходу, а також із декількох прихованих шарів (див. рис. 2.4). Приховані шари ЗНМ зазвичай складаються зі згорткових шарів, агрегувальних шарів, повноз'єднаних шарів та шарів нормалізації.

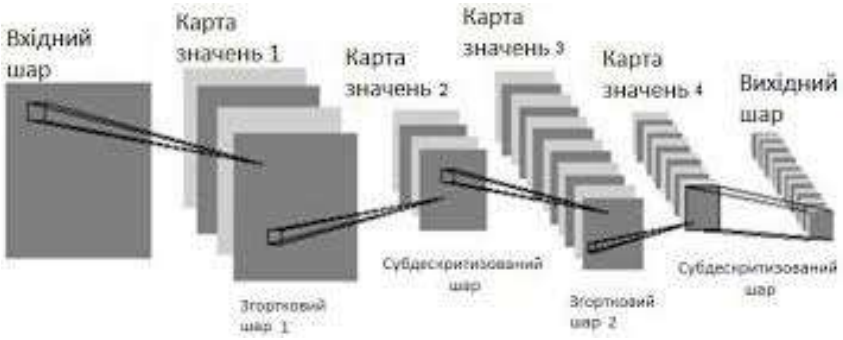

Рисунок 2.4 – Згорткова нейронна мережа

Цей процес описують в нейронних мережах як згортку за домовленістю. З математичної точки зору він є радше взаємною кореляцією, ніж згорткою. Це має значення лише для індексів у матриці, й відтак, які ваги на якому індексі розташовуються.

Згорткові шари застосовують до входу операцію згортки, передаючи результат до наступного шару. Згортка імітує реакцію окремого нейрону на зоровий стимул. Кожен згортковий нейрон обробляє дані лише для свого рецептивного поля. Хоч повноз'єднані нейронні мережі прямого поширення й можливо застосовувати як для навчання ознак, так і для класифікування тональності речень.

Згорткові мережі можуть включати шари локального або глобального агрегування, які об'єднують виходи кластерів нейронів одного шару до одного нейрону наступного шару. Наприклад, максимізаційне агрегування (англ. max pooling) використовує максимальне значення з кожного з кластерів нейронів попереднього шару. Іншим прикладом є усереднювальне агрегування (англ. average pooling), що використовує усереднене значення з кожного з кластерів нейронів попереднього шару.

ЗНМ використовують спільні ваги в згорткових шарах, що означає, що для кожного рецептивного поля шару використовується один і той же фільтр це зменшує обсяг необхідної пам'яті та поліпшує продуктивність.

Для задачі класифікації тональності речень буде використано три згорткових шари з фільтрами. Кількість фільтрів – 16 для кожного шару.

Відповідно до цього кожен згортковий шар утворить 16 ознак. Оскільки ширина «вікна» фільтра дорівнює N, то кількість стовпців карти ознак дорівнює одиниці. Кількість рядків карти ознак обчислюється наступним чином:

$$H = M - FH + 1, \text{де} \quad (2.1)$$

M – максимально можлива кількість слів у тексті,

FH – висота «вікна» фільтра.

Іншим важливим поняттям ЗНМ є агрегування, яке є різновидом нелінійного зниження дискретизації. Існує декілька нелінійних функцій для реалізації агрегування, серед яких найпоширенішою є максимізаційне агрегування:

$$\hat{c} = \max\{c\} \quad (2.2)$$

Воно розділяє вхідну матрицю на набір прямокутників без перекриттів, і для кожної такої підобласті виводить її максимум. Ідея полягає в тому, що точне положення ознаки не так важливе, як її грубе положення відносно інших ознак. Агрегувальний шар слугує поступовому скороченню просторового розміру представлення для зменшення кількості параметрів та об'єму обчислень у мережі. В архітектурі ЗНМ є звичним періодично вставляти агрегувальний шар між послідовними згортковими шарами.

Для задачі класифікації текстів кожен агрегувальний шар перетворюватиме вихід згорткового шару у матрицю розмірністю 16 х 1, що значно знижуватиме розмірність, а відповідно і складність обчислень.

Після кількох згорткових та максимізаційно агрегувальних шарів, високорівневі міркування в нейронній мережі здійснюються повноз'єднаними шарами. Нейрони у повноз'єднаному шарі з'єднуються з усіма нейронами попереднього шару.

Після проведення операцій згортки та агрегування буде виділено по 16 ознак з кожної зв’язки згорткового та агрегувального шарів, які будуть подані на вихідний повнозв’язний шар.

2.3 Математична модель згорткової нейронної мережі

З математичної точки зору штучна нейронна мережа – паралельно розподілений процесор, який володіє здатністю до навчання, збереження і представлення знань, набутих на основі досвіду. Якщо розглядати нейронну мережу як спосіб представлення знань, то в ній зберігаються знання про асоціативні зв'язки між стимулами (вхідними векторами) та відгуками (вихідними векторами). Знання зберігаються (формуються в процесі навчання) зазвичай у формі ваг зв'язків між нейронами.

В основі нейронних мереж лежить елементарний перетворювач – штучний нейрон [9], названий так за аналогією з його біологічним прототипом. Штучний нейрон складається з входів (синапсів), суматора, нелінійного перетворювача і виходу (аксона). Всі нейрони з'єднуються між собою зв'язками, які називаються вагами і визначаються певними величинами – ваговими коефіцієнтами.

У цій моделі нейрона можна виділити три основні елементи:

- синапси, кожен з яких характеризується своєю вагою або силою. Здійснюють зв'язок між нейронами, множать вхідний сигнал на ваговий коефіцієнт синапсу, що характеризує силу синаптичного зв'язку;
- акумулятор, аналог тіла клітини нейрона. Виконує складання зовнішніх вхідних сигналів або сигналів, що надходять по синаптичних зв'язках від інших нейронів. Визначає рівень збудження нейрона;
- функцію активації, визначальну остаточний вихідний рівень нейрона, з яким сигнал збудження (гальмування) надходить на синапси наступних нейронів.

Перед використанням нейромережі проводиться її навчання, що є ітераційним процесом налаштування вагових коефіцієнтів. Для навчання використовуються спеціальні алгоритми. Найбільше розповсюдження отримали градієнтні методи - алгоритм зворотного поширення похибки (Back Propagation), зв'язаних градієнтів, RProp і інші. Основна особливість нейронних мереж полягає в тому, що в процесі навчання вони моделюють складну нелінійну залежність між вхідними і вихідними даними.

Модель штучного нейрона зображено на рисунку 2.5.

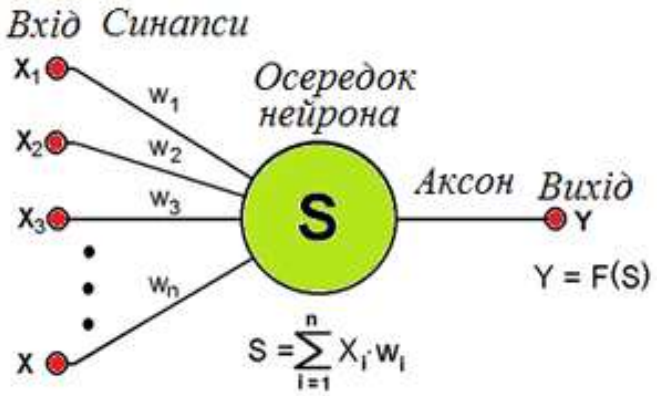

Рисунок 2.5 – Модель штучного нейрона

Розглянемо математичну модель згорткової мережі для класифікації тональності речень. Кількість шарів згорткової мережі обчислюється наступним чином:

$$L = 2a + 2, \text{ де} \tag{2.3}$$

$a \in \mathbb{Z}$ – кількість згорткових шарів.

Вихід кожного згорткового шару зв’язаний з агрегувальним шаром, що пояснює множення кількості згорткових шарів на 2 у формулі 2.5. Також до загальної кількості додається вхідний та вихідний шари.

Розглянемо згортковий шар l. Карта ознак n згорткового шару l обчислюється наступним чином:

$$y_n^l = f_l(x \oplus w_n^l + b_n^l), \text{ де} \quad (2.4)$$

f_l – функція активації шару l

x – вхідний шар,

$\mathbb{w}_n^l = \{w_n^l(i,j)\}$ – n-ий фільтр шару l,

b_n^l– порогові значення, які додаються до карти ознак n шару l,

$\oplus$ – операція двовимірної згортки.

Припустимо, що розмірність вхідного шару дорівнює $H^x \times W^x$, а розмірність фільтра (ядра згортки) дорівнює $r^l \times c^l$, тоді розмірність вихідної карти ознак y_n^l обчислюється наступним чином:

$$y_n^l = (H^x - r^l + 1) \times (W^x - c^l + 1) \quad (2.5)$$

Розглянемо агрегувальний шар a. Карти ознак n, обчислені в згорткових шарах l, подаються на шари агрегації, де розбиваються на блоки розміром $h \times v$. Після чого до кожного блоку застосовується функція агрегування і в результаті отримаємо матрицю $\mathbb{z}_n^a = \{z_n^a(i,j)\}$ розмірністю $(H^x - r^l + 1) - h + 1 \times (W^x - c^l + 1) - v + 1$, елементами якої будуть відповідні значення результатів агрегації блоків. Карта ознак *n* агрегувального шару a обчислюється за формулою 2.8.

$$y_n^a = f_a(z_n^a + b_n^a), \text{ де} \quad (2.8)$$

f_a – функція активації шару a ,

b_n^a– порогові значення, які додаються до карти ознак *n* шару a.

Після проходження карт ознак через шари агрегації вони об’єднуються та подаються на вихідний повнозвязний шар. Розглянемо вихідний повнозвязний шар L. Значення вихідного нейрону обчилюється наступним чином:

$$y_i^L = f_L(\sum_{m=1}^{n} y_m^a + b_n^L), \text{ де} \quad (2.9)$$

f_L – функція активації шару L,

b_n^L– порогові значення, які додаються до карти ознак m шару L.

Таким чином, виходом згорткової нейронної мережі є такий вектор:

$$y = [y_1^L, y_2^L, \dots, y_{N^L}^L]. \quad (2.10)$$

2.4 Аналіз методу навчання згорткової нейронної мережі

Метою навчання нейронної мережі є мінімізація функції втрат протягом певної кількості ітерацій (епох).

Припустимо, що кількість речень, які необхідно класифікувати дорівнює P. Позначимо за Z^P – p-ий текст з множини текстів, а за d_n^p – бажаний результат виходу мережі. Тоді, функція втрат мережі визначається наступним чином:

$$E(w) = -d_n^p * \log y_n^p, \text{ де} \quad (2.11)$$

y_n^p– вихід мережі .

Для навчання мережі буде використано метод зворотного поширення. Це ітеративний градієнтний алгоритм, який використовується з метою мінімізації помилки роботи згорткової мережі та отримання бажаного виходу. Основна ідея цього методу полягає в поширенні сигналів помилки від виходів мережі до її входів, в напрямку, зворотному прямому поширенню сигналів у звичайному режимі роботи.

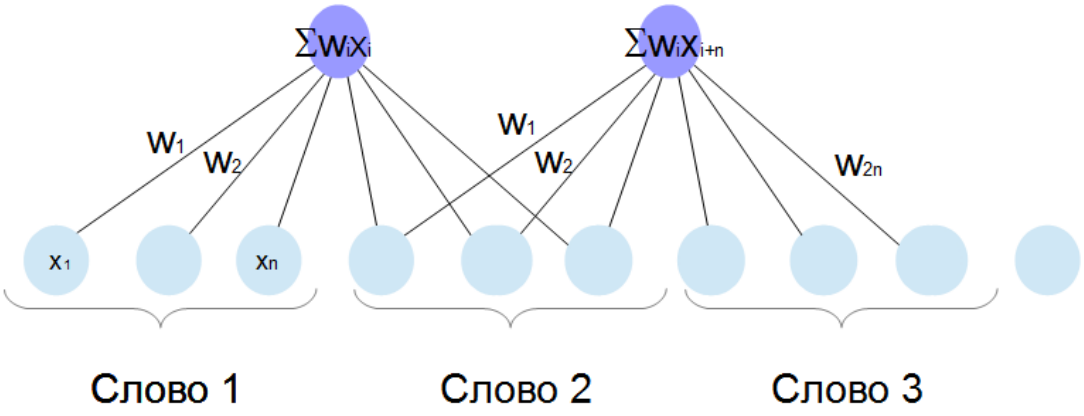

Рисунок 2.6 – Навчання нейронної мережі

На першому кроці навчання мережі вагові коефіцієнти ініціалізуються довільними значеннями. Після цього, сигнал поширюється від попередніх шарів до наступних. На цьому ж кроці обчислюється значення функції втрат вихідного шару.

Другим кроком навчання мережі є поширення помилки, отриманої на першому кроці, в зворотному направленні, тобто від наступних шарів до попередніх. При цьому відбувається корегування ваг і порогових значень кожного нейрона з використанням градієнтного спуска. На кожній ітерації алгоритму зворотного поширення вагові коефіцієнти нейронної мережі модифікуються так, щоб поліпшити рішення одного прикладу. Таким чином, у процесі навчання циклічно вирішуються однокритеріальні задачі оптимізації.

2.5 Структура інформаційної технології класифікації тональності речень на основі згорткової нейронної мережі

Структура інформаційної технології класифікації тональності речень на основі згорткової нейронної мережі, детально описана у попередніх підпунктах, зображена на рис. 2.7.

Для роботи інформаційної технології класифікації тональності речень на основі згорткової нейронної мережі вхідною є інформація у вигляді тексту

(ASCII коди символів). Потім здійснюється процес векторизації слів тексту [10] за методом word2vec [11]. Класифікація тональності речень буде виконуватись згортковою нейронною мережею на основі попереднього її навчання на основі відповідної навчальної вибірки. Тому відбуваться процес створення цієї навчальної вибірки речень. Далі відбувається процес формування (ініціалізації) та навчання згорткової нейронної мережі, а потім процес подачі векторизованого речення на вхід згорткової нейромережі. Далі відбувається процес класифікації тональності речення згортковою нейронною мережею, а потім - процес виведення результату класифікації.

Рисунок 2.7 - Структура інформаційної технології класифікації тональності речень на основі згорткової нейронної мережі

Таким чином, розроблена структура інформаційної технології класифікації тональності речень на основі згорткової нейронної мережі може бути використана для подальшої розробки програмних засобів.

2.6 Планування процесу розробки прикладного програмного забезпечення

Планування робіт це найбільш важлива частина управління проектом [12]. Основною цілю планування – визначення ґрунтовних цілей проекту:

– витрати;
– календарний план;
– рівень якості.

Проект буде вважатися успішним, якщо він досягнув поставлених цілей по грошам, термінам і якості. Без детального планування не є можливим відслідковування і оперативне управління ходом виконання проекту. Ніякі технічні зусилля, що будуть прикладені пізніше, не зможуть компенсувати недолік ретельного планування. Недолік планування великого проекту з розробки ПЗ практично являється гарантією його невдалого виконання.

Вхідними даними для планування проекту розробки ПЗ являються:

– специфікація вимог до ПЗ;
– архітектура ПЗ, що розроблюється.

Для планування не вимагаються детальні вимоги до ПЗ, але повинні бути відомі всі важливі вимоги. Також бажаним являється наявність ключових архітектурних рішень На рисунку 2.8 наочно зображено процеси планування проекту.

В ході планування проекту вирішуються наступні задачі:

– оцінка трудомісткості проекту;
– планування термінів виконання проекту і складу виконавців;
– управління якістю;

– управління ризиками;

– планування моніторингу виконання проекту.

Для планування проекту розробки ПЗ важливою обов’язковою умовою є загальна оцінка трудомісткості і загальна оцінка термінів виконання робіт. Такі оцінки вимагаються до того, як процес розробки буде запущеним, так як вони визначають цілі проекту по вартості та по термінах.

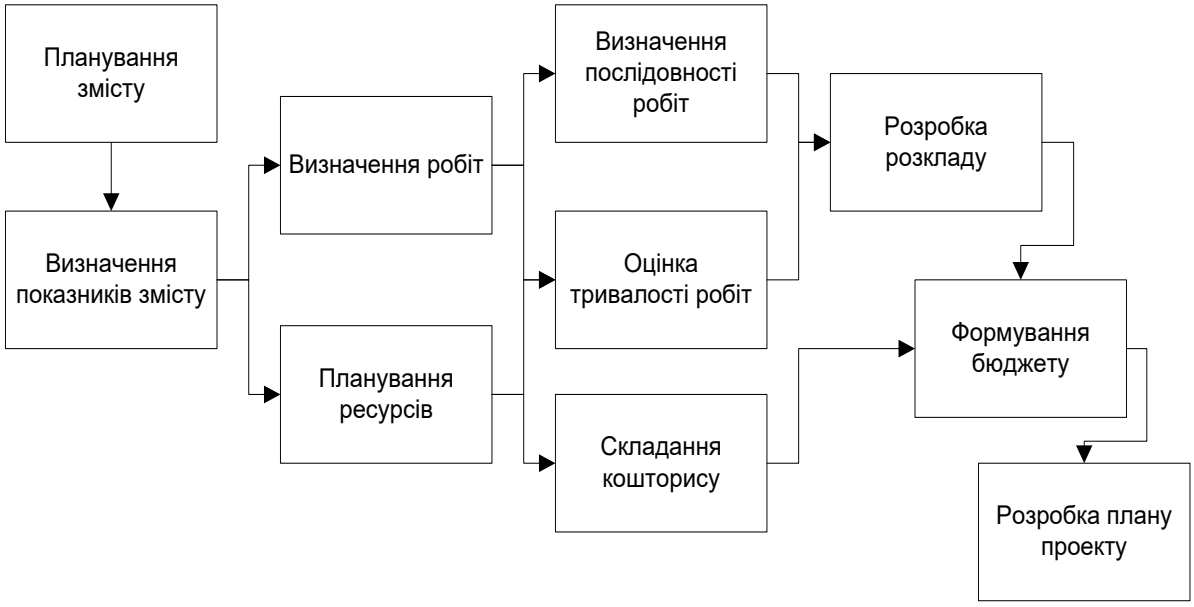

Рисунок 2.8 – Процеси планування розробки програмного забезпечення

Після визначення трудомісткості необхідно визначити планові терміни поставки ПЗ (календарний план). Календарне планування – це процес визначення того, як робота в проекті буде організована у вигляді набору окремих задач і, коли і як ці задачі будуть виконуватися. Протягом календарного планування необхідно оцінити час, що потрібний на виконання кожної задачі, необхідну трудомісткість, хто буде працювати над поставленою задачею. Також необхідно оцінити ресурси, що необхідні для завершення кожної задачі, такі як:

- кількість працівників;
- дисковий простір на сервері;
- час, що потребується на спеціалізованих технічних пристроях і т.п.

План якості – це набір робіт, що призначений для досягнення потрібної якості ПЗ. Для планування якості спочатку потрібно зрозуміти цикл внесення і видалення дефектів, так як саме дефекти визначають якість кінцевого ПЗ. Розробка ПЗ в значній степені зв’язана з діяльністю людей і тому має велику кількість помилок.

До робіт з управління якості відносяться:

- перевірка вимог;
- перевірка результатів проектування;
- перевірка коду;
- тестування одиниць коду;
- інтегрування тестування;
- системне тестування;
- тестування і т.п.

Програмний проект є складним заходом, який потребує створення ПЗ високої якості в задані терміни і в рамках заданого бюджету. На виконання програмного проекту негативно впливають непередбачувані подій. Управління ризиками – це спроба мінімізувати ймовірність вдачі, викликаної незапланованими подіями. Управління ризиками це не відмова від проекту, який має ризики, а робота з передбачення, виявленню і мінімізації ризиків в ході виконання проекту.

План управління проектом це просто документ, який може бути використаний для відслідковування ходу виконання проекту. Любий хороший план є даремним, якщо його не дотримуватися певним чином. Виконання проекту не може бути правильно направлене згідно плану, якщо воно ретельно не відслідковується (не відбувається моніторинг) і реальні результати не порівнюються з даним планом.

Microsoft Project – це програма керування проектами від компанії Microsoft. MS Project створений для полегшення роботи, пов’язаної із розробкою планів, розподілу ресурсів по задачам, відстеженні прогресу та аналізі об’ємів роботи. MS Project дозволяє, зокрема:

- визначати оптимальний склад ресурсів проекту і розподіл їх у часі при відомому плановому завантаженні та кількісному складі персоналу;
- проаналізувати ризики і визначити необхідні резерви надійності;
- обрахувати необхідний бюджет проекту та запланувати витрати у часі;
- розрахувати розподіл потреб проекту у матеріалах та обладнанні у часі;
- моделювати прийняття рішень із зміни параметрів проекту;
- аналізувати відхилення фактичного ходу виконання робіт від запланованого;
- вести облік та архів проектів, використовуючи минулий досвід для реалізації наступних проектів.

Детальне планування процесу розробки програми класифікації тональності речень. Від коректного вирішення задачі планування розробки програмного забезпечення доволі сильно залежить успіх проекту. Так невдале планування часу та ресурсів ставить під загрозу виконання проекту.

При розробці програми класифікації тональності речень є:

1) Опис вимог до системи та складання технічного завдання. Скласти основні вимоги до системи автоматизованого планування прикладного ПЗ, проаналізувавши вимоги ринку до подібних систем. Розглянути рецензії на конкуруючі продукти, їх сильні сторони та претензії користувачів цих систем, щоб запобігти виникнення подібних помилок.

2) Розробка математичної моделі програми класифікації тональності речень. Під час цього етапу відбуваються спроби застосувати існуючі математичні апарати для розв’язання поставленої задачі, висуваються аргументи за і проти кожного кандидату, проводяться порівняльні дослідження.

3) Підготовка робочих місць. Простий, але важливий етап. На цьому етапі організуються робочі місця розробників, налаштовуються комп’ютери, встановлюється спеціалізоване програмне забезпечення та інші дрібні дії, які потенційно заощаджують час під час розробки

4) Проектування архітектури програми – потенційно найважливіший етап. Правильна архітектура значно спрощує як розробку програмного забезпечення, так і його тестування та подальшу підтримку. Неправильна ж архітектура, з іншого боку, заважає зосередитись на дійсно важливих речах під час розробки і створює більше проблем, аніж вирішує.

5) Вибір засобів програмування – на основі відомої математичної моделі та з огляду на затверджену архітектуру обираються програмні засоби та пакети, які зроблять розробку найбільш швидкою та продуктивною, а подальше використання продукту – ефективним.

6) Програмна реалізація алгоритму – на цьому етапі створюється програми, яка потенційно має помилки і не забезпечує захист від помилок користувача – головне підтвердити правильність вибору архітектури та здатність алгоритму вирішувати поставлені задачі.

7) Тестування розробленої програми на реальних та контрольних вибірках. На цьому етапі виявляються помилки роботи алгоритму та математичної моделі. Цей етап є вкрай важливим, адже саме його ретельне виконання гарантує коректність роботи усієї системи в цілому.

8) Випуск реліз-версії – оформлення продукту у тому вигляді, в якому його отримає користувач.

2.7 Висновок

В даному розділі було обгрунтовано вибір згорткової нейронної мережі для побудови інформаційної технології класифікації тональності речень, визначено архітектуру згорткової нейронної мережі, математичну модель штучного нейрона та загальну математичну модель згорткової мережі. Для класифікації тональності речень використано три згорткових шари з фільтрами. Кількість фільтрів для кожного згорткового шару – 16. Функція, яка буде використана в шарах агрегування – максимізаційна (max-pooling).

Для згорткових та агрегувальних шарів буде використана функція активації – ReLU. Метод, за яким буде проходити навчання мережі – це метод зворотного поширення помилки. У підсумку було розроблено структуру інформаційної технології класифікації тональності речень на основі згорткової нейронної мережі та розглянуто планування процесу розробки прикладного програмного забезпечення.

3 ПРОГРАМНА РЕАЛІЗАЦІЯ ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ КЛАСИФІКАЦІЇ ТОНАЛЬНОСТІ РЕЧЕНЬ НА ОСНОВІ НЕЙРОННОЇ МЕРЕЖІ

3.1 Вибір мови програмування

Можливості мови програмування залежать від сфери застосування та наявності альтернативних реалізацій. Наприклад на асемблері найпростіше керувати апаратними ресурсами, а на php зручно зверстати сторінку для веб.

Не для кожної мови програмування існує зручне середовище програмування. Слід також взяти до уваги тип ліцензії та ціну середовища програмування. Ні розрекламованість ні популярність середовища програмування не може бути показником його досконалості.

Python (найчастіше вживане прочитання — «Па́йтон», запозичено назву з британського шоу Монті Пайтон) — інтерпретована об'єктно-орієнтована мова програмування високого рівня з строгою динамічною типізацією. Розроблена в 1990 році Гвідо ван Россумом. Структури даних високого рівня разом із динамічною семантикою та динамічним зв'язуванням роблять її привабливою для швидкої розробки програм, а також як засіб поєднання існуючих компонентів. Python [13] підтримує модулі та пакети модулів, що сприяє модульності та повторному використанню коду. Інтерпретатор Python та стандартні бібліотеки доступні як у скомпільованій так і у вихідній формі на всіх основних платформах. В мові програмування Python підтримується кілька парадигм програмування, зокрема: об'єктно-орієнтована, процедурна, функціональна та аспектно-орієнтована. Серед основних її переваг можна назвати такі:

- чистий синтаксис (для виділення блоків слід використовувати відступи);
- переносність програм (що властиве більшості інтерпретованих мов);

- стандартний дистрибутив має велику кількість корисних модулів (включно з модулем для розробки графічного інтерфейсу);
- можливість використання Python в діалоговому режимі (дуже корисне для експериментування та розв'язання простих задач);
- стандартний дистрибутив має просте, але разом із тим досить потужне - середовище розробки, яке зветься IDLE і яке написане на мові Python;
- зручний для розв'язання математичних проблем (має засоби роботи з комплексними числами, може оперувати з цілими числами довільної величини, у діалоговому режимі може використовуватися як потужний калькулятор);
- відкритий код (можливість редагувати його іншими користувачами).

Python має ефективні структури даних високого рівня та простий, але ефективний підхід до об'єктно-орієнтованого програмування. Елегантний синтаксис Python, динамічна обробка типів, а також те, що це інтерпретована мова, роблять її ідеальною для написання скриптів та швидкої розробки прикладних програм у багатьох галузях на більшості платформ.

Інтерпретатор мови Python і багата Стандартна бібліотека (як вихідні тексти, так і бінарні дистрибутиви для всіх основних операційних систем) можуть бути отримані з сайту Python www.python.org, і можуть вільно розповсюджуватися. Цей самий сайт має дистрибутиви та посилання на численні модулі, програми, утиліти та додаткову документацію.

Інтерпретатор мови Python може бути розширений функціями та типами даних, розробленими на C чи C + + (або на іншій мові, яку можна викликати із C). Python також зручна як мова розширення для прикладних програм, що потребують подальшого налагодження.

Для розробки програмних модулів на магістерську роботу було обрано мову програмування Pyton. Розробка мови Python була розпочата в кінці 1980-х років співробітником голландського інституту CWI Гвідо ван Россумом. Для

розподіленої ОС Amoeba потрібна була розширювана скриптова мова, і Гвідо почав писати Python на дозвіллі, запозичивши деякі напрацювання для мови ABC (Гвідо брав участь у розробці цієї мови, орієнтованої на навчання програмуванню). У лютому 1991 року Гвідо опублікував вихідний текст в групі новин alt.sources. Мова почала вільно поширюватися через Інтернет і сподобалася іншим програмістам. З 1991 року Python є цілком об'єктно-орієнтованим. Python також запозичив багато рис таких мов, як C, C++, Modula-3 і Icon, й окремі риси функціонального програмування з Ліспу.

Python портований і працює майже на всіх відомих платформах — від КПК до мейнфреймів. Існують порти під Microsoft Windows, всі варіанти UNIX (включаючи FreeBSD та GNU/Linux), Plan 9, Mac OS та Mac OS X, iPhone OS 2.0 і вище, Palm OS, OS/2, Amiga, AS/400 та навіть OS/390[en], Symbian та Android.

У міру старіння платформи її підтримка в основній гілці мови припиняється. Наприклад, з версії 2.6 припинена підтримка Windows 95, Windows 98 та Windows ME. Однак на цих платформах можна використовувати попередні версії Python — спільнота активно підтримує версії Python починаючи від 2.3 (для них виходять виправлення).

При цьому, на відміну від багатьох портованих систем, для всіх основних платформ Python має підтримку характерних для даної платформи технологій (наприклад, Microsoft COM/DCOM). Навіть більше, існує спеціальна версія Python для віртуальної машини Java — Jython, що дозволяє інтерпретатору виконуватися на будь-якій системі, яка підтримує Java, при цьому класи Java можуть безпосередньо використовуватися з Python й навіть бути написаними на ньому. Також кілька проектів забезпечують інтеграцію з платформою Microsoft.NET, основні з яких — IronPython та Python.Net.

Python підтримує динамічну типізацію, тобто, тип змінної визначається лише під час виконання. З базових типів слід зазначити підтримку цілих чисел довільної довжини і комплексних чисел. Python має багату бібліотеку для роботи з рядками, зокрема, кодованими в юнікоді.

З колекцій Python підтримує кортежі (tuples), списки (масиви), словники (асоціативні масиви) і від версії 2.4, множини.

Система класів підтримує множинне успадкування і метапрограмування. Будь-який тип, включаючи базові, входить до системи класів, й за необхідності можливе успадкування навіть від базових типів.

Отже вибір однозначний, для розробки буде використано Python [13].

3.2 Граф схема алгоритму

Граф-схема алгоритму (ГСА) — кінцевий зв'язний орієнтований граф відповідають операторам, а дуги задають порядок проходження вершин (операторів) алгоритму, де число вершин графа - число дуг. У ширшому сенсі вершинам графа відповідають не тільки операторні вершини, а й умовні, початкова та кінцева вершини і т.д. При розгляді паралельних алгоритмів вводиться поняття 'паралельної граф-схеми алгоритму, до складу якої входять вершини розпаралелювання / синхронізації, функціональність яких зазвичай поєднується. Іноді до складу ГСА вводяться вершини додаткових типів: об'єднання альтернативних дуг (парна вершина для умовної вершини), фіктивні операторні вершини, вершини маркування (з метою забезпечення можливості моделювання виконання алгоритму мережею Петрі), очікувальні вершини.

Однак не будь-який орієнтований граф, складений з вершин зазначених вище типів, може бути ототожнений з коректним алгоритмом. Наприклад, з операторної вершини не може виходити більше однієї дуги. Тому на практиці зазвичай обмежуються розглядом підкласу граф-схем алгоритмів, що задовольняють умовам безпеки, живучості і стійкості. Алгоритми перетворення ГСА, які є підмножиною алгоритмів обробки графів загального вигляду, часто мають суттєві відмінності через використання особливих властивостей ГСА, що дозволяє їх спрощення, зниження часової або об'ємної складності

До складу граф-схеми алгоритму можуть бути виділені більші елементи, представлені підмножинами її вершини і дуг: гілки (лінійні ланцюжки або

ділянки вершин) і фрагменти (початковий, паралельний, альтернативний, циклічні з перед-, постумовою і перериванням). Еквівалентним записом граф-схеми коректного алгоритму є дерево фрагментів, що відбиває порядок вкладеності фрагментів.

Граф схема алгоритму роботи програми класифікації тональності речень зображена на рисунку 3.1.

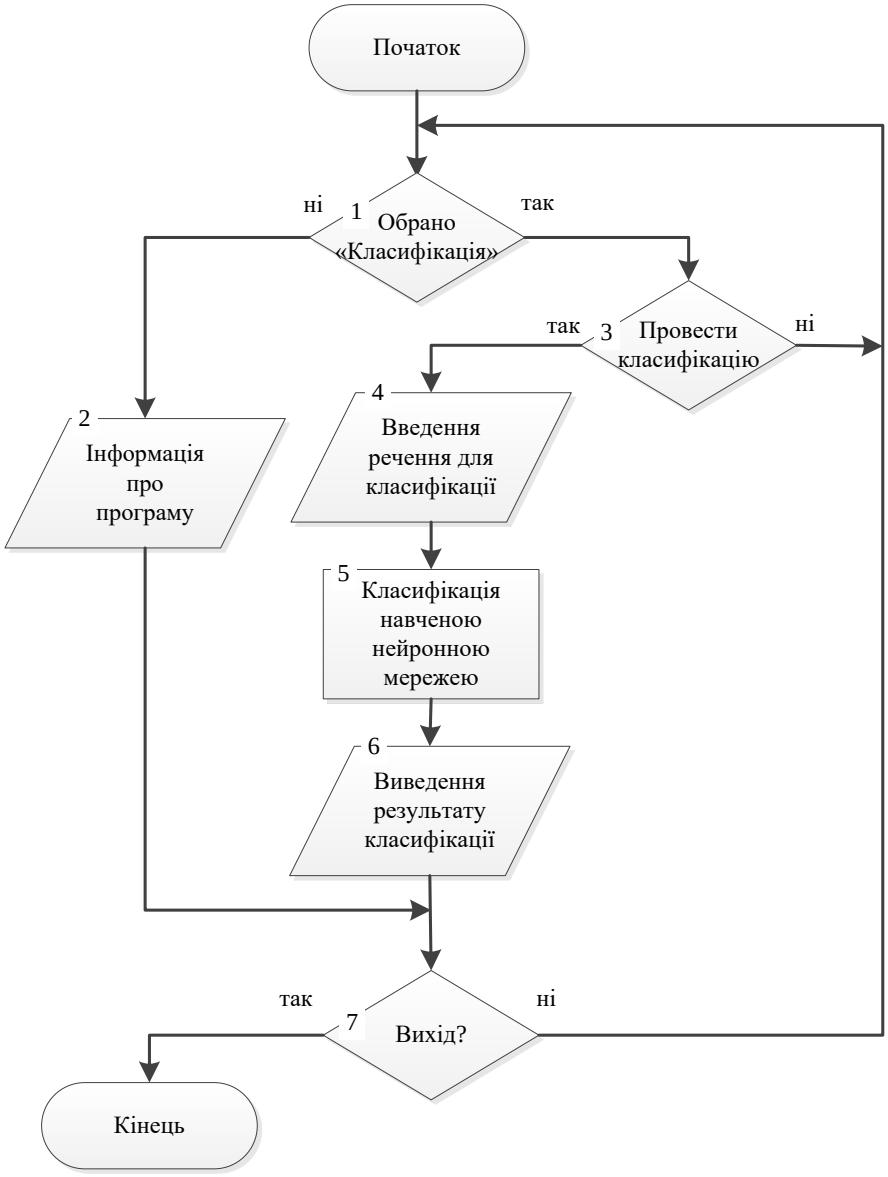

Рисунок 3.1 – Граф-схема алгоритму роботи програми класифікації тональності речень

Опис вершин алгоритму у граф-схемі програми класифікації тональності речень:

Вершина 1 – після запуску програми потрібно обрати або вкладку «Інформація» (за замовчанням) або вкладку «Класифікація». При виборі вкладки «Класифікація» виконується:

Вершина 2 – Виводиться на екран інформація про програму.

Вершина 3 – Провести класифікацію чи повернутися до головного меню.

Вершина 4 – Ввести текст чи речення для класифікації тональності речень.

Вершина 5 – Натискання кнопки класифікувати, після чого відбувається процес класифікації тональності речення попередньо навченою згортковою нейронною мережею.

Вершина 6 – Виведення результату класифікації.

Вершина 7 – вибір чи потрібно вийти із програми.

3.3 Використання фреймворків та бібліотек

Фреймворк – структура програмної системи; програмне забезпечення, що полегшує розробку і об'єднання різних компонентів великого програмного проекту. Вживається також слово «каркас», а деякі автори використовують його в якості основного, в тому числі не базуючись взагалі на англомовному аналогу. Можна також говорити про каркасний підході як про підхід до побудови програм, де будь-яка конфігурація програми будується з двох частин: перша, постійна частина - каркас, не змінний від конфігурації до конфігурації і несе в собі гнізда, в яких розміщується друге, змінна частина - змінні модулі (або точки розширення).

Фреймворк відрізняється від поняття бібліотеки тим, що бібліотека може бути використана в програмному продукті просто як набір підпрограм близької функціональності, не впливаючи на архітектуру програмного продукту і не накладаючи на неї ніяких обмежень. У той час як фреймворк диктує правила

побудови архітектури додатку, задаючи на початковому етапі розробки поведінка за замовчуванням, каркас, який потрібно буде розширювати і змінювати згідно зазначеним вимогам.

Одним з головних переваг при використанні каркасних додатків є те, що такі програми мають стандартну структуру. Каркасні додатки стали популярними з появою графічних інтерфейсів користувача, які мали тенденцію до реалізації стандартної структури для додатків. З їх використанням стало набагато простіше створювати засоби для автоматичного створення графічних інтерфейсів, так як структура внутрішньої реалізації коду програми стала відома заздалегідь. Для забезпечення каркаса зазвичай використовуються техніки об'єктно-орієнтованого програмування (наприклад, частини програми можуть успадковуватися від базових класів фреймворка).

Sclit-learn – scikit-learn використовує Python для математичної та наукової роботи. scikit-learn включає інструменти для багатьох стандартних задач машинного навчання (таких як кластеризація, класифікація, регресія і т. д.). І так як scikit-learn розробляється великим співтовариством розробників і експертів з машинного навчання, перспективні нові методи, як правило, включаються в досить короткий термін.

web2py - фреймворк з відкритим вихідним кодом для розробки веб додатків, написаний на мові програмування Python. Web2py дозволяє веб розробникам створювати динамічні сайти використовуючи Python. Web2py покликаний скоротити рутинні процеси веб розробки, такі як написання веб форм з нуля, хоча розробник може розробити форму з нуля, якщо в цьому виникне необхідність.

3.4 Програмна реалізація процесу навчання нейронної мережі

Для навчання нейронної мережі спочатку необхідно завантажити датасет з текстами та на основі датасету сформувати кінцевий масив міток класів.

Фрагмент функції, яка завантажує датасет та повертає вихідний масив текстів та міток класів до яких вони належать, та масив всіх можливих міток класів.

Після того, як датасет був завантажений, потрібно розбити його на навчальну та тестову вибірки у пропорції 80% : 20%.

```
x_, x_test, y_, y_test = train_test_split(x, y, test_size=0.2)
```

Далі необхідно створити об’єкт згорткової нейронної мережі з бібліотеки та ініціалізувати його:

```
cnn = TextCNN(   sequence_length=x_train.shape[1],
  num_classes=y_train.shape[1],
  vocab_size=len(vocab_processor.vocabulary_),
  embedding_size=params['embedding_dim'],
  filter_sizes=list(map(int, params['filter_sizes'].split(","))),
  num_filters=params['num_filters')
```

Перший та другий параметр об’єкту TextCNN – це розмірність вхідних даних. Третій параметр вказує на кількість слів у словнику word2vec [11] Четвертий параметр вказує на довжину вектора, в який буде перетворено кожне слово вхідного тексту. П`ятий параметр – це масив з розмірністю згорткових фільтрів (3 × 100, 4 × 100 та 5 × 100). Шостий параметр вказує на кількість фільтрів (16 для кожного згорткового шару).

Після ініціалізації мережі починається процес навчання. Процес навчання складається з певної кількості ітерацій. На кожній ітерації обирається деякий текст з навчальної вибірки та подається на вхід нейронної мережі.

Процес навчання мережі складається з повторення даного фрагменту коду:

```
optimizer = tf.train.AdamOptimizer(1e-3)
grads_and_vars = optimizer.compute_gradients(cnn.loss)
train_op = optimizer.apply_gradients(grads_and_vars,
global_step=global_step)
def train_step(x_batch, y_batch):
  feed_dict = {
    cnn.input_x: x_batch,
    cnn.input_y: y_batch,
    cnn.dropout_keep_prob: params['dropout_keep_prob']}
  _, step, loss, acc = sess.run([train_op, global_step, cnn.loss, cnn.accuracy],
feed_dict)
```

Дана функція на кожній ітерації обчислює значення виходу нейронної мережі на основі тексту з навчальної вибірки та на основі порівняння значення виходу нейронної мережі з бажаним результатом проводить корекцію ваг мережі на основі методу стохастичного градієнтного спуску [14].

3.5 Програмна реалізація класифікації тональності речень

WebAPI складається з набору GET та POST методів, які можна викликати через HTTP-протокол. Основним методом API є POST-метод . Тілом даного метода є текст, який потрібно класифікувати.

Першим кроком є імпорт файлу з параметрами нейронної мережі [15] та файу з вагами нейронної мережі, який був отриманий в процесі навчання нейронної мережі.

Після цього буде ініціалізовано об’єкт нейронної мережі та обчислено значення ймовірностей належності тексту до кожного з класів. В результату буде сформовано відповідь та повернено програмі-клієнту:

```
resp = Response(labels[int(all_predictions[0])])
resp.headers['Access-Control-Allow-Origin'] = '*'
return resp
```

Для того, щоб перевірити роботу WebAPI було створено клієнтський веб-додаток, який приймає текст, введений користувачем, робить запит на сервер, та повертає результат. Фрагмент функції, яка виконує дану процедуру:

```
classify() {
   if (this.wordCount == 0) {
      this.snackBar.open('Введіть текст для класифікації!', null, { duration: 3000 });
      return;
    }

    this.api.request({
      body: this.textToClassify,
      method: 'POST',
      url: 'http://localhost:5000/api/predict',
      withAuthorization: true,
      responseType: 'text'
    })
      .subscribe(res => {
        this.dialog.open(ResultDialogComponent, {
          width: '250px',
          data: {
            class: res}});
      }, err => {
        this.snackBar.open('Під час класифікації сталась помилка. Спробуйте пізніше.', null, { duration: 3000 });
```

```
});
```

Головну сторінку веб-додатку зображено на рисунку 3.2.

Рисунок 3.2 – Головна сторінка веб-додатку

3.6 Тестування та аналіз результатів роботи програми класифікації тональності речень

Розроблювана програма працює на серверній стороні. Для того, щоб отримати доступ для класифікації спочатку необхідно перейти до нього у меню.

Web-додатки зазвичай взаємодіють з користувачем за допомогою елементів форм [16] і змінних GET або POST. У разі використання методу GET всі значення, передані додатком можуть бути видні безпосередньо в URL, в той час як у випадку POST-запитів для визначення того, що вводить користувач, часто доводиться вивчити вихідний текст сторінки, що містить форму.

Під час тестування програми були отримані позитивні результати, не було помічено збоїв в роботі.

Так, приклад позитивного речення наведено на рис. 3.2, а результат його класифікації – на рис. 3.3.

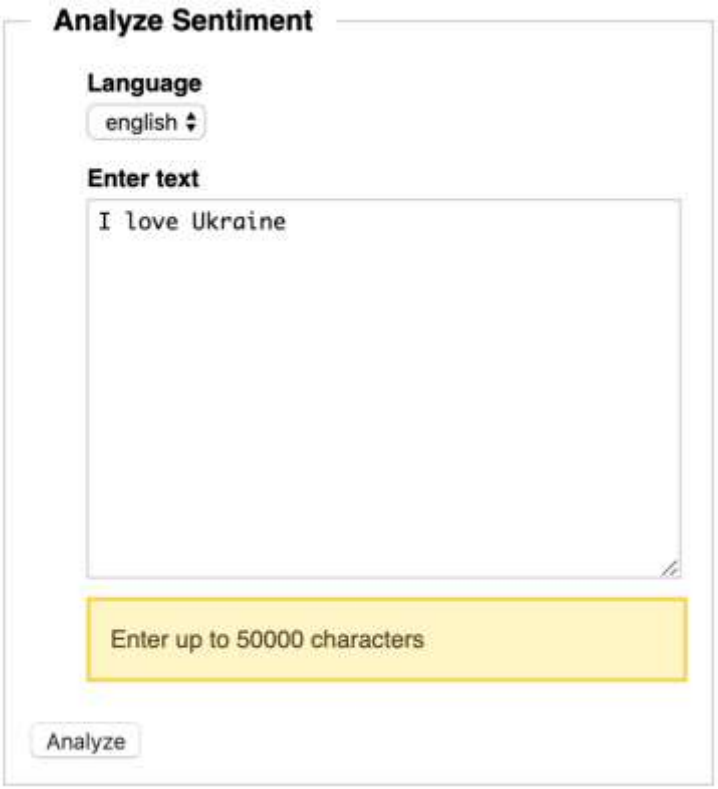

Рисунок 3.2 – Вікно введення даних для класифікації тональності позитивного речення

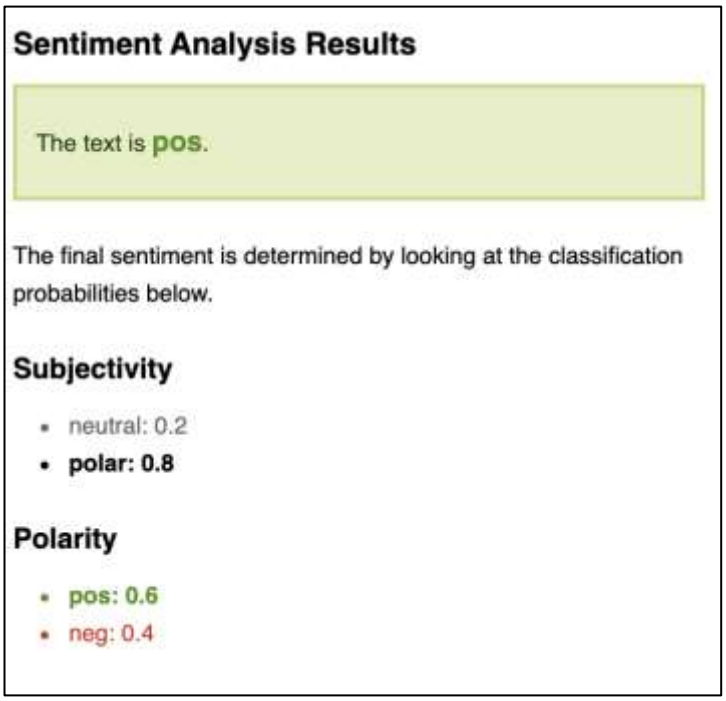

Рисунок 3.3 – Вікно виведення результатів класифікації позитивного речення

Приклад негативного речення наведено на рис. 3.4, а результат його класифікації – на рис. 3.5.

Рисунок 3.4 – Вікно введення даних для класифікації тональності негативного речення

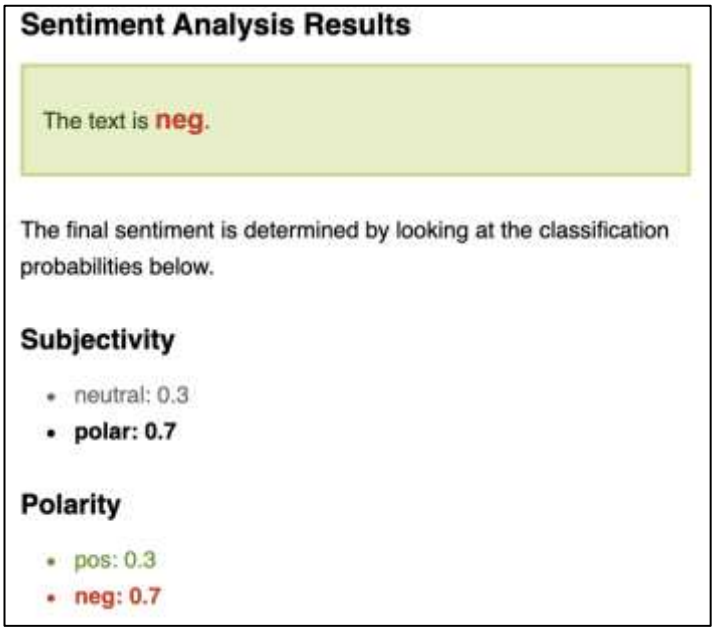

Рисунок 3.5 – Вікно виведення результатів класифікації негативного речення

Приклад нейтрального речення наведено на рис. 3.6, а результат його класифікації – на рис. 3.7.

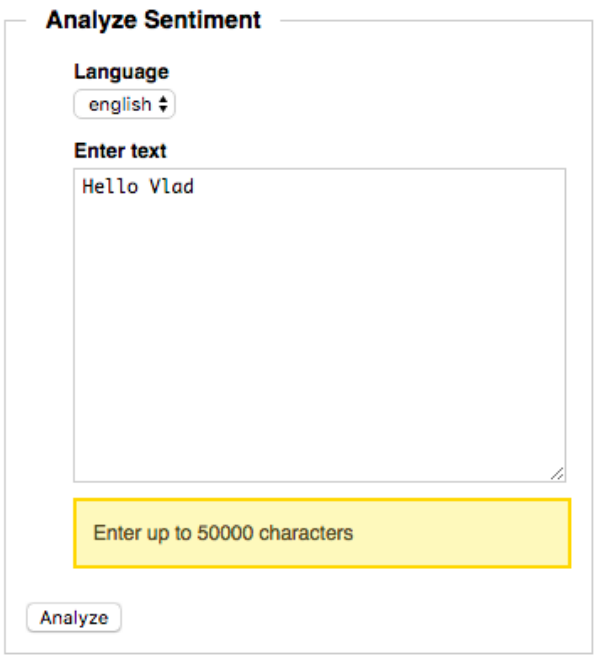

Рисунок 3.6 – Вікно введення даних для класифікації тональності нейтрального речення

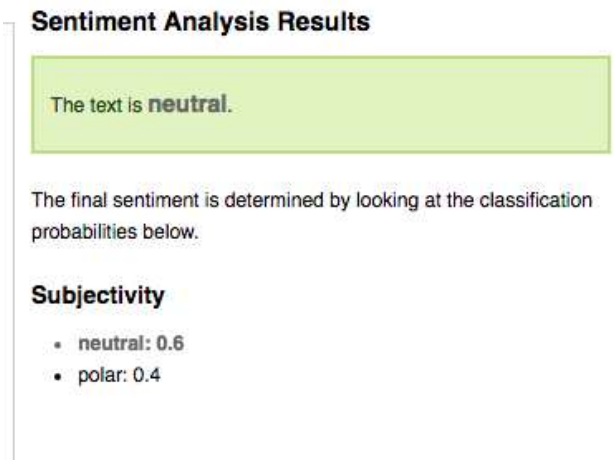

Рисунок 3.7 – Вікно виведення результатів класифікації нейтрального речення

Для тестування якості роботи програми класифікації тональності речень взяли загально доступну в мережі Інтернет базу даних позитивних і негативних речень про кінофільми [17] англійською мовою. Обсяг навчальної вибірки був 1000 речень (350 позитивних, 350 негативних і 300 нейтральних). Обсяг тестової вибірки був 200 речень (70 позитивних, 70 негативних і 60 нейтральних). Порівняння результатів роботи розробленої програми класифікації тональності речень з програмами-аналогами наведено у таблиці 3.1.

Таблиця 3.1 – Порівняння результатів роботи програми класифікації текстів з програмами-аналогами

Характеристика | Засіб	«Twitter Sentiment»	«I-Teko»	Розроблена програма класифікації тональності речень
Достовірність класифікації	79,5%	81%	85,5%

Достовірність класифікації тональності речень визначається за формулою:

$$accuracy = \frac{Nt}{Nall} * 100\% \,, \tag{3.1}$$

де Nt — кількість правильно класифікованих речень тестової вибірки,

$Nall$ — загальна кількість речень тестової вибірки,

Із табл. 3.1 видно, що розроблена програма має вищу на 6% достовірність класифікації тональності речень (85.5%), ніж перша аналогічна програма (79.5%) та вищу на 4,5% достовірність класифікації тональності речень (85.5%), ніж друга аналогічна програма (81%), а значить достовірність класифікації

тональності речень покращена як мінімум на 4,5%, тобто мета роботи досягнута.

Отже, після проведення порівняння розробленої програми класифікації тональності речень на основі згорткової нейронної мережі з аналогами, можна сказати, що розроблена програма має переваги у достовірності класифікації.

Інструкцію до користування програмою наведено в додатку А, деякі ілюстрації до програми (у т.ч. скріншоти) наведено в додатку В.

3.9 Висновок

В ході практичної реалізації інформаційної технології класифікації тональності речень у даному розділі було обрано такі мови програмування: для програми класифікації тональності речень – Python, для програмної реалізації клієнтського додатку для тестування роботи WebAPI та : web2py та sclit-learn. Також було описано роботу основних частин програми та наведено відповідні фрагменти коду. В результаті було розроблено програмне забезпечення класифікації тональності речень на основі згорткової нейронної мережі, яке порівняно з аналогом має кращу на 4,5% достовірність класифікації тональності речень. Таким чином, мета роботи досягнута – достовірність класифікації тональності речень підвищена. Розроблена програма повністю відповідає завданню, що підтверджується її тестуванням.

4 ЕКОНОМІЧНА ЧАСТИНА

4.1 Оцінювання комерційного потенціалу розробки

Метою проведення технологічного аудиту є оцінювання комерційного потенціалу розробки [18]. Для проведення технологічного аудиту було залучено 2-х незалежних експертів. Такими експертами будуть Колесницький О.К. та Перевозніков С.І.

Здійснюємо оцінювання комерційного потенціалу розробки за 12-ма критеріями за 5-ти бальною шкалою.

Результати оцінювання комерційного потенціалу розробки наведено в таблиці 4.1.

Таблиця 4.1 – Результати оцінювання комерційного потенціалу розробки

Критерії	Прізвище, ініціали, посада експерта	
	1. Колесницький О.К.	2. Перевозніков С.І.
	Бали, виставлені експертами:	
1	4	4
2	3	3
3	3	4
4	4	3
5	3	4
6	4	4
7	3	3
8	4	4
9	4	3
10	4	3
11	3	4
12	3	4
Сума балів	$СБ_1 = 43$	$СБ_2 = 43$
Середньоарифметична сума балів $\overline{СБ}$	$\overline{СБ} = \frac{\sum_{1}^{3} СБ_i}{2} = 43$	

Отже, з отриманих даних таблиці 4.1 видно, що нова розробка має високий рівень комерційного потенціалу.

4.2 Прогнозування витрат на виконання науково-дослідної роботи та конструкторсько–технологічної роботи.

Для розробки нового програмного продукту необхідні такі витрати [18].

Основна заробітна плата для розробників визначається за формулою (4.1):

$$З_о = \frac{М}{Т_р} \cdot t, \qquad (4.1)$$

де М- місячний посадовий оклад конкретного розробника;

$Т_р$ - кількість робочих днів у місяці, $Т_р$ = 21 день;

t - число днів роботи розробника, t = 45 днів.

Розрахунки заробітних плат для керівника і програміста наведені в таблиці 4.2.

Таблиця 4.2 – Розрахунки основної заробітної плати

Працівник	Оклад М, грн.	Оплата за робочий день, грн.	Число днів роботи, t	Витрати на оплату праці, грн.
Науковий керівник	5500	261,90	10	2619
Інженер-програміст	4000	190,47	45	8571,15
Всього:				11190,15

Розрахуємо додаткову заробітну плату:

$$З_{дод} = 0,1 \cdot 11190,15 = 1119,01 (грн.)$$

Нарахування на заробітну плату операторів НЗП розраховується як 37,5…40% від суми їхньої основної та додаткової заробітної плати:

$$Н_{зп}= (З_о + З_р)\cdot \frac{\beta}{100}, \quad (4.2)$$

$$Н_{зп}= (11190,15+1119,01)\cdot \frac{36,3}{100} = 4468,22 \text{ (грн.)}.$$

Розрахунок амортизаційних витрат для програмного забезпечення виконується за такою формулою:

$$А=\frac{Ц\cdot Н_а}{100}\cdot \frac{Т}{12}, \quad (4.3)$$

де Ц – балансова вартість обладнання, грн;

$Н_а$ – річна норма амортизаційних відрахувань % (для програмного забезпечення 25%);

Т – Термін використання (Т=3 міс.).

Таблиця 4.3 – Розрахунок амортизаційних відрахувань

Найменування програмного забезпечення	Балансова вартість, грн.	Норма амортизації, %	Термін використання, міс.	Величина амортизаційних відрахувань,грн
Персональний комп'ютер	9000	25	2	375
Всього:				375

Розрахуємо витрати на комплектуючі. Витрати на комплектуючі розрахуємо за формулою:

$$K = \sum_1^n Н_і \cdot Ц_і \cdot К_і, \quad (4.4)$$

де n – кількість комплектуючих;

Ні - кількість комплектуючих і-го виду;

Ці – покупна ціна комплектуючих і-го виду, грн;

Кі – коефіцієнт транспортних витрат (приймемо Кі = 1,1).

Таблиця 4.4 - Витрати на комплектуючі, що були використані для розробки ПЗ.

Найменування матеріалу	Одиниці виміру	Ціна, грн.	Витрачено	Вартість витрачених матеріалів, грн.
Флешка	шт.	180	1	180
Пачка паперу	уп.	130	1	130
Ручка	шт.	10	1	10
Всього з урахуванням транспортних витрат				352

Витрати на силову електроенергію розраховуються за формулою:

$$В_е=В\cdot П\cdot Ф\cdot К_п\ ; \qquad (4.5)$$

де В – вартість 1кВт-години електроенергії (В=1,7 грн/кВт);

П – установлена потужність комп'ютера (П=0,6кВт);

Ф – фактична кількість годин роботи комп'ютера (Ф=180 год.);

$К_п$ – коефіцієнт використання потужності ($К_п< 1$, $К_п = 0,8$).

$$В_е= 1,7 \cdot 0,6 \cdot 180 \cdot 0,8 = 146,88 \text{ (грн.)}$$

Розрахуємо інші витрати $В_{ін}$.

Інші витрати $І_в$ можна прийняти як (100…300)% від суми основної заробітної плати розробників та робітників, які були виконували дану роботу, тобто:

$$В_{ін} = (1..3) \cdot (З_о + З_р). \qquad (4.6)$$

Отже, розрахуємо інші витрати:

$$В_{ін} = 1 * (11190,15 + 1119,01) = 12309,16 \text{ (грн).}$$

Сума всіх попередніх статей витрат дає витрати на виконання даної частини роботи:

$$В = З_о + З_д + Н_{зп} + А + К + В_е + І_в$$

В = 11190,15+1119,01+4468,22+375+352+146,88+12309,16= 29960,42 (грн.)

Розрахуємо загальну вартість наукової роботи $B_{заг}$ за формулою:

$$В_{заг} = \frac{В_{ін}}{\alpha} \quad (4.7)$$

де α – частка витрат, які безпосередньо здійснює виконавець даного етапу роботи, у відн. одиницях = 1.

$$В_{заг} = \frac{29960,42}{1} = 29960,42$$

Прогнозування загальних витрат ЗВ на виконання та впровадження результатів виконаної наукової роботи здійснюється за формулою:

$$ЗВ = \frac{В_{заг}}{\beta} \quad (4.8)$$

де β – коефіцієнт, який характеризує етап (стадію) виконання даної роботи.

Отже, розрахуємо загальні витрати:

$$ЗВ = \frac{29960,42}{0,9} = 33289,35 \text{ (грн.)}$$

4.3 Прогнозування комерційних ефектів від реалізації результатів розробки.

Спрогнозуємо отримання прибутку від реалізації результатів нашої розробки. Зростання чистого прибутку можна оцінити у теперішній вартості грошей. Це забезпечить підприємству (організації) надходження додаткових коштів, які дозволять покращити фінансові результати діяльності .

Оцінка зростання чистого прибутку підприємства від впровадження результатів наукової розробки. У цьому випадку збільшення чистого прибутку підприємства $\Delta П_і$ для кожного із років, протягом яких очікується отримання позитивних результатів від впровадження розробки, розраховується за формулою:

$$\Delta П_i = \sum_1^n (\Delta П_я \cdot N + П_я \Delta N)_i \quad (4.9)$$

де $\Delta П_я$ – покращення основного якісного показника від впровадження результатів розробки у даному році;

N – основний кількісний показник, який визначає діяльність підприємства у даному році до впровадження результатів наукової розробки;

ΔN – покращення основного кількісного показника діяльності підприємства від впровадження результатів розробки;

$П_я$ – основний якісний показник, який визначає діяльність підприємства у даному році після впровадження результатів наукової розробки;

n – кількість років, протягом яких очікується отримання позитивних результатів від впровадження розробки.

В результаті впровадження результатів наукової розробки витрати на виготовлення інформаційної технології зменшаться на 20 грн (що автоматично спричинить збільшення чистого прибутку підприємства на 20 грн), а кількість користувачів, які будуть користуватись збільшиться: протягом першого року – на 250 користувачів, протягом другого року – на 200 користувачів, протягом

третього року – 150 користувачів. Реалізація інформаційної технології до впровадження результатів наукової розробки складала 600 користувачів, а прибуток, що отримував розробник до впровадження результатів наукової розробки – 300 грн.

Спрогнозуємо збільшення чистого прибутку від впровадження результатів наукової розробки у кожному році відносно базового.

Отже, збільшення чистого продукту $\Delta П_1$ протягом першого року складатиме:

$$\Delta П_1 = 20 \cdot 600 + (300 + 20) \cdot 250 = 92000 \text{ грн.}$$

Протягом другого року:

$$\Delta П_2 = 20 \cdot 600 + (300 + 20) \cdot (250 + 200) = 156000 \text{ грн.}$$

Протягом третього року:

$$\Delta П_3 = 20 \cdot 600 + (300 + 20) \cdot (250 + 200 + 150) = 204000 \text{ грн.}$$

4.4 Розрахунок ефективності вкладених інвестицій та період їх окупності

Визначимо абсолютну і відносну ефективність вкладених інвестором інвестицій та розрахуємо термін окупності.

Абсолютна ефективність $Е_{абс}$вкладених інвестицій розраховується за формулою:

$$Е_{абс} = (ПП - PV), \qquad (4.10)$$

де $\Delta П_i$ – збільшення чистого прибутку у кожному із років, протягом яких виявляються результати виконаної та впровадженої НДДКР, грн;

т – період часу, протягом якого виявляються результати впровадженої НДДКР, 3 роки;

τ – ставка дисконтування, за яку можна взяти щорічний прогнозований рівень інфляції в країні; для України цей показник знаходиться на рівні 0,1;

t – період часу (в роках) від моменту отримання чистого прибутку до точки 2, 3,4.

Рисунок, що характеризує рух платежів (інвестицій та додаткових прибутків) буде мати вигляд, рисунок 4.1.

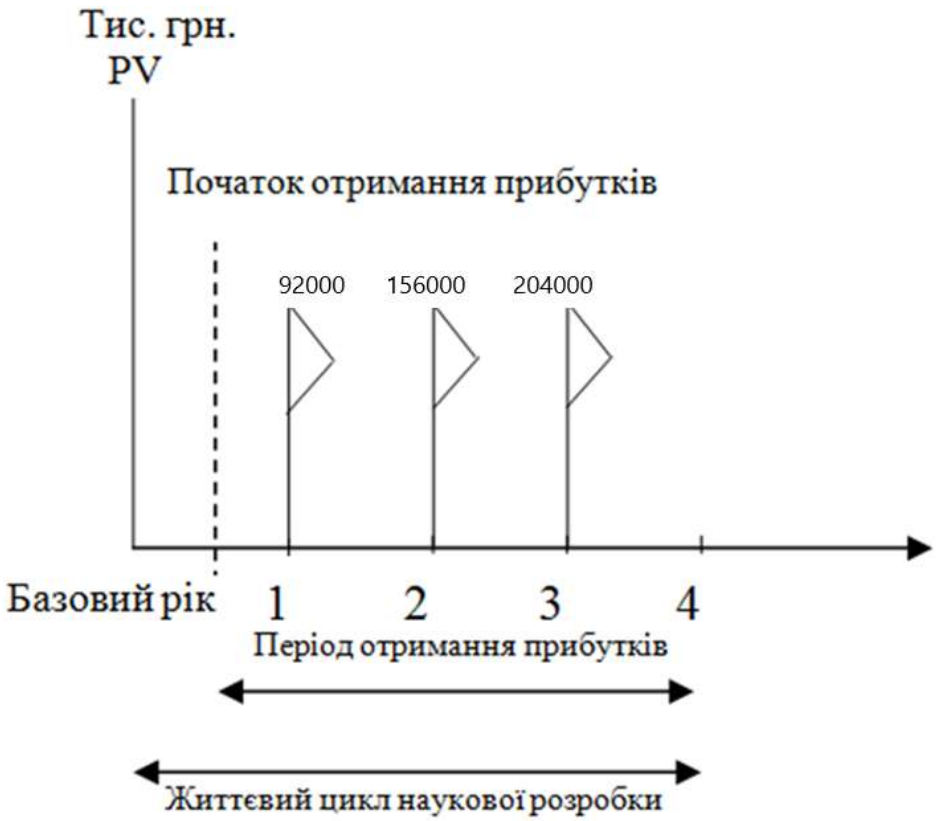

Рисунок 4.1 – Вісь часу з фіксацією платежів, що мають місце під час розробки та впровадження результатів НДДКР

Розрахуємо вартість чистих прибутків за формулою:

$$ПП = \sum_1^m \frac{\Delta П_i}{(1+\tau)^t} \quad (4.11)$$

де $\Delta П_i$ – збільшення чистого прибутку у кожному із років, протягом яких виявляються результати виконаної та впровадженої НДДКР, грн;

т – період часу, протягом якого виявляються результати впровадженої НДДКР, роки;

τ – ставка дисконтування, за яку можна взяти щорічний прогнозований рівень інфляції в країні; для України цей показник знаходиться на рівні 0,1;

t – період часу (в роках) від моменту отримання чистого прибутку до точки.

Отже, розрахуємо вартість чистого прибутку:

$$\text{ПП} = \frac{33289,35}{(1+0,1)^0} + \frac{92000}{(1+0,1)^2} + \frac{156000}{(1+0,1)^3} + \frac{204000}{(1+0,1)^4} = 365863,9 \text{ (грн.)}$$

Тоді розрахуємо $E_{\text{абс}}$:

$$\text{Е}_{\text{абс}} = 365863,9 - 33289,35 = 332574,55 \text{ грн.}$$

Оскільки $E_{\text{абс}} > 0$, то вкладання коштів на виконання та впровадження результатів НДДКР буде доцільним.

Розрахуємо відносну (щорічну) ефективність вкладених в наукову розробку інвестицій $\text{Е}_\text{в}$ за формулою:

$$\text{Е}_\text{в} = \sqrt[\text{Т}]{1 + \frac{\text{Е}_{\text{абс}}}{\text{PV}}} - 1 \qquad (4.12)$$

де $\text{Е}_{\text{абс}}$ – абсолютна ефективність вкладених інвестицій, грн;

PV – теперішня вартість інвестицій PV = ЗВ, грн;

$\text{Т}_\text{ж}$ – життєвий цикл наукової розробки, роки.

Тоді будемо мати:

$$\text{Е}_\text{в} = \sqrt[3]{1 + \frac{332574,55}{33289,35}} - 1 = 1,22 \text{ або } 122\ \%$$

Далі, розраховану величина $\text{Е}_\text{в}$ порівнюємо з мінімальною (бар'єрною) ставкою дисконтування $\tau_{\text{мін}}$, яка визначає ту мінімальну дохідність, нижче за яку інвестиції вкладатися не будуть. У загальному вигляді мінімальна (бар'єрна) ставка дисконтування $\tau_{\text{мін}}$ визначається за формулою:

$$\tau = d + f,$$

де d – середньозважена ставка за депозитними операціями в комерційних банках; в 2019 році в Україні d = 0,2;

f – показник, що характеризує ризикованість вкладень, величина f = 0,1.

$$\tau = 0{,}2 + 0{,}1 = 0{,}3$$

Оскільки $E_в$ = 122% > $\tau_{мін}$ = 0,3 = 30%, то у інвестор буде зацікавлений вкладати гроші в дану наукову розробку.

Термін окупності вкладених у реалізацію наукового проекту інвестицій. Термін окупності вкладених у реалізацію наукового проекту інвестицій $T_{ок}$ розраховується за формулою:

$$T_{ок} = \frac{1}{E_в}$$

$$T_{ок} = \frac{1}{1{,}22} = 0{,}81 \text{ року}$$

Обрахувавши термін окупності даної наукової розробки, можна зробити висновок, що фінансування даної наукової розробки буде доцільним.

4.5 Висновок

В даному розділі було проведено економічне обґрунтування доцільності розробки програми для класифікації тональності речень з використанням нейронної мережі. Незалежними експертами було здійснено оцінювання комерційного потенціалу розробки, за результатами якого було визначено, що нова розробка має високий рівень комерційного потенціалу, оскільки середньоарифметична сума балів становить 43. Також було виконано прогнозування витрат на виконання розробки, де розраховано основну заробітну плату кожного із розробників, додаткову заробітну плату всіх розробників, нарахування на заробітну плату, амортизацію обладнання, комп’ютерів та приміщень, витрати на допоміжні матеріали, витрати на силову електроенергію тощо. Загальна сума витрат на виконання означених робіт

склала 33289,35 грн. Виконано розрахунок ефективності вкладених інвестицій та періоду їх окупності. Визначено, що абсолютна ефективність вкладених інвестицій становить 365863,9 грн, і це свідчить про те, що вкладання коштів на виконання та впровадження результатів НДДКР є доцільним. Було розраховано відносну (щорічну) ефективність вкладених в наукову розробку інвестицій – 122 %, її величина більша за мінімальну (бар'єрну) ставку дисконтування, отже інвестор буде зацікавлений у фінансуванні даної наукової розробки. Проведено розрахунок терміну окупності - 0,81 року (10 місяців). Це означає, що вже починаючи з 11 місяця розробка буде приносити прибуток. В загальному можна зробити висновок, що фінансування розробки програми для класифікації тональності речень з використанням нейронної мережі є економічно доцільним проектом.

ВИСНОВКИ

В результаті виконання магістерської кваліфікаційної роботи було розв’язано задачу розробки інтелектуальної інформаційної технології та програмного забезпечення класифікації тональності речень згортковою нейронною мережею.

У першому розділі магістерської роботи був проведений аналіз суті технічної проблеми класифікації тональності речень, поставлене завдання та проаналізовані існуючі способи вирішення проблеми. Для задачі класифікації тональності речень було обрано методт на основі штучних нейронних мереж, адже вони мають здатність до навчанн та стійкість до шумів вхідних даних. Перспективним є використання саме згорткової нейронної мережі, адже за допомогою шарів згортки можна зменшити вхідні даніі, а відтак і зменшити складність обчислень. Також було проаналізовано існуючі програми-аналоги та зроблено постановку задачі. На основі вихідних даних, аналізу сучасних методів класифікації тональності речень та існуючих програм-аналогів було зроблено висновок, що метою роботи є підвищення достовірності.

У другому розділі магістерської кваліфікаційної роботи було обгрунтовано вибір згорткової нейронної мережі для побудови інформаційної технології класифікації тональності речень, визначено архітектуру згорткової нейронної мережі, математичну модель штучного нейрона та загальну математичну модель згорткової мережі. Для класифікації тональності речень використано три згорткових шари з фільтрами. Кількість фільтрів для кожного згорткового шару – 16. Функція, яка буде використана в шарах агрегування – максимізаційна (max-pooling). Для згорткових та агрегувальних шарів буде використана функція активації – ReLU. Метод, за яким буде проходити навчання мережі – це метод зворотного поширення помилки. У підсумку було розроблено структуру інформаційної технології класифікації тональності речень на основі згорткової нейронної мережі та розглянуто планування процесу розробки прикладного програмного забезпечення.

У третьому розділі в ході практичної реалізації інформаційної технології класифікації тональності речень було обрано такі мови програмування: для програми класифікації тональності речень – Python, для програмної реалізації клієнтського додатку для тестування роботи WebAPI та : web2py та sclit-learn. Також було описано роботу основних частин програми та наведено відповідні фрагменти коду. В результаті було розроблено програмне забезпечення класифікації тональності речень на основі згорткової нейронної мережі, яке порівняно з аналогом має кращу на 4,5% достовірність класифікації тональності речень. Таким чином, мета роботи досягнута – достовірність класифікації тональності речень підвищена. Розроблена програма повністю відповідає завданню, що підтверджується її тестуванням.

У четвертому розділі було проведено економічне обґрунтування доцільності розробки програми для класифікації тональності речень з використанням нейронної мережі. Було визначено, що нова розробка має високий рівень комерційного потенціалу, оскільки середньоарифметична сума балів становить 43. Також було виконано прогнозування витрат на виконання розробки. Загальна сума витрат на виконання означених робіт склала 33289,35 грн. Визначено, що абсолютна ефективність вкладених інвестицій становить 365863,9 грн, і це свідчить про те, що вкладання коштів на виконання та впровадження результатів НДДКР є доцільним. Відносна (щорічна) ефективність вкладених в наукову розробку інвестицій – 122 %. Проведено розрахунок терміну окупності - 0,81 року (10 місяців). Це означає, що вже починаючи з 11 місяця розробка буде приносити прибуток. В загальному можна зробити висновок, що фінансування розробки програми для класифікації тональності речень з використанням нейронної мережі є економічно доцільним проектом.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Барановський В.С. Програмний модуль класифікації тональності речень. НТКП ВНТУ. Факультет інформаційних технологій та комп'ютерної інженерії. / Барановський В.С. [Електронний ресурс] – режим доступу : https://conferences.vntu.edu.ua/index.php/all-fitki/all-fitki-2018/paper/view/5157 (дата звернення 01.12.2019) – Назва з екрана.

2. Барановський В. С., Колесницький О. К., Денисов І. К. Інформаційна технологія класифікації банківських текстів на основі згорткової нейронної мережі // Матеріали конференції «Молодь в науці: дослідження, проблеми, перспективи-2020», Вінниця, 2019. [Електронний ресурс]. Режим доступу: https://conferences.vntu.edu.ua/index.php/mn/mn2020/paper/view/8497. Дата звернення: Грудень 2019.

3. M. Toman. Influence of Word Normalization on Text Classification / M. Toman, R. Tesar, K. Jezek. – Plzen, Czech Republic, 2006. – 5 p.

4. Sentence Classification CNN – [Електронний ресурс]. – Режим доступу: https://github.com/umairqadir97/Sentense-Classification-with-CNN

5. Multiclass CNN Classifier – [Електронний ресурс]. – Режим доступу: https://github.com/jiegzhan/multi-class-text-classification-cnn

6. Штучні нейронні мережі – [Електронний ресурс]. – Режим доступу: http://archive.ws-conference.com/wp-content/uploads/pw0060

7. Руденко О.В. Штучні нейронні мережі: Навчальний посібник / О.В.Руденко, Є.В.Бодянський. - Харків : ТОВ «Компанія СМІТ», 2006. — 404 с. - ISBN 966-8630-73-X.

8. Згорткові нейронні мережі – [Електронний ресурс]. – Режим доступу: https://uk.wikipedia.org/wiki/Згорткова_нейронна_мережа

9. Штучний нейрон – [Електронний ресурс]. – Режим доступу: https://uk.wikipedia.org/wiki/Штучний_нейрон

10. GloVe: Global Vectors for Word Representation – [Електронний ресурс]. – Режим доступу: https://nlp.stanford.edu/pubs/glove.pdf

11. Метод векторизації слів - word2vec – [Електронний ресурс]. – Режим доступу: http://nlpx.net/archives/179

12. Декомпозиція інформаційних систем – [Електронний ресурс]. – Режим доступу: http://helpiks.org/1-27670.html

13. Python – [Електронний ресурс]. – Режим доступу: https://uk.wikipedia.org/wiki/Python

14. Softmax функція – [Електронний ресурс]. – Режим доступу: https://uk.wikipedia.org/wiki/Softmax_функція

15. Flask – [Електронний ресурс]. – Режим доступу: https://ru.wikipedia.org/wiki/Flask_(веб-фреймворк)

16. UML-діаграми у процесі проектування ПО – [Електронний ресурс]. – Режим доступу: http://baumanki.net/lectures/10-informatika-i-programmirovanie/273-uml/3457-2-vidy-diagramm-uml.html

17. Movie Review Data – [Електронний ресурс]. – Режим доступу: http://www.cs.cornell.edu/people/pabo/movie-review-data/

18. Методичні вказівки до виконання студентами-магістрантами економічної частини магістерських кваліфікаційних робіт / Уклад. В. О. Козловський – Вінниця: ВНТУ, 2012. – 22 с.