УДК 519.854.2

Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет *Комп'ютерних наук*
(повна назва)

Кафедра *Системотехніки*
(повна назва)

АТЕСТАЦІЙНА РОБОТА
Пояснювальна записка

Рівень вищої освіти другий (магістерський)

ГЮИК 501610.010 ПЗ

Розробка метода аналізу контенту сайту на основі нейронних мереж
(тема)

Виконав:
Студент 2 курсу, групи *СПРм-19-1*
Спеціальність *122 – Комп'ютерні науки*
(код і повна назва напряму)
Тип програми *освітньо-професійна*
(освітньо-професійна або освітньо-наукова)
Освітня програма *Системне проектування*
(повна назва освітньої програми)

Сіряченко М.О.
(прізвище, ініціали)

Керівник доц. Ситніков Д.Е.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ *Гребеннік І. В.*
(підпис) (прізвище, ініціали)

2020 р.

Харківський національний університет радіоелектроніки

Факультет *Комп'ютерних наук*

(повна назва)

Кафедра *Системотехніки*

(повна назва)

Рівень вищої освіти другий (магістерський)

Спеціальність *122 – Комп'ютерні науки*

(код і повна назва)

Тип програми *освітньо-професійна*

(освітньо-професійна або освітньо-наукова)

Освітня програма *Системне проектування*

(повна назва)

ЗАТВЕРДЖУЮ:

Зав. кафедри _____

(підпис)

«____»_____ 20___ р.

ЗАВДАННЯ

НА АТЕСТАЦІЙНУ РОБОТУ

Студентові ***Сіряченко Максиму Олександровичу***

(прізвище, ім'я, по батькові)

1. Тема роботи *Розробка метода аналізу контенту сайту на основі нейронних мереж*
затверджена наказом по університету від " *02* **"** *листопада* **2020р. №** *1516Ст*

2. Термін подання студентом роботи *22.12.2020 р.*

3. Вихідні дані до роботи *Функції системи: Розробка методу аналізу контенту сайту на основі нейронних мереж. Організація даних: база даних. Форма діалогу: в інтерактивно-графічному режимі. Перелік використовуваних програмних засобів: ОС Microsoft Windows 10, інтегроване середовище програмування Google Colab, браузер Google Chrome версії 75, PostegreeSQL. Технічне забезпечення: комп'ютер для виконання тестів з підтримкою Python та зі встановленим браузером Google Chrome версії 41 та вище й не менш, ніж 2Гб оперативної пам'яті.*

4. Зміст пояснювальної записки (перелік питань, що потрібно розробити)
4.1 *Вступ.* ***4.2*** *Аналіз предметної області.* ***4.3*** *Методи інтелектуального аналізу.* ***4.4*** *Розробка моделі.* ***4.5*** *Експерименти.* ***4.6*** *Висновки*

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, плакатів)
5.1.Типова структура нейрона. **5.2** Модель Перцептрона. **5.3** Залишковий блок. **5.4** Приклад одновимірного згорткового шару. **5.5** Двовимірний згортковий шар. **5.6** Архітектура згорткової мережі для класифікації речення. **5.7** Приклади вхідних, вихідних та братських сторінок. **5.8** Огляд моделі. **5.9** Витяг веб-сторінки. **5.10** Передобробка веб-ресурсів. **5.11** Процес класифікації веб-сторінки із використанням методу WPCM. **5.12** Схема структури бази даних. **5.13** Накопичена частка основних компонентів, що генерують PCA. **5.14** Приклад коду з підключенням бази даних. **5.15** Вектори ознак з PCA та CPBF передаються в нейронні мережі для класифікації. **5.16** Результати першого запуску для mse та epochs для класифікації веб-сторінок з використанням нейронних мереж (швидкість імпульсу = 0.01) **5.17** Результати першого запуску для mse та epochs для класифікації веб-сторінок з використанням нейронних мереж (швидкість імпульсу = 0.001

6. Консультанти з роботи із зазначенням розділів роботи, що їх стосуються

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		(підпис)	(дата)
Розділи спеціальної частини	*доц. Ситніков Д. Е.*		

7. Дата видачі завдання *02.11.2020 р.*

КАЛЕНДАРНИЙ ПЛАН

Пор. №	Назва етапів атестаційної роботи	Термін виконання етапів роботи	Примітка
1.	*Отримання завдання атестаційної роботи*	*02.11.20*	
2.	*Аналіз завдання, літератури та аналогів з теми атестаційної роботи*	*03.11.20*	
3.	*Вибір засобів для розробки технічних вимог до програми*	*05.11.20*	
4.	*Структурне проектування*	*05.11.20*	
5.	*Вибір середовища розробки програми*	*10.11.20*	
6.	*Розробка метода аналізу контенту*	*13.11.20*	
7.	*Розробка програми*	*17.11.20*	
8.	*Тестування програми*	*25.11.20*	
9.	*Оформлення пояснювальної записки та програмної документації*	*11.12.20*	
10.	*Оформлення графічної частини та презентаційних матеріалів комп'ютерного захисту*	*16.12.20*	
11.	*Представлення на рецензування*	*18.12.20*	
	Представлення атестаційної роботи в ДЕК	*18.12.20*	

Студент *Сіряченко М. О.*
(підпис)

Керівник роботи доцент Ситніков Д. Е.
(підпис)

РЕФЕРАТ

Пояснювальна записка до атестаційної роботи містить: 79 сторінок, 17 рисунків, 3 таблиці, 14 джерел. Графічна частина атестаційної роботи містить 17 плакатів.

Метою атестаційної роботи - розробка методу аналізу контенту сайту на основі нейронних мереж.

Об’єктом дослідження – інформаційна система аналізу контенту сайту.

Предмет дослідження – методи аналізу контенту на основі нейронних мереж.

Результатом атестаційної роботи є спроектований та розроблений метод аналізу контенту сайтів за певною тематикою.

PRINCIPAL COMPONENT ANALYSIS, PYTHON, АНАЛІЗ КОНТЕНТУ, ВЕБ-САЙТ, ІНТЕЛЕКТУАЛЬНИЙ АНАЛІЗ, НЕЙРОННІ МЕРЕЖІ, ПАРСИНГ, СТЕМЕР

ABSTRACT

The explanatory note to the attestation work contains: 79 pages, 17 figures, 3 tables, 14 sources. The graphic part of the certification work contains 17 posters.

The purpose of the certification work is to develop a method of site content analysis based on neural networks.

The object of research is the information system of site content analysis. The subject of research - methods of content analysis based on neural networks.

The result of the certification work is a designed and developed method of analyzing the content of sites on certain topics.

CONTENT ANALYSIS, INTELLECTUAL ANALYSIS, NEURAL NETWORKS, PARSING, PRINCIPAL COMPONENT ANALYSIS, PYTHON, STEMER, WEBSITE

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

БД – база даних;

СКБД – Система керування базами даних;

SQL – structured query language;

URL – Uniform Resource Locator (уніфікований покажчик інформаційного ресурсу);

PCA - principal component analysis;

CPBF - class profile-based features;

WPCB - web page classification based on principal component analysis.

ВСТУП

Сучасний світ не можна уявити без читання і обробки інформації. Обсяг інформації, яку отримує людина, росте у величезній кількості. І ця інформація може бути оброблена різними інформаційними системами. На даний час найпростіший для інженера-програміста спосіб, це нейронна мережа. Нейронними мережами обробляється будь-яка інформація, від графічної до величезних масивів даних.

Машинне навчання - великий підрозділ штучного інтелекту, що вивчає методи побудови алгоритмів, здатних навчатися[10]. Навчання по прецедентах, або індуктивне навчання, засноване на виявленні загальних закономірностей по приватним емпіричним даним. Багато методів індуктивного навчання розроблялися як альтернатива класичним статистичним підходам і тісно пов'язані з витяганням інформації та інтелектуальним аналізом даних.

Оскільки Інтернет продовжує розширюватись, пошук актуальної інформації за допомогою традиційних пошукових систем стає все важчим, бо кількість сторінок, що підлягають індексуванню, перевищує вісім мільярдів, пошуковим системам стає все важче вести сучасний та всебічний пошук. Багато користувачів починають свою веб-діяльність, надсилаючи запит пошуковій системі. Користувачам часто важко шукати корисну та якісну інформацію в Інтернеті за допомогою пошукових систем загального призначення, особливо під час пошуку конкретної інформації з певної теми. Багато вертикальних пошукових систем або специфічних пошукових систем створено для полегшення більш ефективного пошуку в різних доменах. Ці пошукові системи певною мірою полегшують проблему перевантаження інформацією, забезпечуючи більш точні результати та більш персоналізовані функції. Однак розробникам тематичних пошукових систем потрібно вирішити два питання: як знайти відповідні сайти (URL-адреси) в Інтернеті та як відфільтрувати недоречні сайти з набору сайтів, зібраних з Інтернету. Завдяки швидкому розвитку машинного навчання, його можна застосувати в методі, який поєднує аналіз веб-вмісту та аналіз веб-структури[11].

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Опис предметної області

В останні роки структура веб-посилань широко використовується для виведення важливої інформації про сторінки. Інтуїтивно автор веб-сторінки A поміщає посилання на веб-сторінку B, якщо він або вона вважає, що B має відношення до A або має хорошу якість. Звичайно чим більше кількість внутрішніх посилань, тим краще вважається сторінка. Обгрунтування цього полягає в тому, що сторінка, на яку посилається більше людей, ймовірно, буде більш важливою, ніж сторінка, на яку рідко посилаються. Також можна отримати текст прив'язки, що описує посилання. Якірний текст - це інтерактивний текст вихідного посилання на веб-сторінці. Якірний текст може дати гарний опис цільової сторінки, тому що він представляє, як насправді її описують інші люди, які посилалися на сторінку.

Крім того, розумно надати посиланням з авторитетного джерела (наприклад, Yahoo) більшу вагу, ніж посилання з неважливої особистої домашньої сторінки.

Було проведено багато досліджень різних способів представлення та аналізу змісту і структури Інтернету. В цілому їх можна розділити на дві категорії: на основі контенту і на основі посилань. Фактичне HTML-вміст веб-сторінки надає багато корисної інформації про саму сторінці. Наприклад, основний текст веб-сторінки можна проаналізувати, щоб визначити, чи відповідає сторінка цільовим домену.

Методи індексування можуть використовуватися для отримання ключових понять, що представляють сторінку. Інформація, витягнута з документа з використанням різних методів, може бути корисна для класифікації тексту [1]. Крім того, релевантність сторінки часто можна визначити по заголовку. Слова і фрази, які з'являються в заголовку або заголовках в структурі HTML, зазвичай отримують більш високий вагу. Такі ваги можна розрахувати на основі оцінок TFIDF.

1.1.1 TFIDF - показник для оцінки важливості слів у контексті документа

TFIDF означає термін «частота-зворотна частота документа», а вага TFIDF - це вага, котра часто використовується при пошуку інформації та інтелектуальному аналізі тексту. Ця вага - статистична міра, яка використовується для оцінки того, наскільки важливо слово для документа в колекції. Важливість зростає пропорційно тому, скільки разів слово з'являється в документі, але компенсується частотою появи слова в колекції. Варіанти схеми зважування TFIDF часто використовуються пошуковими системами як центрального інструменту для оцінки і ранжирування релевантності документа за запитом користувача. Одна з найпростіших функцій ранжирування обчислюється шляхом підсумовування TFIDF для кожного терміна запиту; багато складніші функції ранжирування є варіантами цієї простої моделі.

TFIDF може успішно використовуватися для фільтрації стоп-слів в різних предметних полях, включаючи текстове узагальнення і класифікацію. Знання предметної області також можуть бути включені в аналіз для поліпшення результатів. Знання предметної області відноситься до експертних знань, таких як лексика або правила предметної області, часто одержувані від експертів-людей. Наприклад, слова на веб-сторінках можна порівняти зі списком термінів, що відносяться до предметної області. Веб-сторінка, що містить слова зі списку, може вважатися більш релевантною.

1.1.2 Веб-ресурс, як об’єкт

До веб-ресурсів можна віднести різноманітні розважальні сайти, інтернет магазини, віртуальні бібліотеки, тощо. Веб-ресурс можна представляти у вигляді одного окремого об’єкта. Під характеристиками конкретного об'єкта розуміється деревоподібна система значущих понять (концептів), що описують його певні значимі властивості. Для цього включені окремі типові характеристики для визначення приналежності до певної групи і узагальнені характеристики для всього об'єкта. Для порівняння значень аналізованих характеристик і відповідних їм довідкових даних необхідно проаналізувати інформацію, представлену на відповідних веб-сторінках .

Структура опису інформації про аналізовані об'єкти апріорі невідома і може варіюватися в залежності від тематики, змісту і технічної реалізації цих сайтів.

Доцільно, щоб інформація на сайтах була структурована, а не просто розбита на блоки або параграфи. Ця вимога може значно спростити процес аналізу релевантності контенту. В даному випадку структурування означає представлення інформації у вигляді певних структур.

1.1.3 Поняття нейронної мережі

1.1.3.1 Складові штучної нейронної мережі

Штучні нейронні мережі були побудовані за принципом біологічних нейронних мереж, які представляють собою мережі нервових клітин, які виконують певні фізіологічні функції. Складовим елементом нейронних мереж є нейрони (представлені на рис.1).

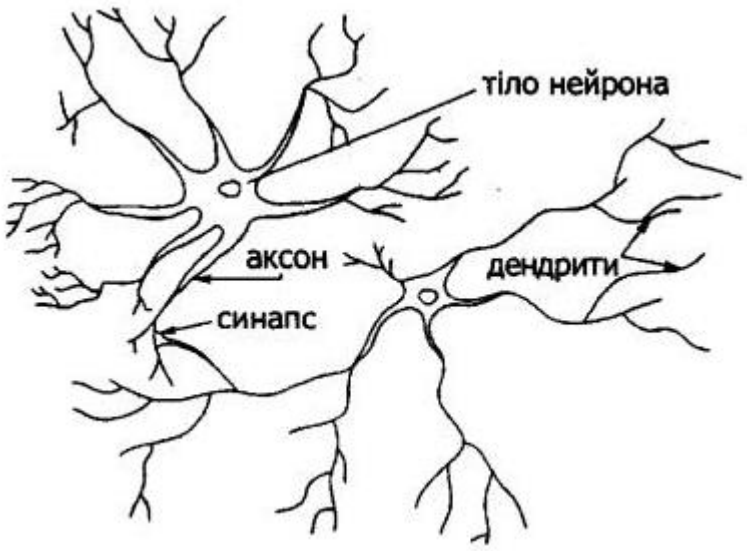

Рис. 1.1 – Типова структура нейрона

У нейрона є кілька функцій:

– Приймальна функція: синапси отримують інформацію;

– Інтегративна функція: на виході нейрона сигнал, який несе інформацію про всі підсумовуваних в нейроні сигналах;

– Провідникова функція: по аксону проходить інформація синапси;

– Передає функція: імпульс, який досяг закінчення аксона, змушує медіатор передавати збудження наступного нейрона.

Синапсами називають зв'язки, за якими вихідні сигнали одних нейронів надходять на входи інших. Кожний зв'язок характеризується своєю вагою. Зв'язки з позитивною вагою називаються збудливими, а з негативною - гальмують. Вихід нейрона називається аксоном. В штучної нейронної мережі штучний нейрон - це деяка нелінійна функція, аргументом якої є лінійна комбінація всіх вхідних сигналів. Така функція називається активаційною. Потім результат активаційної функції посилається на вихід нейрона. Об'єднуючи такі нейрони з іншими, отримують штучну нейронну мережу.

1.1.3.2 Функція активації

Функції активації нейронної мережі - важливий компонент глибокого навчання. Функції активації визначають результат роботи моделі глибокого навчання, її точність, а також обчислювальну ефективність навчання моделі, яка може створити або зламати великомасштабну нейронну мережу. Функції активації також мають великий вплив на здатність нейронної мережі до конвергенції і швидкість конвергенції, а в деяких випадках функції активації можуть взагалі перешкоджати конвергенції нейронних мереж.

Функції активації - це математичні рівняння, які визначають вихідний сигнал нейронної мережі. Функція прив'язана до кожного нейрона в мережі і визначає, чи слід його активувати («активувати») чи ні, в залежності від того, чи релевантний вхід кожного нейрона для передбачення моделі. Функції активації також допомагають нормалізувати вихідний сигнал кожного нейрона до діапазону від 1 до 0 або від -1 до 1.

Додатковим аспектом функцій активації є те, що вони повинні бути ефективними з обчислювальної точки зору, оскільки вони розраховуються для тисяч або навіть мільйонів нейронів для кожної вибірки даних. Сучасні нейронні мережі для

навчання моделі використовують метод, званий зворотним поширенням, що збільшує обчислювальну навантаження на функцію активації та її похідну функцію.

1.1.3.3 Перцептрон

Персептрони, які вважаються першим поколінням нейронних мереж, представляють собою просто обчислювальні моделі окремого нейрона. Їх популяризував Френк Розенблат на початку 1960-х років. У них виявився дуже потужний алгоритм навчання, і було зроблено багато гучних заяв про те, що вони можуть навчитися робити. У 1969 році Minsky and Papers опублікували книгу під назвою «Персептрони», в якій проаналізували їх можливості і показали їх обмеження. Багато хто думав, що ці обмеження можуть застосовуватися до всіх моделей нейронних мереж. Однак процедура навчання персептрона до сих пір широко використовується для задач з величезними векторами ознак, які містять не один мільйон ознак.

У стандартній парадигмі статистичного розпізнавання образів ми спочатку перетворимо вихідний вхідний вектор в вектор активацій функцій. Потім ми використовуємо рукописні програми, засновані на здоровому глузді, для визначення функцій. Потім ми дізнаємося, як зважити кожну активацію функції, щоб отримати одну скалярну величину. Якщо ця кількість перевищує певний поріг, ми вирішуємо, що вхідний вектор є позитивним прикладом цільового класу.

Стандартна архітектура Перцептрона - наслідок моделі з прямим зв'язком, тобто вхідні дані відправляються в нейрон, обробляються і призводять до вихідних даних. На рисунку 2 це означає, що мережа читає від низу до верху: введення йде знизу, а висновок виходить зверху.

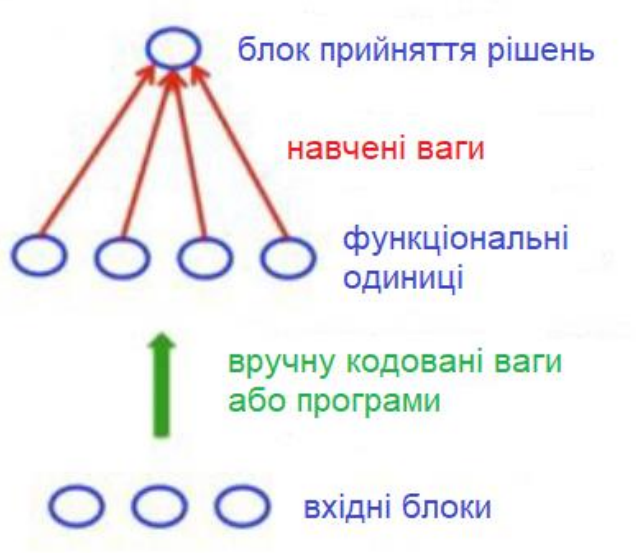

Рис. 1.2 – Модель Перцептрона

Однак у перцептронів є обмеження: якщо за вами стежать, щоб вибирати функції вручну і якщо ви використовуєте достатньо функцій, ви можете робити майже все. Для двійкових вхідних векторів у нас може бути окремий блок ознак для кожного з експоненціально багатьох довічних векторів, і тому ми можемо робити будь-яку можливу дискримінацію на довічних вхідних векторах. Але як тільки закодовані вручну функції визначені, існують дуже сильні обмеження на те, що перцептрон може вивчити.

Цей результат є руйнівним для перцептронів, тому що весь сенс розпізнавання образів полягає в розпізнаванні образів, незважаючи на перетворення, такі як переклад. «Теорема груповий інваріантності» Мінскі і паперті говорить, що навчається частина персептрона не може цьому навчитися, якщо перетворення утворюють групу. Щоб мати справу з такими перетвореннями, персептрону необхідно використовувати кілька функціональних одиниць для розпізнавання перетворень інформативних подшаблонов. Таким чином, складна частина розпізнавання образів повинна вирішуватися за допомогою кодованих вручну детекторів ознак, а не за допомогою процедури навчання.

Мережі без прихованих модулів дуже обмежені в відображеннях введення-виведення, які вони можуть навчитися моделювати. Більше шарів лінійних одиниць

не допомагає. Він як і раніше лінійний. Фіксованою вихідний нелінійності недостатньо. Таким чином, нам потрібно кілька рівнів адаптивних, нелінійних прихованих одиниць. Але як ми навчаємо такі мережі? Нам потрібен ефективний спосіб адаптації всіх ваг, а не тільки останнього шару. Це важко. Вивчення ваг в прихованих одиницях еквівалентно вивченню функцій. Це складно, тому що ніхто не говорить нам безпосередньо, що повинні робити приховані блоки.

1.1.3.4 Nearest Neighbour rule

Серед різних методів контрольованого статистичного розпізнавання образів правило найближчого сусіда забезпечує стабільно високу продуктивність без апріорних припущень про розподіли, котрих взято навчальні приклади. Він включає навчальний набір як позитивних, так і негативних випадків. Нова вибірка класифікується шляхом обчислення відстані до найближчого навчального набору; знак цієї точки потім визначає класифікацію зразка. Класифікатор k-NN розширює цю ідею, беручи k найближчих точок і привласнюючи знак більшості. Зазвичай для розриву зв'язків вибирають k маленькими і непарними (зазвичай 1, 3 або 5). Великі значення k допомагають зменшити вплив зашумленних точок в наборі навчальних даних, а вибір k часто виконується за допомогою перехресної перевірки.

Існує безліч методів підвищення продуктивності і швидкості класифікації найближчого сусіда. Один з підходів до цієї проблеми - попередньо впорядкувати навчальні набори якимось чином (наприклад, kd-деревами або ланками Вороного). Інше рішення - вибрати підмножину навчальних даних таким чином, щоб класифікація за правилом 1-NN (з використанням підмножини) наближалася до класифікатору Байеса. Це може привести до значного збільшення швидкості, оскільки тепер k можна обмежити до 1, а надлишкові точки даних були видалені з навчального набору. Ці методи модифікації даних можуть також поліпшити продуктивність за рахунок видалення точок, що викликають неправильну класифікацію.

Вищезазначене обговорення зосереджено на проблемах двійкової класифікації; є тільки два можливих вихідних класи. У прикладі розпізнавання цифр є десять класів виведення, що трохи змінює ситуацію. Маркування навчальних вибірок та

обчислення відстані не змінилися, але тепер зв'язку можуть виникати навіть з непарним k. Якщо все k найближчих сусідів належать до різних класів, ми не ближче до вирішення, ніж з правилом єдиного найближчого сусіда. Тому ми повернемося до правила 1-NN, коли у всіх немає більшості серед k найближчих сусідів. Правило найближчого сусіда досить просте, але вимагає великих обчислювальних ресурсів.

1.1.3.5 Затухаючий градієнт

Проблема зникаючого градієнта - це складність, яка виникає при навчанні штучних нейронних мереж із застосуванням методів навчання на основі градієнта і зворотного поширення помилки. В таких методах любий каприз нейронної мережі оновлюється пропорційно градієнту функції помилки щодо поточної ваги на кожній ітерації навчання. Стандартні функції активації, такі як гіперболічний тангенс, мають градієнти в діапазоні (-1, 1), а метод зворотного поширення помилки обчислює їх по ланцюговому правилу. Після множення цих чисел для обчислення градієнтів "фронтальних" шарів в n-шарової мережі, що означає, що градієнт (сигнал помилки) експоненціально зменшується разом з n, а передні шари навчаються дуже повільно. У міру того, як до нейронних мереж додається більше шарів, що використовують певні функції активації, градієнти функції втрат наближаються до нуля, що ускладнює навчання мережі.

Деякі функції активації, такі як сигмоїдальна функція, стискають великий вхідний простір в маленький вхідний простір між 0 і 1. Тому велика зміна входу сигмоїдной функції викличе невелику зміна вихідного. Отже, похідна стає маленькою.

Найпростіше рішення - використовувати інші функції активації, такі як ReLU, які не утворюють малу похідну. Залишкові мережі - інше рішення, оскільки вони забезпечують залишкові з'єднання прямо до більш ранніх рівнів. Як видно на рисунку 3, залишкове з'єднання безпосередньо додає значення на початку блоку, x, в кінець блоку (F (x) + x). Цей залишковий зв'язок не проходить через функції активації, які «розчавлюють» похідні, що призводить до більш високої загальної похідної блоку.

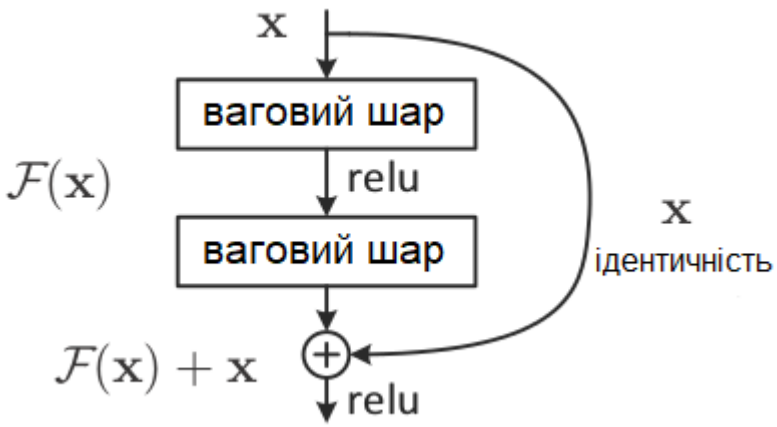

Рис. 1.3 – Залишковий блок

1.1.3.6 Сигмоїдальні активаційні функції

Застосування сигмоїдальних активаційних функцій може викликати проблеми в навчанні глибинних мереж, а саме значення активацій в кінцевому шарі будуть близькі до нуля на ранніх етапах навчання, сповільнюючи цей процес. Були запропоновані альтернативні активаційні функції, які не так страждають від обмеження.

1.1.3.7 Вибір відповідних ваг

Вибір відповідних ваг і momentum schedule в імпульсному стохастичною градієнтному спуску (momentum-based stochastic gradient descent) суттєво впливають на здатність навчати глибокі мережі.

1.1.3.8 Згорткові нейронні мережі

Дослідження в області машинного навчання протягом довгого часу приділяли велику увагу проблемам виявлення об'єктів. Є різні речі, які погано розпізнають об'єкти:

– Сегментація: реальні сцени захаращені іншими об'єктами. Важко сказати, які частини йдуть разом, як частини одного об'єкта. Частини об'єкта можуть бути приховані за іншими об'єктами.

– Освітлення: інтенсивність пікселів визначається як освітленням, так і об'єктами.
– Деформація: об'єкти можуть деформуватися різними неаффінними способами. Наприклад, рукописний текст теж може мати велику петлю або просто виступ.
– Можливості: класи об'єктів часто визначаються тим, як вони використовуються. Наприклад, стільці - це предмети, на яких можна сидіти, тому вони мають найрізноманітніші фізичні форми.
– Точка зору: зміна точки зору викликає зміни зображень, з якими не можуть впоратися стандартні методи навчання. Інформаційні скачки між вхідними розмірами (тобто пікселями).

Наприклад, візьмемо медичну базу, в якій вік пацієнта іноді залежить від вхідного вимірювання, яке зазвичай відповідає вазі. Щоб застосувати машинне навчання, ми спочатку хотіли б позбутися стрибкоподібної зміни параметрів. Підхід з реплікованими функціями в даний час є домінуючим підходом для нейронних мереж для вирішення проблеми виявлення об'єктів. Він використовує багато різних копій одного і того ж детектора ознак з різними позиціями. Він також може відтворюватися в будь-якому масштабі і орієнтації, що складно і дорого. Реплікація значно скорочує кількість вільних параметрів, які необхідно вивчити. Він використовує кілька різних типів функцій, кожен зі своєю власною карткою реплікованих детекторів. Це також дозволяє уявити кожен фрагмент зображення декількома способами.

За допомогою реплікації детекторів ознак, можна отримати наступне:

– Еквівалентні дії: реплікованих функції не роблять нейронну активність інваріантною для перекладу. Дії еквіваріантні.
– Незмінні знання: якщо функція корисна в деяких місцях під час навчання, детектори для цієї функції будуть доступні у всіх місцях під час тестування.

У 1998 році Янн ЛеКун і його співробітники розробили дійсно хороший розпізнавач рукописних цифр під назвою LeNet. Він використовував зворотне поширення в мережі з прямим зв'язком з багатьма прихованими шарами, безліччю карт реплікованих одиниць в кожному шарі, об'єднанням вихідних даних сусідніх

реплікованих одиниць, широкою мережею, яка може справлятися з декількома символами одночасно, навіть якщо вони перекриваються, і розумною спосіб навчання всієї системи, а не тільки розпізнавача. Пізніше це формалізовано під назвою згорткові нейронні мережі.

Згортка - це перший шар для вилучення функцій з вхідного зображення. Згортка зберігає взаємозв'язок між пікселями, вивчаючи особливості зображення за допомогою невеликих квадратів вхідних даних. Це математична операція, яка приймає два входи, такі як матриця зображення і фільтр або ядро - f і g і породжує третю послідовність.

$$(f * g)(c) = \sum_a f(a)\, g(c-a), \text{где } a = b + c$$

Формула для двовимірної згортки:

$$(f * g)(c_1, c_2) = \sum_a f(a_1, a_2)\, g(c_1 - a_1, c_2 - a_2)$$

Розглянемо одновимірний згортковий шар з входами xn і виходами yn (рис. 4). Функція для виходів буде представлений таким чином:

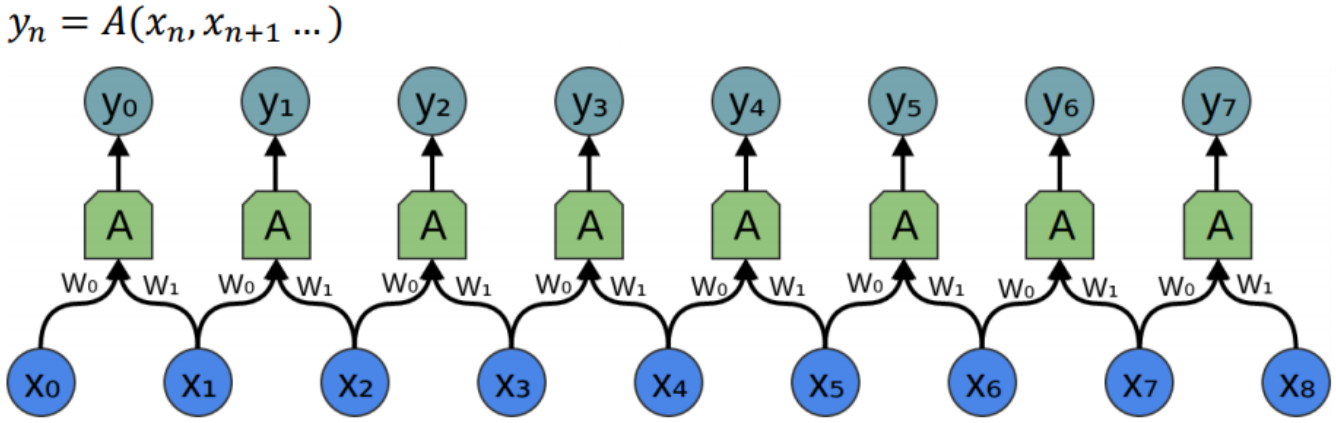

Рис. 1.4 – Приклад одновимірного згорткового шару

У згортковому шарі знаходиться безліч копій одного і того ж нейрона, тому багато цифр в декількох позиціях.

$$y0 = \sigma(W0x0 + W1x1 - b)$$

$$y1 = \sigma(W0x1 + W1x2 - b)$$

Стандартна матриця ваг з'єднує кожен вхід з кожним матрицею нейрону з різними вагами. Матриця для згорткового шару відрізняється тим, що різні ваги з'являються на декількох позиціях, оскільки нейрони пов'язані з усіма можливими входами, матриця містить безліч нульових елементів:

$$M = \begin{bmatrix} \omega_0 & \omega_1 & 0 & \dots \\ 0 & \omega_0 & \omega_1 & \dots \\ 0 & 0 & \omega_0 & \dots \end{bmatrix}$$

Тобто множення на матрицю вище - те ж саме, що і згортка у [... 0, w1, w0, 0 ...]. Ядро згортки, ковзаюче по різних частинах зображення, відповідає наявності нейронів в цих частинах. Згортку можна пояснити на прикладі обробки зображень. Якщо уявити, що зображення - двовимірні функції, то різні перетворення зображень не що інше, як згортка функції зображення з локальної функцією, яка називається ядром згортки. Кожен новий піксель зображення являє собою зважену суму пікселів, які ядро пройшло до цього моменту часу. Двовимірний згортковий шар представлений на рис. 5.

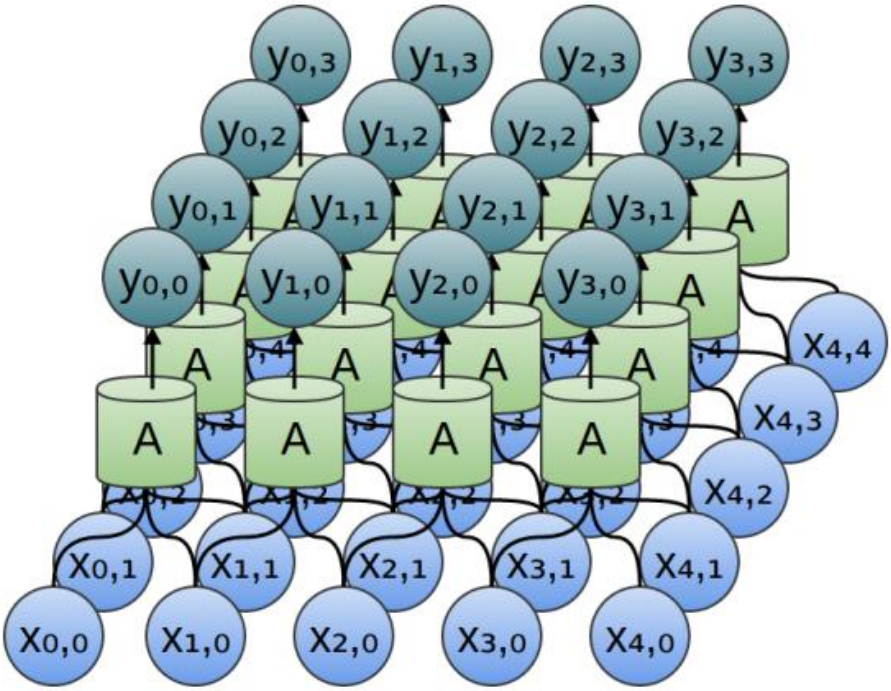

Рис. 1.5 – Двовимірний згортковий шар

Згорткові нейронні мережі можуть використовуватися для всіх робіт, пов'язаних з розпізнаванням об'єктів, від рукописних цифр до тривимірних об'єктів. Однак розпізнати реальні об'єкти на кольорових фотографіях, завантажених з Інтернету, набагато складніше, ніж розпізнати рукописні цифри. Класів в сто разів більше (1000 проти 10), в сто раз більше пікселів (256 х 256 кольорів проти 28 х 28 сірого), двовимірні зображення тривимірних сцен, захаращені сцени, що вимагають сегментації, і кілька об'єктів в кожному образі.

1.1.3.9 Застосування згорткових нейронних мереж в аналізі тексту

На вхід нейронної мережі буде подаватися матриця, кількість рядків якої залежить від розмірності словника, а ширина фільтрів дорівнює кількості стовпців цієї матриці (тобто використовуються розмірності для кодування кожного слова). Висота (або розмір фрагмента вхідних даних) може змінюватися, але зазвичай вона складає близько 2-5 слів. Перші шари представляють слова в вигляді низькорозмірних векторів. Наступний шар виконує згортки над векторними уявленнями слів,

використовуючи фільтри різних розмірів (тобто вони захоплюють 3-5 слів одночасно).

Потім проводиться пулінг (max-pool) над результатом згортки. До отриманого довгого вектору ознак додаємо регуляризацію (dropout в цьому випадку). Нарешті, відбувається класифікація результату за допомогою шару softmax (див. Рис. 6). У якості входів задаються не тільки стандартні X і Y, а і ймовірність того, що нейрон виявиться в шарі дропаутів (дропаут задається тільки під час тренування мережі). Перший шар - шар уявлення слів у вигляді векторів word2vec - є таблицею перетворення (відповідності). Результат застосування цього шару - тривимірний тензор розмірності [None, sequence_length, embedding_size].

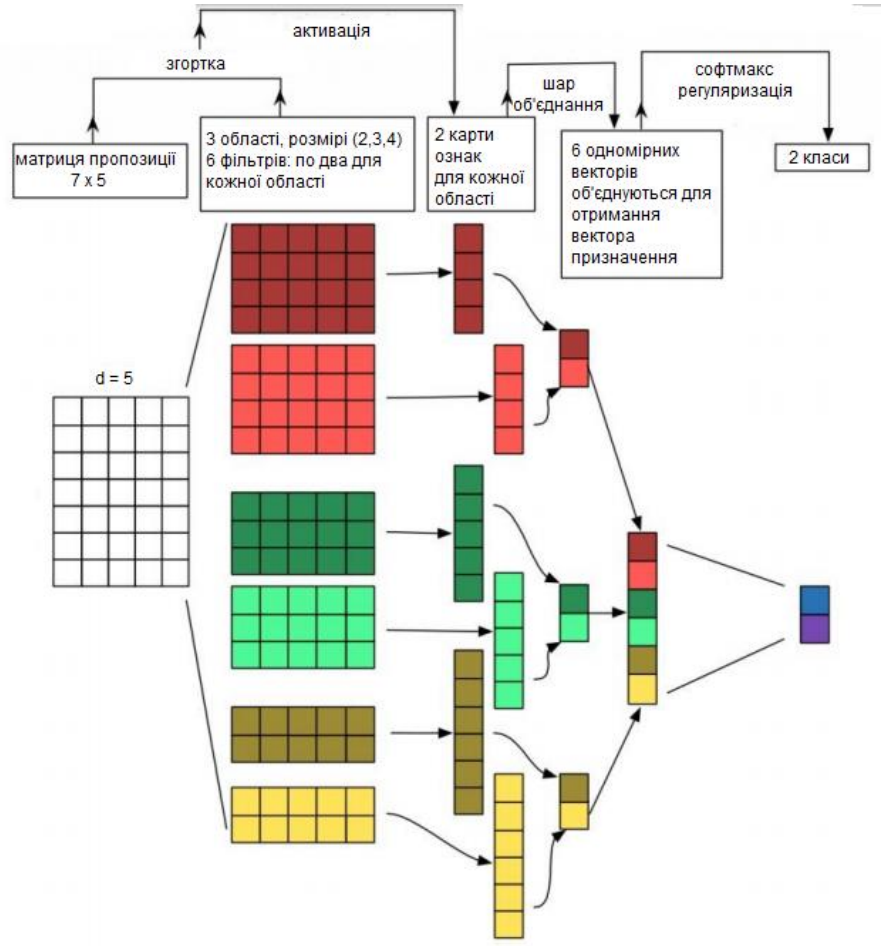

Рис. 1.6 – Архітектура згорткової мережі для класифікації речення

Наступний шар згортки приймає на вхід 4-мірний тензор (Батч, ширина, висота і канал). До отриманого на попередньому кроці тензора просто додається новий вимір [None, sequence_length, embedding_size, 1]. Кожен фільтр проходить повністю через все уявлення, відмінність тільки в тому, скільки він обробляє слів. У даній роботі використовується вузька згортка (narrow convolution) - тому на виході тензор має форму [1, sequence_length - filter_size + 1, 1, 1]. Після застосування пулінгу над виходом певного фільтра вийде тензор форми [batch_size, 1, 1, num_filters] По суті це і є вектор ознак, останній вимір якого відповідає потрібним нам ознаками.

Після отримання всіх тензорів, потрібно об'єднати їх в один довгий вектор ознак форми [batch_size, num_filters_total] Найпопулярнішим методом регуляризації згорткових нейронних мереж є dropout. Dropout блокує групу нейронів випадковим чином. Це змушує їх "вивчати" корисні ознаки самостійно. Група нейронів, яка бере участь в навчанні, визначається dropout_keep_prob входом в мережі. Після отримання обробленого вектора ознак, можна будувати припущення до якого класу належить дана рецензія. Необхідно перемножити матриці і вибрати клас з найбільшим результатом.

Шар softmax допоможе нормалізувати отримані ймовірності. Використовуючи отримані значення, стає можливим визначити функцію втрат. Наша мета - мінімізувати втрати, які як раз вимірюють помилку (похибка) нейронної мережі. Для цього використовується перехресна ентропія.

В роботі використовується функція softmax_cross_entropy_with_logits. Потім береться середнє значення втрат. Як класифікуються пропозиції за допомогою даної архітектури:

Визначити розмірності трьох фрагментів даних: 2, 3 і 4, для кожного фрагмента існує два фільтра;

Кожен фільтр виконує згортку над матрицею пропозицій і генерує карти ознак (feature maps);

Шар 1-max pooling обробляє кожну карту, тобто зберігається найбільша кількість з кожної карти. Отриманий одновимірний вектор ознак з карт об'єднується в вектор ознак для передостаннього шару;

Останній (softmax) шар отримує цей вектор на вхід і використовує його для класифікації пропозиції (бінарна класифікація).

Завдяки використанню згортокових шарів, кількість параметрів в них різко скорочується, і навчити мережу стає набагато простіше. А використовуючи різні види регуляризації (особливо dropout), вийшло значно зменшити перенавчання мережі. Нарешті, застосування ReLU замість сигмоїдальних нейронів допомогло прискорити навчання.

Може здатися, що ідея згорткових нейронних мереж не дуже застосовна для задач природної мови: дійсно, якщо на зображенні сусідні пікселі в основному є частиною одного і того ж об'єкта, то це невірно в разі слів у реченні (частини фраз можуть бути розділені іншими словами) . Можливо, рекурентні нейронні мережі - більш інтуїтивно зрозуміла модель (пропозиція представлено у вигляді дерева розбору), однак це не означає, що згорткові мережі зовсім не застосовні для поставлених в роботі задач.

1.2 Опис сучасних проблем

На основі огляду ми виявили кілька проблем із традиційними підходами до фільтрації веб-сторінок. По-перше, ручний підхід дуже трудомісткий. Хоча такий підхід може досягти високої якості, він, як правило, неможливий за обмежених ресурсів. Підходи, засновані на ключових словах і лексиконі, можуть автоматизувати процес, але обидва вони мають недоліки.

Простий підхід, заснований на ключових словах, не може вирішити проблему багатозначності, тобто слів, що мають більше одного семантичного значення. Наприклад, веб-сторінка, що містить слово рак, цілком може бути медичним висновком про лікування раку легенів або гороскопом для людей, народжених під знаком зодіаку рак. Як результат, такий підхід може не усунути непотрібні сторінки, тим самим знижуючи точність. З іншого боку, оскільки люди часто використовують різні терміни для позначення одного і того ж поняття, наприклад, рак легенів та

новоутворення легенів, цей підхід також може легко пропустити відповідні сторінки, тим самим знижуючи вдалість пошуку.

Підхід, заснований на лексиконі, який використовував оцінку подібності на основі TFIDF між кожним документом та даним доменним лексиконом, полегшує проблему, враховуючи всі терміни в документах. Однак обчислення TFIDF може бути упередженим у збірнику; якщо колекція "галаслива", нерелевантні терміни можуть отримати дуже високий бал IDF і, отже, високий бал TFIDF.

Крім того, що підходи, засновані на ключових словах, що на основі лексикону, не стійко протистоять спаму тексту, що є популярною практикою, коли автори веб-сторінок маніпулюють своїм вмістом сторінки для підвищення рейтингу.

Використання класифікаторів тексту для фільтрації веб-сторінок видається найбільш перспективним підходом, враховуючи їх добру ефективність у традиційній класифікації тексту. Однак одна проблема полягає в тому, що більшість класифікаторів оцінювались з використанням принаймні 2/3 даних для навчання методу вибіркових витяжок. Проблема стає ще гіршою в інших методах оцінки, таких як k-кратна перехресна перевірка, в якій було використано (100 − k)% даних та всі екземпляри, крім одного, відповідно, для навчання. Неможливо отримати такий великий набір навчальних даних при створенні вертикальної пошукової системи, тому що зазвичай тільки невелика кількість документів маркується для класифікації великої кількості документів. Визначення великої кількості документів вручну було б дуже дорогим і трудомістким. Крім того, в більшості існуючих методів класифікації тексту не використовуються знання предметної області, що важливо для вертикальних пошукових систем.

З іншого боку, структура гіперпосилань в Інтернеті була вивчена і зі значним успіхом застосовувалася в дослідженнях інтелектуального аналізу веб-структури. Наприклад, алгоритми PageRank і HITS широко використовуються для ранжирування результатів веб-пошуку. Ці методи можуть допомогти ідентифікувати веб-сторінки з високою якістю і релевантність [2]. Крім того, такі методи можуть допомогти ідентифікувати веб-сторінки, що належать однієї спільноти і, отже, одному домену. Саме ці проблеми необхідно вирішувати в додатках для фільтрації

веб-сторінок. Однак застосування цих методів інтелектуального аналізу веб-структури для фільтрації веб-сторінок мало вивчено. Було б цікаво вивчити питання про ефективність використання інтелектуального аналізу веб-структур при фільтрації веб-сторінок.

1.3 Аналіз існуючих методів добутку знань

1.3.1 Вертикальна пошукова система

Гарна вертикальна пошукова система повинна містити якомога більше релевантних високоякісних сторінок і якомога менше нерелевантних низькоякісних сторінок. З огляду на великий розмір Інтернету і різноманітність контенту, непросто створити всеосяжну і актуальну колекцію для вертикальної пошукової машини. Є дві основні проблеми:

Пошукової системі необхідно знайти URL-адреси, які вказують на відповідні веб-сторінки. Для підвищення ефективності необхідно, щоб система збору сторінок передбачала, яку URL-адресу з найбільшою ймовірністю буде вказувати на відповідний матеріал і, отже, повинен бути отриманий першим.

Після того, як веб-сторінки зібрані, пошуковій системі необхідно визначити зміст і якість колекції, щоб уникнути нерелевантних або неякісних сторінок.

Пошукові системи зазвичай використовують павуків (так званих веб-роботів, сканерів, хробаків або мандрівників) в якості програмного забезпечення для отримання сторінок з Інтернету, рекурсивно переходячи по URL-посиланням на сторінках з використанням стандартного HTTP протоколи [6]. Ці павуки використовують різні алгоритми для управління пошуком. Для вирішення першої проблеми, згаданої вище, були використані наступні методи для пошуку веб-сторінок, що належать до певного домену:

Павуки можуть бути обмежені перебуванням в певних веб-доменах, тому що багато веб-доменів мають спеціалізований зміст. Наприклад, більшість веб-сторінок в домені www.toyota.com матимуть відношення до автомобілів.

Деякі «павуки» обмежуються збором тільки сторінок, максимум фіксованої кількості посилань від початкових URL-адрес або початкових доменів.

Припускаючи, що ближчі сторінки мають більш високі шанси бути релевантними, цей метод не дозволяє павукам піти занадто «далеко» від початкових доменів.

Більш витончені павуки використовують більш досконалі алгоритми пошуку по графам, які аналізують веб-сторінки і гіперпосилання, щоб вирішити, які документи слід завантажити. Чо [8] запропонував пошукового робота «кращий перший», який використовує PageRank як евристики ранжирування; URL-адреси з більш високим рейтингом PageRank спочатку будуть відвідані павуком. Павук, розроблений Маккаллумом [6], використовував навчання з підкріпленням, щоб направляти своїх павуків до пошуку дослідних робіт з університетських веб-сайтів. Focused Crawler знаходить веб-сторінки, які стосуються заздалегідь певного набору тем, на основі прикладів сторінок, наданих користувачем. Крім того, він також аналізує структуру посилань між зібраними веб-сторінками . Пошуковий робот, орієнтований на контекст, використовує наївний баєсовський класифікатор для управління процесом пошуку. Також було запропоновано павук Hopfield Net, заснований на активації поширення. Оцінка вмісту сторінки і оцінка аналізу посилань об'єднуються, щоб визначити, яка URL-адреса має бути відвідана павуком в наступний раз. Павук порівнювався з павуком пошуку в ширину і павуком пошуку, який спочатку отримує кращий, з використанням PageRank як евристики, і результати оцінки показали, що павук Hopfield Net працює краще, ніж два інших. Незважаючи на те, що ці методи мають різні рівні продуктивності і дієвості, в більшості випадків результуюча колекція попередньо зашумлена і вимагає подальшої обробки. Програми фільтрації необхідні для видалення нерелевантних і неякісних сторінок з колекції, яка буде використовуватися в вертикальній пошуковій системі. Використовувані методи фільтрації можна розділити на наступні чотири категорії:

Фахівці в предметної області вручну визначають релевантність кожної веб-сторінки.

У простій автоматичній процедурі релевантність веб-сторінки може бути визначена по появі конкретних ключових слів [8]. Веб-сторінки вважаються

релевантними, якщо вони містять цей ключовий термін, і вважаються нерелевантними в іншому випадку.

TFIDF (частота терміна, зворотна частоті документа) розраховується на основі словника, створеного експертами в предметній області. Потім веб-сторінки порівнюються з набором відповідних документів, і ті, у яких показник подібності перевищує певний поріг, вважаються релевантними.
Методи класифікації тексту, такі як наївний байесовский класифікатор, також можуть бути застосовані до фільтрації веб-сторінок [6].

1.3.2 Класифікація за текстом

Класифікація тексту - це дослідження класифікації текстових документів за заздалегідь визначеними категоріями. Є кілька основних підходів. Наприклад, широко використовується наївний байесовский метод [6]. Він використовує спільні ймовірності слів і категорій для оцінки ймовірності того, що даний документ належить кожній категорії. Документи з ймовірністю вище певного порогу вважаються відносяться до цієї категорії.

Метод k-найближчих сусідів - ще один популярний підхід до класифікації тексту. Для даного документа спочатку визначаються k-сусідів, найбільш схожих на даний документ. Категорії цих сусідів потім використовуються для визначення категорії даного документа. Поріг також використовується для кожної категорії.

Програми нейронних мереж, розроблені для моделювання нейронної системи людини і вивчення патернів шляхом зміни ваг між вузлами на основі навчальних прикладів, також застосовувалися для класифікації тексту. Зазвичай використовується нейронна мережа прямого / зворотного поширення (FF / BP NN). Частоти термінів або TFIDF термінів використовуються в якості вхідних даних для мережі. На основі навчальних прикладів мережа може бути навчена передбачати категорію документа.

Інший новий метод, який використовується в класифікації тексту, називається машиною опорних векторів (SVM), підхід, який намагається знайти гіперплоскість, яка найкращим чином розділяє два класи. Іоахімс вперше застосував SVM до

проблеми класифікації тексту. Було показано, що SVM показала кращу продуктивність серед різних класифікаторів на наборі даних.

Крім загальних текстових документів, також вивчалася класифікація веб-сторінок. Веб-сторінки часто галасливі, але вони надають додаткову інформацію про кожний документ. Наприклад, термінам, зазначеним різними HTML-тегами (такими як заголовки), може бути призначено більш високу вагу, ніж у звичайного тексту. Терміни з сусідніх веб-сторінок також використовувалися в спробі підвищити ефективність класифікації. Однак виявляється, що продуктивність погіршується, тому що часто буває занадто багато сусідніх термінів і занадто багато перехресних зв'язків між різними класами. Було запропоновано використання іншої інформації про сусідні веб-сторінки. Приклади такої інформації включають прогнозовану категорію сусідів сторінки, текст прив'язки, який вказує на сторінку, або витікаючі посилання сторінки на всі інші документи.

1.3.3 PageRank

Алгоритм PageRank вычисляется путем взвешивания каждой входящей ссылки на страницу пропорционально качеству страницы, содержащей входящую ссылку [2]. Качество этих ссылающихся страниц также определяется PageRank. Таким образом, PageRank страницы p вычисляется рекурсивно следующим образом:

$$\mathrm{PageRank}(p) = (1-d) + d \times \sum_{\text{all } q \text{ linking to } p} \left(\frac{\mathrm{PageRank}(q)}{c(q)} \right)$$

d - коефіцієнт демпфування від 0 до 1, c (q) - кількість вихідних ланок в q.

Веб-сторінка має високий рейтинг PageRank, якщо на сторінку посилається безліч інших сторінок, і оцінки будуть ще вище, якщо ці посилаються також є хорошими сторінками (сторінки з високими оцінками PageRank). Також цікаво відзначити, що алгоритм PageRank посилається на модель випадкового блукання. Оцінка PageRank сторінки пропорційна ймовірності того, що випадковий користувач, клацнувши по випадковим посиланням, потрапить на цю сторінку. Ця

оцінка, застосована в комерційній пошуковій системі Google, виявилася дуже ефективною для ранжирування результатів пошуку [2]. Однак час обчислення є основною проблемою при використанні PageRank. Оцінка PageRank кожної веб-сторінки повинна обчислюватися ітеративно, що робить його витратним з точки зору обчислень.

1.3.4 Hyperlink-Induced Topic Search

Клейнберг запропонував міру, звану алгоритмом HITS (пошук по темам, викликаним гіперпосиланнями), який аналогічний PageRank. В алгоритмі HITS авторитетні сторінки визначаються як високоякісні сторінки, які стосуються певної теми або пошуковому запиту. Хаб-сторінки - це сторінки, які не обов'язково самі є авторитетними, але надають покажчики на інші авторитетні сторінки. Сторінка, на яку вказують багато інших, має бути хорошим авторитетом, а сторінка, яка вказує на багато інших, має бути хорошим центром. Грунтуючись на цій інтуїції, можна розрахувати рейтинг авторитету і рейтинг хаба для кожної веб-сторінки наступним чином:

$$\text{AuthorityScore}(p) = \sum_{\text{all } q \text{ linking to } p} (\text{HubScore}(q))$$

$$\text{HubScore}(p) = \sum_{\text{all } r \text{ linking to } p} (\text{AuthorityScore}(r))$$

Сторінка з високим авторитетом вказує на багато вузлів, і сторінка з високим рейтингом - це та, яка вказує на багато авторитетів. Одним із прикладів, який застосовує алгоритм HITS, є пошукова машина Clever, яка отримала вищу оцінку користувачів, ніж каталог Yahoo, складений вручну.

1.3.5 Порівняння PageRank і Hyperlink-Induced Topic Search

Hyperlink-Induced Topic Search:

- HITS заснований на двох значеннях якості: «Оновлення авторитету» і «Оновлення концентратора». Оновлення авторитету розраховується за

кількістю посилань хаба, пов'язаних з авторитетним веб-сайтом, а оновлення хаба розраховується за кількістю авторитетних веб-сайтів, підключених до цього сайту. Загальний результат HITS буде заснований на зв'язку між цими двома значеннями. Фактично він розраховує дві оцінки для кожного документа.

- HITS працює з невеликими підграфами, що представляють зв'язок між веб-сайтами Hub і Authority.
- У HITS збільшення ваги авторитету збільшує вагу вузлових сайтів.
- HITS підраховує бали без індексації.
- HITS особливо використовується в реляційному аналізі веб-сайтів.

PageRank:

- Рейтинг сторінки залежить від ряду різних факторів, особливо від кількості якісних зворотніх посилань. Якісні зворотні посилання - це ті посилання, які відносяться до ніші веб-сайту і розміщуються на веб-сайтах з високим рейтингом сторінок. Таким чином, Page Rank розраховує в основному один бал за документ.
- Page Rank працює на великому веб-графіку, приділяючи особливу увагу всім зворотним посиланням і факторинговим компаніям релевантності.
- У Page Rank якісні зворотні посилання на веб-сайті з високим PR підвищують рейтинг сторінки.
- Page Rank обчислює бал після процесу індексації.
- Page Rank може використовуватися для декількох факторів, таких як Street Rank (рейтинг інших місць, крім веб-сайтів, на основі відвідувань населення). Так само рейтинг сторінки використовується в різних середовищах від інститутів до пошукових роботів.

І HITS, і Page Rank мають свої плюси і переваги, і обидва можуть застосовуватися в різних сценаріях. Page Rank більш популярний, тому що його можна використовувати в різних середовищах, крім веб-пошуку. HITS дуже корисний через його особливу увагу до категоризації вузлових і авторитетних веб-сайтів.

1.4 Постановка задачі

Представляємо кожну веб-сторінку набором функцій, які можна використовувати як вхідні дані для різних алгоритмів машинного навчання, а саме:

- Відображення текстової інформації
- Можливості редіректу на інші веб-ресурси з іншою тематикої.

Необхідно визначити спосіб збору інформації з веб-ресурсу за функціями зазначеними вище. Необхідно розробити алгоритм з використанням нейронної мережі прямого / зворотного розповсюдження, котра могла б бути натренована на моделі для обробки тексту. Модель котра буде оброблена для обробки тексту може використовуватися і для обробки інформації с гіперпосилань цільового веб-ресурсу.

Таким чином можна виділити три основні критерії для розробки алгоритму:

- Тренована модель для розпізнання тематики тексту за ключовими словами.
- Тренована модель для розпізнання тематики сайту на який веде посилання.

За критеріями можна буде скласти загальну характеристику контенту наповненості сайту, в яку може входити розпізнані об’єкти зображення, ключові слова тексту визначені за величиною ITIDF, та ключові слова сторінок на котрі ведуть посилання.

Таким чином в даній роботі ставляться такі завдання:

1. Провести порівняльний аналіз методів.

Необхідно порівняти існуючі методи, котрі будуть виділені після ознайомлення з існуючими джерелами інформації. При порівнянні слід відзначити слабкі та сильні сторони, котрих у розробляємому методі хотілося б позбутись, або реалізувати. Необхідно розглянути такі критерії як складність реалізації, вимогливість до аппаратних ресурсів, обсяг даних, котрий можна обробляти за короткий час.

2. Розробити метод

Розробити метод, котрий враховував би всі зазначені раніше недоліки та плюси рішень, призначених для вирішення цього самого питання. Необхідно реалізувати в методі найбільш вдале поєднання плюсів з аналогічних рішень для досягнення найбільшої ефективності.

3. Провести дослідження ефективності розробленого методу

Необхідно більш глибше оцінити сильні та слабкі сторони розробленого алгоритму з метою визначення шляхів можливого його удосконалення в майбутньому. Провести порівняльний аналіз зі схожими методами. Розглянути та визначити більш зручні сфери для використання методу.

4. Спроектувати інформаційну систему

Спроектувати інформаційну систему, котра реалізує розроблений метод, та дозволяє в повній мірі використовувати його потенціал. Визначити до котрих сфер метод в даній інформаційній системі буде практичніше використовувати та визначити чи може спроектована інформаційна система в майбутньому бути масштабована до більшої пошукової системи для різних веб-додатків.

2 МЕТОДИ ІНТЕЛЕКТУАЛЬНОГО АНАЛІЗУ

2.1 Обгрунтування вибору метода аналізу контента

Щоб вирішити проблеми з поточними підходами до фільтрації веб-сторінок, пропонується підхід, який об'єднує веб-контент і аналіз структури в веб-фільтрацію. Замість того, щоб представляти кожен документ у вигляді пакету слів, кожна веб-сторінка представлена обмеженою кількістю елементів контенту і посилань. Це зменшує розмірність (кількість використовуваних атрибутів) класифікатора і, отже, кількість необхідних навчальних прикладів. Характеристики веб-структури також можуть бути включені в ці «веб-функції».

Можна визначити, що в цілому релевантність і якість веб-сторінки можуть бути відображені в таких аспектах:

- зміст самої сторінки
- зміст сусідніх сторінок
- інформація про посилання на сторінку.

Для кожного аспекту визначені кілька функцій.

2.1.1 Контент сторінок

Зміст сторінки, ймовірно, є основним фактором при визначенні того, чи відповідає сторінка певному домену. Вміст кожної сторінки представляється набором показників, а не вектором слів. Застосуємо автоматичний підхід, який витягує всі терміни зі сторінки і порівнює їх з лексиконом предметної області. Затим переглянемо, як на кількість релевантних термінів, що з'явилися в заголовку сторінки, так і на оцінки TFIDF термінів, що з'явилися в тілі сторінки. Були визначені дві характеристики:

- Title (p) = кількість термінів в заголовку сторінки p, знайдених в лексиконі домену.
- TFIDF (p) = сума TFIDF термінів на сторінці p, знайдених в лексиконі домену.

2.1.2 Контент сусідніх сторінок

Щоб включити вміст сусідів сторінки, можна використовувати оцінку з кожного сусіднього документа замість включення всіх термінів з сусідніх документів, що здається швидше шкідливим, ніж корисним[12]. У підході розглядаються три типи сусідів: вхідні, вихідні і однорівневі. Для будь-якої сторінки p входні сусіди (батьки) - це набір всіх сторінок, гіперпосилання яких вказує на p. Вихідні сусіди - це сторінки, гіперпосилання яких знаходяться на сторінці. Сторінки-брати - це ті сторінки, на які вказує один з батьків p. Приклад показаний на рис. 7. У цьому прикладі сторінки a, b і c є вхідними сусідами p; сторінки f і g є вихідними сусідами p, а сторінки d і e є братами p.

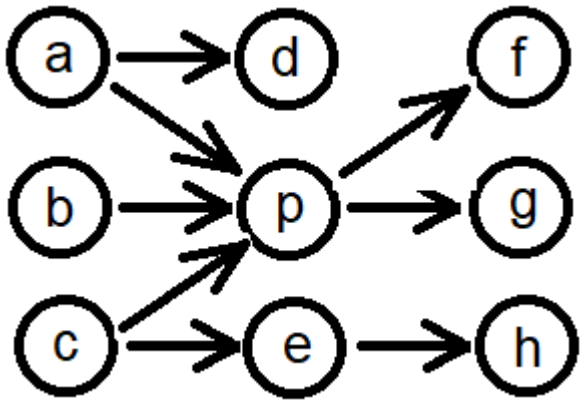

Рис. 2.1 – Приклади вхідних, вихідних та братських сторінок.

Дві оцінки змісту (оцінка заголовка і TFIDF) сусідніх документів були визначені аналогічно тим, які були створені в попередньому аспекті. Були використані шість функцій: середні значення двох оцінок для всіх вхідних сусідів, середні значення для всіх вихідних сусідів і середні значення для всіх братів.

- InTitle (p) = Середнє (кількість термінів в заголовку сторінки q, знайдених в лексиконі домену) для всіх вхідних сторінок q з p.
- InTFIDF (p) = Середнє (сума TFIDF термінів на сторінці q, знайдених в лексиконі домену) для всіх вхідних сторінок q з p.

- OutTitle (p) = Середнє (кількість термінів в заголовку сторінки r, знайдених в лексиконі домену) для всіх вихідних сторінок r з p.
- OutTFIDF (p) = Середнє (сума TFIDF термінів на сторінці r, знайдених в лексиконі домену) для всіх вихідних сторінок r з p.
- SiblingTitle (p) = Середнє (кількість термінів в заголовку сторінок, знайдених в лексиконі домену) для всіх сторінок-братів p.
- SiblingTFIDF (p) = Середнє (сума TFIDF термінів на сторінках, знайдених в лексиконі домену) для всіх дочірніх сторінок s з p.

2.1.3 Аналіз посилань

Зв'язок (аналіз посилань) використовувався для визначення якості сторінки. Аналіз посилань, такий як кількість внутрішніх посилань, HITS і PageRank, був корисний у багатьох веб-додатках, таких як ранжування результатів пошуку [2], але не використовувався при класифікації тексту. Щоб включити оцінки аналізу посилань в наш підхід до фільтрації, в якості функцій використовувалися шість оцінок, а саме оцінка хаба, оцінка авторитету, оцінка PageRank, кількість вхідних посилань, кількість вихідних посилань і кількість відповідних термінів в якірних текстах.

- Hub (p) = оцінка Hub сторінки p, розрахована за допомогою алгоритму HITS.
- Authority (p) = оцінка авторитету сторінки p, розрахована за допомогою алгоритму HITS.
- PageRank (p) = оцінка PageRank сторінки p.
- Inlinks (p) = кількість вхідних посилань, що вказують на p.
- Outlinks (p) = Кількість вихідних посилань з p.
- Anchor (p) = кількість термінів у якірних текстах, що описують сторінку p, знайдених в лексиконі домену.

2.2 FF/BP NN Текстовий класифікатор

Всього було ідентифіковано 14 ознак, які можуть використовуватися в якості вхідних значень для класифікатора. Як класифікатор ми використовували нейронну мережу (NN) і машину опорних векторів (SVM). Нейронна мережа прямого / зворотного поширення (FF / BP NN) була прийнята через її надійність і широке використання в класифікації. Розроблений алгоритм резюмується далі.

2.2.1 Ініціалізація мережі

Спочатку створюється нейронна мережа з трьома шарами: вхідним, прихованим і вихідним. Вихідний шар складається з єдиного вихідного вузла, який визначає релевантність сторінки (чи повинна веб-сторінка бути включена в вертикальну пошукову систему). Кількість вузлів в прихованому шарі встановлюється рівною 16, а швидкість навчання - 0,10. Ці параметри встановлюються на основі деяких початкових експериментів з використанням невеликого набору даних.

2.2.2 Навчання і налаштування мережі

Навчальні документи передаються в мережу для навчання. Потім навчальні документи розділяються на два набори: 80% документів використовуються для навчання, а 20% - для налаштування. У мережі представлені 14 характеристик кожного навчального документа, а також двійкова оцінка, що показує, чи актуальний документ. Оцінка кожної ознаки нормалізована до значення від 0 до 1 з використанням сигмоїдальной функції. Потім мережа оновлює ваги свого з'єднання на основі навчальних документів. Після того, як всі навчальні документи пройшли через мережу один раз, документи налаштування представлені в мережу і записується середньоквадратична помилка (MSE) мережі.

2.2.3 Тестування

Кожен тестовий документ представляється навченій мережі, яка спробувала передбачити, чи актуальний цей документ. Прогнози записуються і використовуються для розрахунку показників ефективності.

2.3 Класифікатор тексту SVM

Щоб забезпечити краще порівняння, також використовується машина опорних векторів через її видатні характеристики в традиційній класифікації тексту. Класифікатор провів класифікацію на основі того ж набору оцінок характеристик. Класифікатор SVM включає наступні кроки:

2.3.1 Вибір моделі

Для порівняння з класифікатором SVM обрана лінійна функція ядра, оскільки вона проста, швидко навчається і забезпечує продуктивність, яку можна порівняти з продуктивністю нелінійних моделей, таких як поліноміальні класифікатори та радіальні базисні функції в додатках класифікації тексту. Кожен навчальний приклад був представлений у вигляді вектора з 14 обраних характеристик і представлений SVM для вивчення ваг характеристик.

2.3.2 Тестування

Подібно алгоритму нейронної мережі, SVM намагається передбачити на основі своєї моделі класифікації, чи відповідає кожен документ в наборі тестування обраній області. Результати записуються і використовуються для оцінки.

2.4 Коефіцієнт Жаккара

Індекс Жаккара, також відомий як коефіцієнт подібності Жаккара, являє собою статистику, яка використовується для розуміння подібності між наборами зразків. Вимірювання підкреслює схожість між кінцевими наборами зразків і формально визначається як розмір перетину, поділений на розмір об'єднання наборів зразків. Математичне подання індексу записується як:

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}$$

Подібно індексу Жаккара, який є мірою подібності, відстань Жаккара вимірює відмінність між наборами зразків. Відстань Жаккара розраховується шляхом знаходження індексу Жаккара і віднімання його з 1 або, альтернативно, ділення різниць на перетині двох наборів. Формула для відстані Жаккар представлена як:

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}$$

Розбиваючи формулу, індекс Жаккара, по суті, являє собою число в обох наборах, поділене на число в будь-якому наборі, помножене на 100. Це дасть процентне вимірювання подібності між двома наборами зразків. Відповідно, щоб знайти відстань Жаккара, просто треба відняти процентне значення з 1.

Згорткові нейронні мережі, які зазвичай використовуються в додатках для ідентифікації зображень, застосовують вимірювання індексу Жаккара як спосіб концептуалізації точності виявлення об'єктів. Наприклад, якщо алгоритму комп'ютерного зору доручено виявляти особи на зображенні, індекс Жаккара може кількісно оцінити схожість між комп'ютерної ідентифікацією ,осіб і даними навчання.

Тому ми можемо використати індекс Жаккара для перевірки нейронних мереж в двох запропонованих та розроблених методах, щоб визначити точність визначення контенту мережами та порівняти результати.

2.5 Стеммер Портера

Алгоритм визначення слів Портера (або «Стеммер Портера») - це процес видалення загальних морфологічних і флексійних закінчень з слів в мові. Його основне використання - як частина процесу нормалізації термінів, який зазвичай виконується при налаштуванні систем пошуку інформації.

Видалення суфіксів автоматичним способом - операція, яка особливо корисна в області пошуку інформації. У типовому IR-середовищі у одного є зібрання документів, кожен описаний словами в назві документа і, можливо, слова в анотації документа. Ігнорування питання про те, де саме відбуваються слова, ми можемо сказати, що документ представлений вектором слів. Терміни із загальною основою будуть зазвичай мати схожі значення,

Часто продуктивність IR-системи поліпшується, якщо групи термінів об'єднані в один термін. Це можна зробити шляхом видалення різних суфіксів. Крім того, процес видалення суфіксів знизить загальну кількість термінів в IR-системі, і, отже, зменшити розмір і складність даних в системі, що завжди вигідно.

2.7 Реалізація підходів

У підході, заснованому на лексиці, для вилучення словосполучень з кожного документа можна використати різноманітні мови програмування, проте одним з найлегших засобів для роботи з інструментами навчання є мова програмування Python. Бібліотека Scikit-learn - найпоширеніший вибір для вирішення завдань класичного машинного навчання. Вона надає широкий вибір алгоритмів навчання з учителем і без вчителя. Одна з основних переваг бібліотеки полягає в тому, що вона працює на основі декількох поширених математичних бібліотек, і легко інтегрує їх один з одним.

3 РОЗРОБКА МОДЕЛІ

3.1 Умови попередньої підготовки веб-сторінки

Модель в основному опиратиметься на аналіз текстового контенту, який використовує метод ентропії як схеми зважування термінів. Модель складається з декількох частин: витяг веб-сторінки, попередня обробка (складається з синтаксичного аналізу HTML, виділення тексту і зупинки), функція на основі профілю класу (CPBF) і класифікатор штучної нейронної мережі (АНН). На рис. 8 показаний огляд моделі. Назвемо модель Web Page Classification Based on PCA та в подальшому будемо скорочено називати WPCB.

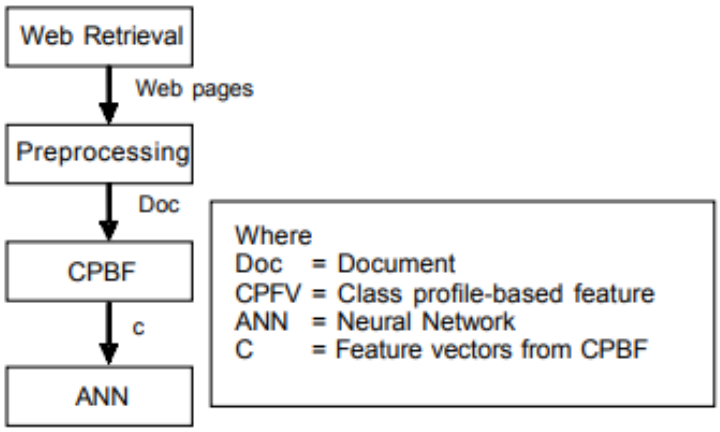

Рис. 3.1 – Огляд моделі

По-перше, під час пошукової частини веб-робот переглядає веб-сторінки з Інтернету, котрі будут сусідніми для обраної сторінкі контент якої буде визначатися. Ці веб-сторінки пройдуть етап обробки і зупинять процес світової фільтрації, щоб зменшити шум в частині попередньої обробки. Відносини всередині функцій і веб-сторінок будуть побудовані і розраховані з використанням схеми зважування ентропійних термінів в CPBF. Тим часом, кожна категорія функцій буде збережена в

іншому профілі класу в якості вхідних даних для класифікатора АНН. Функції будуть навчені, а веб-сторінки будуть класифіковані за класифікатором АНН.

3.2 Витяг веб-сторінки

Витяг веб-сторінки - це частина, яка використовує веб-робота для вилучення потрібної веб-сторінки в базу даних. Операційний процес отримання цієї веб-сторінки показаний на рисунку 9. Однак частина отримання веб-сторінки також може виконуватися вручну за допомогою веб-браузера. Ця частина може бути виконана за допомогою веб-краулера або веб-браузера, якщо веб-сторінки витягуються в міру необхідності і зберігаються в базі даних. Якщо коротко, основне завдання пошуку веб-сторінки - отримати веб-ресурси з WWW і скопіювати їх в базу даних, яка буде зберігати їх для подальшого аналізу.

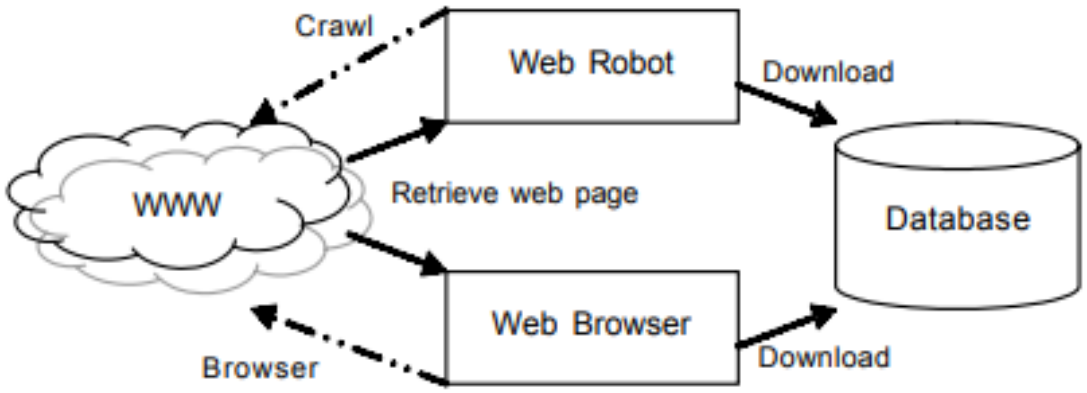

Рис. 3.2 – Витяг веб-сторінки

Парсинг сайтів - автоматизований процес вилучення даних або інформації з веб-сторінок. Після вилучення необхідних даних по ним можна здійснювати пошук, переформатувати їх, копіювати і так далі.

3.2.1 Mozenda

Парсер Mozenda допомагає компаніям в зборі та організації даних найбільш економічним і ефективним способом. Компанія пропонує хмарну архітектуру, яка забезпечує масштабованість, простоту використання і швидке розгортання.

Впровадити Mozenda можна досить швидко, до того ж розгорнути це ПО можна за лічені хвилини на рівні бізнес-підрозділу без будь-якої участі IT-відділу. Його простий point-and-click інтерфейс допомагає користувачам створювати проекти і швидко експортувати результати, самостійно або за розкладом.

Завдяки простоті інтеграції користувачі можуть публікувати отримані результати в форматі CSV, TSV, XML або JSON.

3.2.2 Automation Anywhere

Компанія Automation Anywhere складається з групи експертів, які зосереджені на наданні повністю розуміють і гнучких процесів створення ботів, призначених для автоматизації завдань.

Такі боти не тільки прості у використанні, але і досить потужні, щоб автоматизувати завдання будь-якого рівня складності. Це єдина роботизована платформа, розроблена для сучасних підприємств, яка може створювати програмних ботів для автоматизації завдань від початку і до кінця.

3.2.3 Beautiful Soup

Надаючи вам прості кроки і ідіоми Python для навігації, Beautiful Soup дає доступ до інструментів вилучення будь-якої необхідної інформації. Програмне забезпечення для парсинга веб-сторінок автоматично перетворює вхідні документи в Unicode і вихідні документи в UTF-8. Це дозволяє вам використовувати різні стратегії парсинга або змінювати швидкість і гнучкість процесів.

3.2.4 Web Harvy

Інтерфейс Web Harvy дозволяє легко вибрати елементи з потрібною інформацією. Витягнуті дані можуть бути збережені в файли CSV, JSON, XML або в базі даних SQL.

Програмний продукт є багаторівневою системою парсинга категорій, яка може відстежувати посилання на категорії будь-яких рівнів та видавати дані зі сторінок зі списками. Інструмент пропонує вам велику гнучкість і дає можливість використовувати регулярні вирази.

3.2.5 Content Grabber

Простий інтерфейс Content Grabber має прекрасну можливість автоматичного виявлення і налаштування команд. Він миттєво створює списки контенту, обробляє нумерацію сторінок і веб-форм, а також сам викачує або закачує файли.

Content Grabber може витягувати контент з будь-якого сайту, а потім зберігати його у вигляді структурованих даних в потрібному вам форматі, будь то таблиці Excel, XML, CSV або більшість використовуваних зараз баз даних. Його висока продуктивність і стабільність забезпечується оптимізованими браузерами, а також налагодженим процесом парсинга.

Компанія також розробляє і продає Content Grabber Enterprise (CG Enterprise), який є преміальним продуктом для отримання даних з сайтів, і він сьогодні вважаємо найсучаснішим інструментом на ринку.

3.2.6 FMiner

FMiner підтримує як Windows, так і Mac, він має інтуїтивно зрозумілий інтерфейс і надзвичайно простий у використанні. У цієї програми потужний

інструмент візуального дизайну, який фіксує кожен ваш крок і моделює процес збору інформації, коли ви взаємодієте з цільовими сторінками сайту.

FMiner дозволяє збирати дані з різних веб-сайтів, включаючи онлайн-каталоги продукції, оголошення про нерухомість і каталоги жовтих сторінок.

3.2.7 Import.io

Import.io - інструмент парсинга, який дозволяє без проблем отримувати дані з сайтів. Все, що потрібно зробити, це ввести URL-адресу, і система негайно перетворить сторінки в дані.

Це програмне забезпечення є ідеальним рішенням для моніторингу числових значень, щоб, наприклад, визначити очікування ринку. Він допомагає вам генерувати якісні ліди і надає щоденні або щомісячні оновлення, щоб допомогти відстежувати дії конкурентів.

3.2.8 Visual Web Ripper

Visual Web Ripper - це просунутий парсер для веб-сторінок, який дозволяє витягувати дані з динамічних сторінок, з каталогів продуктів, сайтів з оголошеннями або фінансових сайтів.

Після отримання даних він поміщає їх в зручну і структуровану базу даних, електронну таблицю, файл CSV або XML. Оскільки він може обробляти сайти з підтримкою AJAX і багаторазово відправляти форми з усіма можливими значеннями, він може працювати там, де інші парсери пасують.

3.2.9 Webhose.io

Webhose.io за запитом надає вам доступ до структурованих веб-даних. Це дозволяє створювати, запускати і масштабувати операції з великими даними незалежно від того, чи є ви дослідником, підприємцем або керівником компанії.

Програмне забезпечення структурує, зберігає та індексує мільйони веб-сторінок в день в різних вертикалях, таких як новини, блоги та онлайн-обговорення.

3.2.10 Scrapinghub platform

Scrapinghub Platform відома тим, що створює, розгортає і запускає веб-краулери, забезпечуючи отримання новітньої інформації. Дані можна легко переглянути в красивому інтерфейсі. Програмне забезпечення також надає вам платформу з відкритим вихідним кодом під назвою Portia, яка призначена для парсинга веб-сайтів.

Ви можете створювати шаблони, натискаючи на елементи на сторінці, а Portia обробить все інше. Компанія також створює автоматизовану утиліту, яка видаляє схожі сторінки з веб-сайту.

3.3 Передобробка

3.3.1 Розробка модуля передобробки

Новинні веб-сторінки мають різні характеристики, при цьому довжина тексту для кожної з них є змінною. Крім того, структури сторінок розрізняються по використанню тегів. Більш того, на цих сторінках існує величезна кількість різних слів, оскільки немає обмежень на використання слів на новинних веб-сторінках. На сторінках веб-новин є багато категорій новин, таких як спорт, погода, політика, економіка. У кожній категорії є багато різних класів. Наприклад, класи, які існують в бізнес-категорії, - це фондовий ринок, фінансові інвестиції, особисті фінанси. Для

класифікації новинних веб-сторінок ми пропонуємо WPCM, який використовує PCA і CPBF в якості вхідних даних для нейронних мереж. По-перше, використовуючи алгоритм PCA, можна зменшити вихідні вектори даних до невеликого числа відповідних функцій. Потім ми об'єднуємо ці функції в CPBF перед тим, як вводити їх в нейронні мережі для класифікації, як показано на рисунку 11.

Веб-сторінки в базі даних будуть проходити аналіз HTML, щоб перетворити веб-сторінки в текстові документи. Документи будуть виконувати зупинку перед їх передачею в секцію CPBF. Стоп-список - це словник, який містить найбільш часто зустрічаємі слова, такі як «Я», «Ви», «і» і т.д. Зупинка - це процес, який фільтрує ті загальні слова, які існують в веб-документі, за допомогою стоп-списку. Стемінг грає важливу роль в зменшенні частоти виникнення термінів, які мають аналогічне значення в тому ж документі. Це процес вилучення кожного слова з веб-документа шляхом скорочення його до можливого кореневого слова. Наприклад, «краса» і «красивий» мають однакове значення. В результаті алгоритм виділення коренів зведе його до кореневого слова «краса». Робочий процес попередньої обробки показаний на рисунку 10. Вихідні дані цієї попередньої обробки стануть вхідними даними для функцій на основі профілю класів (CPBF) для подальшого процесу.

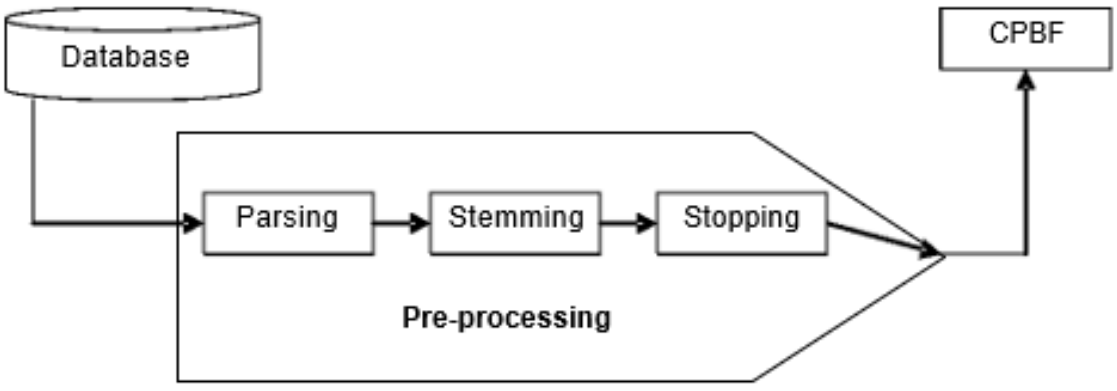

Рис. 3.3 – Передобробка веб-ресурсів

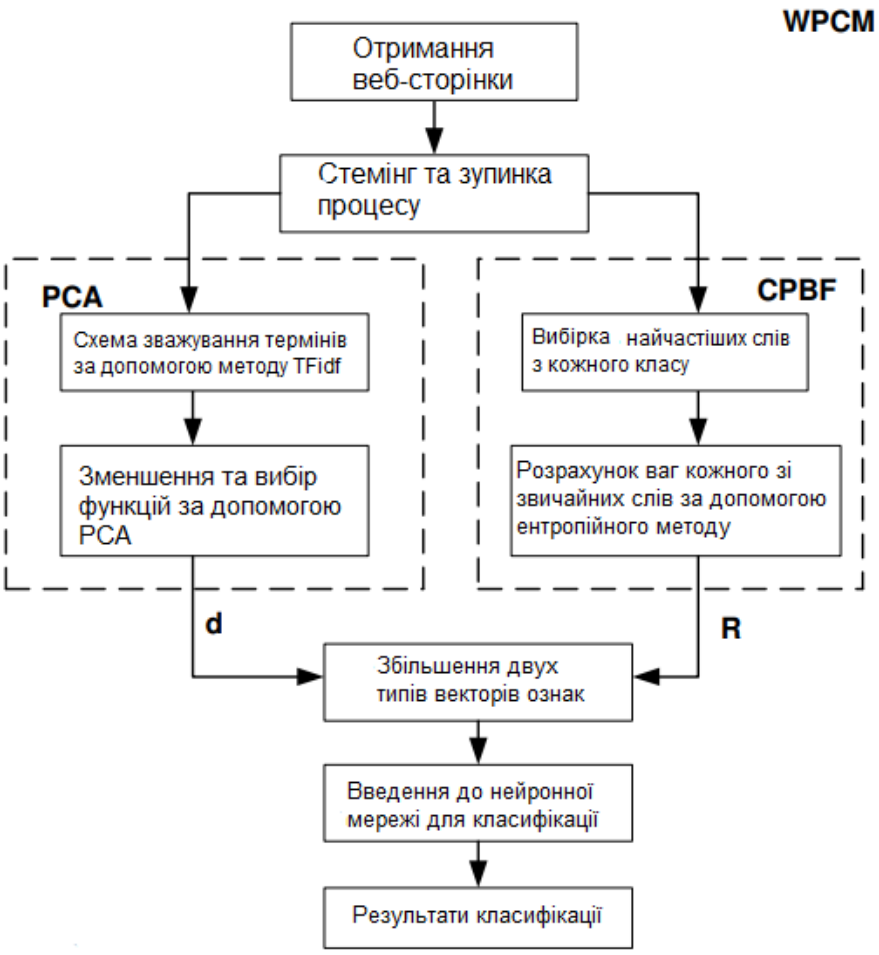

Рис. 3.4 - Процес класифікації веб-сторінки із використанням методу WPCM.

Щоб почати збір інформації потрібно визначити область дослідження. Зазвичай веб-сайт складається з веб-сторінок, котрі представлені у вигляді текстових файлів з розширенням html. Веб-сторінки можуть містити посилання на інші файли в інших форматах, а також гіперпосилання. У сторінках можуть міститися гіперпосилання або ж посилання на інші формати файлів. Гіперпосилання представляють з себе слово, речення або медіа-об’єкт, котрий посилається на іншу частину сторінки, або й зовсім на іншу сторінку.

При створенні веб-сторінки використовують так звані HTML-теги. У кожного тегу є призначення, він передає браузеру інформацію стосовно того, як він має відображатися. Також використовуються класи та ідентифікатори, котрі дозволяють використовувати каскадні таблиці стилів (CSS) для зміни відображення. При написанні на HTML веб-сторінки, її структура, як правило, ділиться на 3 частини:

header, content, footer.

В блоках знаходиться унікальна інформація, стосовно веб-сторінки – відповідно назва, основна інформація та контактна інформація про сайт і автора. Наприклад, вниз веб-сторінки зазвичай встановлюється елемент <footer> (від англ. Footer - нижній колонтитул, підвал), що задає «підвал» сайту або розділу веб-сторінки, в ньому може розташовуватися ім'я автора, дата документа, контактна і правова інформація.

Теги також вводяться, аби збагатити семантичний зміст сторінки. Це може допомогти браузеру у з'ясуванні типу мультимедійних об'єктів, будь то зображення, аудіо або відео об”єкти,допомогти в розширенні блоків текстової інформації при визначенні теми певного блоку. З приходом стандарту HTML5 розробники почали використовувати саме його, бо він може забезпечити легкочитаємий код, що спростить аналіз у ньому пошуковими інструментами.

При класифікації треба врахувати багато особливостей та факторів, помимо тектової інформації. Різноманітний порталам та сайтам притаманний контент, котрий буде дуже відрізнятися на різних гіперпосиланнях, тому важливо враховувати інформацію с батьківських та дочірніх сторінок. Проте при визначенні треба обмежуватись рамками одного домену. Треба враховувати усі ці фактори, адже гнучка та правильна вибірка даних стосовно веб-ресурсу є найважливішою умовою при аналізі контенту.

Щоб отримати дані з веб-сторінок треба написати окремий модуль, або скористатися сервісами парсингу сайтів, котрі були вказані вище. При написанні модулю інформація повинна поділятися на різні категорії, з котрими зручно окремо працювати.

Категорії мають бути наступними:

– посилання (теги «a» і «href»);
– text;
– headers;
– спеціалізовані теги (head, title та meta);
– теги конкретизації вибірки (strong, div, теги навігації);
– медіа (теги «src» та «alt»);

Через те, що зараз майже всі сторінки відповідають стандарту HTML5, то ми маємо можливість легко розкласти її на окремі семантичні одиниці , що спростить питання класифікації. Завдяки цьому можна виділити ще декілька категорій:

– content («div.content, article, section»);
– header («#header, .header, header»);
– footer («#footer, .footer, footer»);
– navigation («.nav a, #nav a, #menu a, .menu a, ul a»).

При цьому ми можемо враховувати як теги HTML5, так і теги HTML 4-ї версії. При наявності відповідного ідентифікатору використовується тег 4-ї версії, інакше тег HTML5.

Коли стали відомі основні категорії контенту, котрі можуть бути є можливість розглянути варіанти для збереження зібраних даних. Оскільки модель може піддаватись модифікації, виникає необхідність зберігати дані так, аби їх можна було легко використовувати при дослідженні впливу якихось параметрів.

Так як виникає необхідність у використанні відношень, то зберігання в файлі буде не оптимальним рішенням, а подолати цю проблему допоможе використання бази даних, де це вже реалізовано. Коли буде виконуватись збір даних, необхідно також зберігати дані необробленими, тобто безпосередньо код веб-сторінок. Завдяки такому методу зберігання організація даних, пошук та сортування будуть досить простими, а легким запитом до БД ми можемо отримати вибірку.

Щоб зберігати інформацію у базі даних для неї треба розробити структуру. В таблицях має міститися інформація про те, що міститься на сайті, інформація про домен, батківську сторінку та категорії сайту.

Основні дані будуть представлені в таблиці Pages:

– url - адреса веб-сторінки;
– html_code – безпосередньо код веб-сторінки;
– html_text – основний текст веб-сторінки;
– title - заголовок веб-сторінки;
– meta_keyword - ключові слова метаінформації;
– meta_description – опис веб-сторінки;
– p_text - вміст тегів <p>;
– content – контент, що міститься у однойменному блоці;
– header – інформація, що міститься у шапці;
– footer – інформація, що міститься у футері;
– error – код, котрий повертає сторінка при запиті;
– domain_id – id до таблиці Domains.

Нижче представлена структура таблиць, з якої буде складатися база даних для зберігання інформації:

– Domains – містить домени;
– Category_Page - містить категорії веб-сторінок;
– Categories –містить можливі категорії;
– Relatives – містить зв'язки між сторінками;
– Pages – містить веб-сторінки;
– Tags – містить інформацію розбиту за окремими тегами с веб-сторінки.

Схема представлена на рисунку 12.

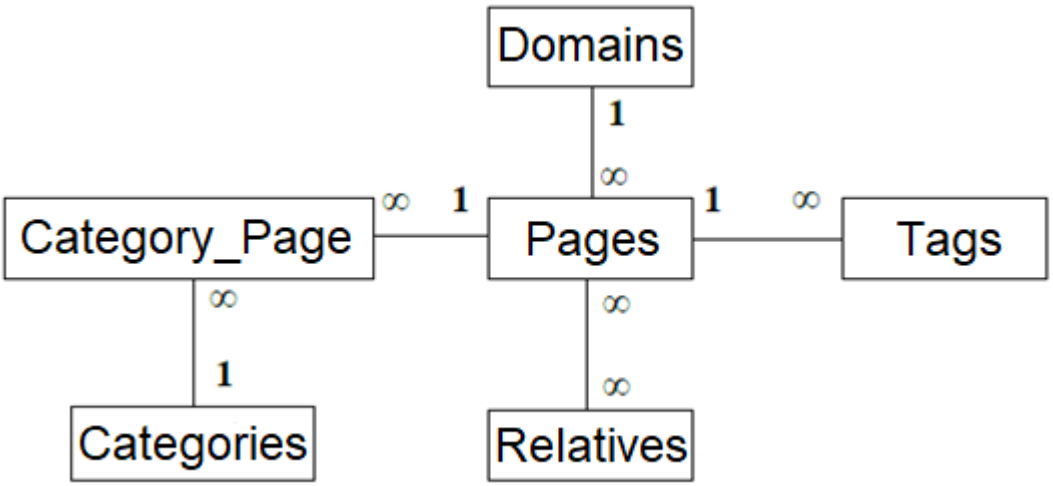

Рис. 3.5 - Схема структури бази даних

Зберігатися в базі даних будуть дані зі сторінок в різних таблицях, об’єднані за групами – доменами, усіма категоріями, відношеннями між сторінками та з кодом, повертаємим при запиті.

Аби прискорити роботу можна використати списки сторінок із файлів. В базі будуть зберігатись дані розбиті по групах з веб-сторінок.

Аби працювати з адресами можно представити такі етапи роботи:

– Зчитати з файлу адресу.

– Зберегти адресу з кодом 1 до бази даних, що буде означати, можливість завантаження веб-сторінки.

– Вилучимо з бази усі сторінки в котрих вказаний код помилки – 1. Тобто сторінки, готові до завантаження.

– Вилучені посилання передаються для обробки, а саме отримання сторінки, подальше обробка та збреження отриманих даних, котрі знаходяться за певною адресою. Таким чином отримуються абсолютні адреси сторінок.

– Затим необхідно обробити абсолютні адреси, таким самим чином, як і на попередньому кроці.

– Виконується перевірка, аби адреси посилань-дітей були унікальними.

– Посилання-діти записуються до списку обробки у вигляді [id-батька * -1]. Потім обираються записи з бази, що мають від’ємні коди, де батко – цей самий код по модулю.

В ході роботи було поставлено задачу з розробки окремого модуля для збору даних с веб-сторінок.

В основі модуля будуть 4 базових класи:

– AppCrawler;
– DBContext;
– ProxyConnection;
– Page.

Основним класом являється AppCrawler, який працює з об'єктами класів DbContext, ProxyConnection та реалізує інтерфейси IApp і IPage.

Модуль запускається на виконання через екземпляр класу AppCrawler. В конструкторі класу створюється об’єкт класу DBContext до котрого передаються дані для підключення бази даних. Безпосередньо DBContext використовується для того, аби спростити взаємодію з базою даних. Цей клас працює з підключенням та управлінням базою даних, в котрій збережена інформація зі сторінок.

Одразу як буде створено об’єкт AppCrawler модуль буде готовий до запуску через виконання методу launch. Цей метод відповідає за обробку – він виконує перевірку наявності списку з посиланнями, затим передає цей список до методу prepareQueue.

В свою чергу, отримавши список, метод prepareQueue передає його до бази даних ,як тимчасову чергу з кодом 0 та в якості результату повертає набір RelustSet з яким буде виконуватись подальша обробка. Використання безпосередньо черги дасть можливість використовувати необхідні зупинки в роботі, при необхідності.

Як набір ResultSet буде отримано, з нього буде вилучено дані про сторінку, визвано метод Parse через котрий буде передано до бази даних отримані дані та нащадки.

Parse відпрацює із самою сторінкої та її посиланнями-дітьми. “Дітей” метод передає у вигляді тимчасових сторінок з ключем [id-батька * - 1], котрий дає можливість обрати всіх дітей на наступній ітерації складання. На вхід метод отримує сторінку, використовуючи метод getRootPage.

getRootPage – метод який повертає сторінку, котру реалізує інтерфейс IPage. Клас- Page це абстрактний клас, реалізує IPage та виконує роботу із самим сторінками, таку як обробка адрес, транспортування даних до БД, обробка HTTP відповідей.

Є декілька типів посилань, котрі не потрібні при зборі інформації. Оброблятися вони будуть методом PrepareLink, що відфільтровує та зводить посилання до загального виду, а потім повертає у якості результату. Відфільтровувати необхідно скрипти на мові JavaScript, хеш-символи та mailto елементи. Викликається метод перед роботою з будь-яким посиланням, що було отримано. Безпосередньо переводить адресу у вигляд http(s)://link.domen/ У конструкторі буде визначено домен веб-сторінки із її адреси та додано до бази даних. Домен з посилання отримуватиметься через метод getDomain.

getDomain – метод, через котрий можна отримати домен другого рівня з посилання. Щоб це зробити – необхідно текст між «//» і «/», що містить крапку, вирізати. Також треба видалити з посилання параметри GET-запиту та виконати перевірку на відповідність домену першого рівня. Це можна реалізувати визначивши те, що не є доменом – html, php, asp та ін.

Метод Page може приймати дані, як із Інтернету, так і з бази даних тому, залежно від вхідних даних, він реалізує відповідні методи для отривання коду через конструктор класу.

Коли отримується сторінка з Інтернету, то треба провести фільтрацію по вмісту: css, javascript, csv (при цьому не враховуючи xml.), media, application,

image. Якщо сторінка повертає код 200, то це означає що вона успішно отримана. Посилання за заголовком Location буде змінюватись при перенаправленні. Якщо ж відповідь сторінки буде не 200, то в базу даних буде зроблено запис за поверненим кодом помилки і робота зі сторінкою буде перервана.

Метод Parse відповідає за передачу та отримання даних в базу даних із коду HTML. Даний метод повертає набір пройдених перевірку абсолютних посилань-дітей, після того як робить вибірку даних із цільової сторінки. Потім метод передає дані до бази через клас DBContext. В решті, після того, як дані будуть передані виконується фільтрація посилань-дітей методом getAbsLink.

getAbsLink – метод, котрий обробляє посилання і повертає його в абсолютному вигляді. Посилання бувають відносними або ж абсолютними. За відносними посиланнями отримувати дані немає можливості, тому цей метод виконується для попередньої перевірки та, в разі необхідності, обробки відносного посилання в абсолютне. Спочатку виконується попередня підготовка посилання. Якщо посилання є валідним, то ми визначаємо його домен, і якщо він в наявності, то це вже абсолютне посилання, інакше треба з’єднати посилання, що оброблюється та посилання предка і повернути його в якості результату метода.

Після того, як буде додано список вихідних посилань з їхніми даними починається додавання їх підсторінок. В класі AppCrawler реалізовано метод setProxy, для додання можливості використання проксі сервера. Цей метод приймає вхідні дані у вигляді екземпляру класу ProxyConnection та встановлює з’єднання з проксі-сервером. Клас ProxyConnection відповідає за роботу з проксі-сервером та підключення до нього.

Виконуючи принцип поліморфізму було реалізовано можливість змінювати вид транзакцій таблиці Pages (для додання і оновлення записів), а також додавання веб-сторінок з різних видів джерел.

3.3.2 Зменшення функцій за допомогою PCA

Припустимо, що у нас є A, яка представляє собою матрицю з вагою термінів документа, як показано нижче,

$$\mathbf{A} = \begin{pmatrix} x_{11} & x_{12} & x_{1k} & \cdots & x_{1m} \\ x_{21} & x_{22} & x_{2k} & \cdots & x_{2m} \\ x_{31} & x_{32} & x_{3k} & \cdots & \dots \\ \vdots & \vdots & \vdots & \ddots & \dots \\ x_{n1} & x_{n2} & x_{nk} & \cdots & x_{nm} \end{pmatrix}$$

де x_{jk} - це вага термінів, які існують в колекції документів. Визначення j; k; m; і n були описані в попередньому абзаці. Щоб обчислити основні компоненти матриці даних A, необхідно виконати кілька кроків. Середнє значення m змінних в матриці даних A буде обчислено наступним чином

$$\overline{x}_k = \frac{1}{n}\sum_{j=1}^{n} x_{jk}$$

,

де k = 1; 2; ... m. Коваріація s_{ik} визначається виразом

$$s_{ik} = \frac{1}{n}\sum_{j=1}^{n}(x_{ji} - \overline{x}_i)(x_{jk} - \overline{x}_k)$$

,

де i = 1; … m. Потім ми визначаємо власні значення і власні вектори коваріаційної матриці S, яка є дійсною симетричною позитивною матрицею. Власне значення λ і ненульовий вектор e можуть бути знайдені так, що Se = λe, де e - власний вектор S.

Щоб знайти ненульовий вектор e, необхідно вирішити характеристичне рівняння $|S - \lambda I| = 0$. Якщо S - матриця повного рангу m x n, можна знайти m власних значень (1; 2; … m). Використовуючи $(S - \lambda I)e = 0$; всі відповідні власні вектори можуть бути знайдені. Власні значення і відповідні власні вектори будуть відсортовані так, щоб $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_m$: Матриця власних векторів була представлена як $e = [u_1, u_2, u_3, u_4 \ldots u_m]$: Діагональна матриця ненульових власних значень представлена як

$$\mathbf{\Lambda} = \begin{pmatrix} \lambda_1 & 0 & \cdots & 0 \\ 0 & \lambda_2 & 0 & 0 \\ \vdots & \vdots & \ddots & \cdots \\ 0 & 0 & 0 & \lambda_m \end{pmatrix}$$

Щоб отримати головні компоненти матриці S, ми виконаємо розкладання за власними значеннями, яке дається виразом

$$S = \mathbf{e\Lambda e^T}$$

Потім ми вибираємо перші власні вектори $d \leq m$, де d - бажане значення, наприклад, 100, 200, 400 і т.д. Набір головних компонентів представлений як $Y_1 = e_1^T x$, $Y_2 = e_2^T x$, …, $Y_d = e_d^T x$. Матриця M n x d представлена як

$$\mathbf{M} = \begin{pmatrix} f_{11} & f_{12} & \cdots & f_{1d} \\ f_{21} & f_{22} & \cdots & f_{2d} \\ f_{31} & f_{32} & \cdots & \cdots \\ f_{41} & f_{42} & \ddots & \cdots \\ f_{n1} & f_{n2} & \cdots & f_{nd} \end{pmatrix}$$

де f_{ij} - це зменшені вектори ознак від вихідного розміру даних m x m до розміру n x d.

3.3.3 Вибір функції за допомогою CPBF

Для вибору ознак із використанням підходу, заснованого на класі, ми вручну визначили найбільш регулярні слова, що існують у кожній категорії, та зважили їх за допомогою схеми зважування ентропії, перш ніж додавати їх до векторів ознак, які були обраний з PCA. Наприклад, слова, які регулярно існують у класі бейсболу, це "Ichiro", "baseball", "league", "baseman" тощо. Тоді в якості векторів ознак разом із зменшені основні компоненти від PCA. Потім ці вектори характеристик використовуються як вхідні дані до нейронних мереж для класифікації. Схема зважування ентропії для кожного доданка обчислюється як L_{jk} x G_k, де L_{jk} - локальне зважування терміна k, а G_k - загальне зважування терміна k. Ljk і Gk задані

$$L_{jk} = \begin{cases} 1 + \log TF_{jk} & (TF_{jk} > 0) \\ 0 & (TF_{jk} = 0) \end{cases}$$

та

$$G_k = \frac{1 + \sum_{k=1}^{n} \frac{TF_{jk}}{F_k} \log \frac{TF_{jk}}{F_k}}{\log n}$$

де n - кількість документів в колекції, а TF_{jk} - частота терміна для кожного слова в Doc_j, як згадувалося раніше. F_k - це частота терміна k в усій колекції документів. Ми вибрали R = 50 слів з найвищим значенням ентропії, які потрібно додати до перших d компонентів з PCA в якості вхідних даних для нейронних мереж при класифікації.

3.3.4 Введення даних до нейронних мереж

Після попередньої обробки веб-сторінок було створено словник, який містить усі унікальні слова в базі даних. Ми обмежили кількість унікальних слів у словниковому запасі до 1800, оскільки кількість різних слів велика. Кожне зі слів у словниковому запасі представляє один вектор ознак. Кожен вектор об’єкта містить вагу термінів документа. Висока розмірність векторів характеристик, які повинні бути вхідними даними до нейронних мереж, не є практичною через низьку масштабованість та продуктивність. Отже, PCA був використаний для зменшення

початкових векторів ознак m = 1800 на невелику кількість основних компонентів. У нашому випадку ми вибрали значення d = 400 разом із R = 50 функціями, вибраними з підходу CPBF, оскільки цей параметр працює ефективніше для класифікації веб-сторінок порівняно з іншими параметрами, що вводяться в нейронні мережі. Графік коефіцієнта навантаження для накопиченої частки власних значень наведено на рисунку 13.

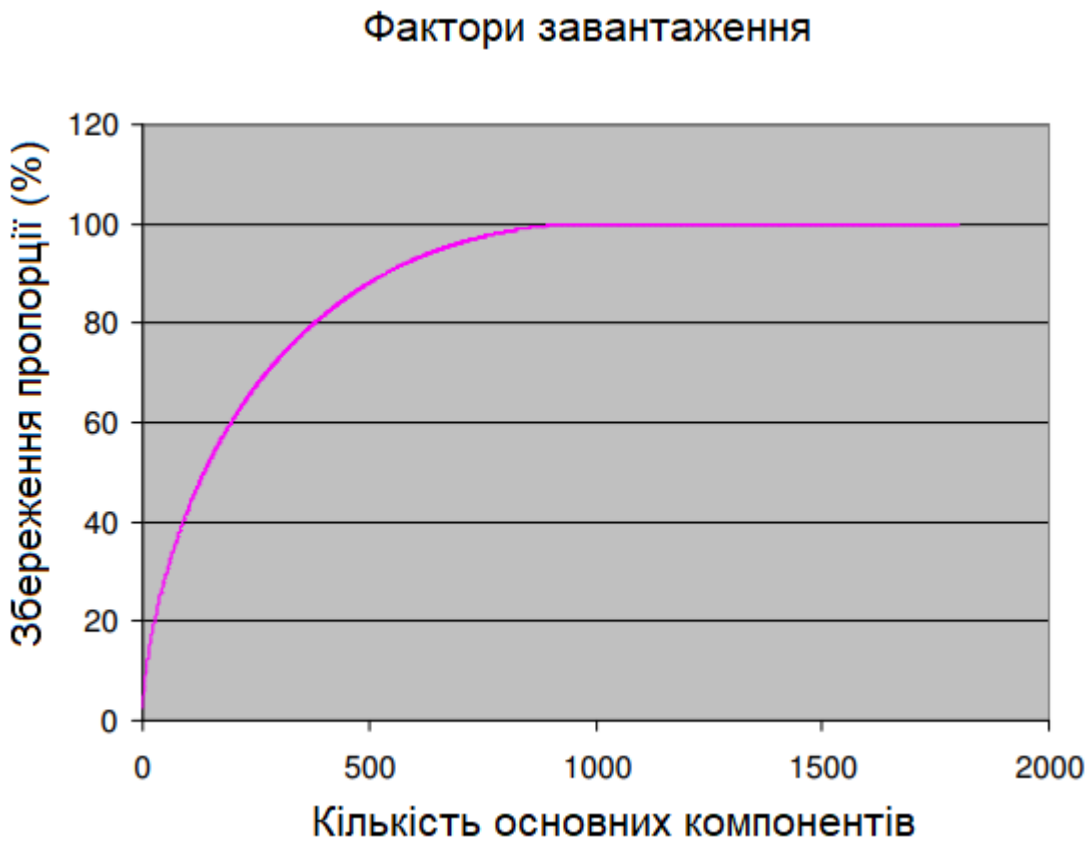

Рис. 3.6 - Накопичена частка основних компонентів, що генеруються PCA

3.4 Налаштування інструментів для нейронної мережі

В цьому підрозділі з’ясуємо методи та розглянемо варіанти навчання моделей аналізу даних, для класифікації контенту сторінок, беручи за основу бібліотеку машинного навчання scikit-learn.

Щоб почати роботу з бібліотекою scikit-learn треба завантажити інтерпретатор ддля мови програмування Python і додатково бібліотеки numpy та sciру. Щоб мати можливість працювати з базою даних необхідно встановити бібліотеку pandas. За допомогою бібліотеки NLTK можна виконувати роботу з текстовими даними та обробляти їх. Встановити вищезазначені бібліотеки можна за допомогою вбудованого до Python менеджера пакетів pip, що дозволяє не лише завантажувати та встановлювати необхідні додатки, а й проводити оновлення ПО та завантажувати необхідні залежності для бібліотек.

Для роботи з мовою Python було обрано середовище Jupiter Notebook. Інтерфейс представлений у вигляді вікна в будь-якому браузері, в якому можно створити та запустити проект на мові програмування Python. Вводити дані до проекту можна через спеціальну вбудовану консоль. Можно створити одразу декілька вікон або форм в одному вікні для відкриття та запуску декілької окремих програм, при чому кожна форма представлена незалежним блоком для запуску окремого скрипта. Дані в результаті обчислень будуть занесені до RAM, а потім можуть бути зчитані звідти іншими програмними блоками.

Серед баз даних була обрана база даних PostgreeSQL, з котрою просто та зручно працювати ха допомогою пакету psycopg2 для мови Python, котрий надає зручний та простий спосіб взаємодії з базами даних.

PostgreSQL також відома як Postgres - безкоштовна система керування базами даних (СКБД) з відкритим вихідним кодом, при цьому особлива увага приділяється розширюваності і сумісності з SQL.

PostgreSQL пропонує транзакції з властивостями атомарности, узгодженості, ізоляції, довговічності (ACID), автоматично оновлювані уявлення, матеріалізовані уявлення, тригери, зовнішні ключі і процедури. Вона розроблена для обробки ряду робочих навантажень, від окремих комп'ютерів до систем даних або веб-служб з безліччю одночасних користувачів.

Після установки пакетів можна почати установку з'єднання з базою даних, де config - це dict з налаштуваннями для під’єднання, що показано на рисунку 14.

```
conn = psycopg2.connect(host=config["hostname"],
                        user=config["username"],
                        password=config["password"],
                        database=config["database"])
```

Рис. 3.7 – Приклад коду з підключенням бази даних

Підключення показане на рисунку 12 надає змогу працювати з базою даних, користуючись курсором, що є незручним в рамках інтелектуального аналізу. В такому випадку ми можемо використати встановлений раніше пакет pandas, котрий створить об’єкт DataFrame, з даних, що отримає і над котрим можна виконувати різноманітні операції, наприклад обробка, вибірка або які-небудь обчислення.

3.5 Характеристика нейронної мережі

Архітектура нейронних мереж, які використовуються для процесу класифікації, показана на рисунку 15. Кількість вхідних шарів (p) дорівнює 450, де головні компоненти (d = 400) і R (= 50). Кількість прихованих шарів (q) дорівнює 25. Метод проб і помилок був використаний для пошуку потрібної кількості прихованих шарів, які забезпечують високу точність класифікації на основі вхідних даних в нейронні мережі. Кількість вихідних шарів (r) дорівнює 11, що базується на кількості класів в категорії.

Ми визначили t як номер ітерації, η це швидкість навчання, α це швидкість імпульсу, θ_q - зміщення на прихованому модулі q, θ_r - зміщення на одиницю виведення

r, δ_q - це узагальнена помилка на рівні q, а δ_r - узагальнена помилка між шарами q і r . Рівні введення нейронної мережі представлені як f_1, f_2, … , f_{450}. Адаптація ваг між прихованим (q) і вхідним (p) шарами визначається виразом

$$W_{qp}(t+1) = W_{qp}(t) + \Delta W_{qp}(t+1)$$

где

$$\Delta W_{qp}(t+1) = \eta \delta_q O_p + \alpha \Delta W_{qp}(t)$$

$$\delta_q = O_q(1 - O_q) \sum_r \delta_r W_{rq}.$$

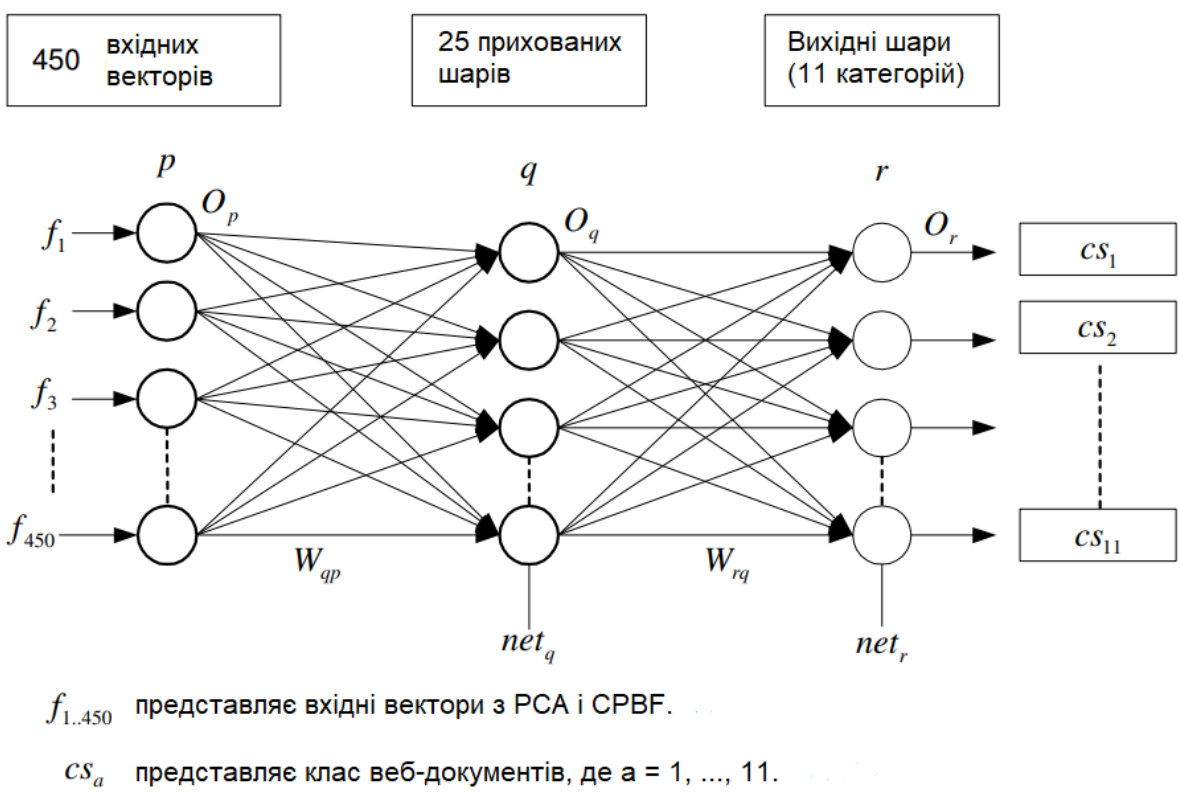

Рис. 3.8 - Вектори ознак з PCA і CPBF передаються в нейронні мережі для класифікації.

Розглядати приклад класифікації ми будемо для сайтів новин зі спорту. Саме тому тут ми можемо виділити 11 основних класифікацій, що показано в таблиці 3.1, хоча в цілому систему можна перебудувати під інші тематики з різною кількістю вхідних та вихідних параметрів.

Таблиця 3.1 - Кількість документів, збережених у базі даних новин

Клас (cs)	Тренувальні документи (Т)	Тестові документи
baseball	100	101
boxing	70	12
cycling	84	40
football	50	10
golf	100	120
motor-sports	50	11
hockey	70	20
rugby	50	13
skiing	70	26
swimming	70	12
tennis	70	17
Total	784	382

Зверніть увагу, що перша функція, що передається на прихованому шарі (q) визначається виразом:

$$net_q = \sum_q W_{qp} O_p + \theta_q$$

$$O_q = f(net_q) = 1/(1 + e^{-net_q})$$

Адаптація ваг між виходом (r) і прихованими (q) шарами виражається:

$$W_{rq}(t+1) = W_{rq}(t) + \Delta W_{rq}(t+1)$$

де

$$\Delta W_{rq}(t+1) = \eta \delta_r O_q + \alpha \Delta W_{rq}(t)$$

$$\delta_r = O_r(1 - O_r)(t_r - O_r)$$

Тоді вихідна функція на вихідному шарі (k) визначається виразом:

$$net_r = \sum_r W_{rq} O_q + \theta_r$$

$$O_r = f(net_r) = 1/(1 + e^{-net_r})$$

Параметри нейронних мереж зі зворотним поширенням помилок наведені в таблиці 3.2 .

Таблиця 3.2 - Параметри нейронних мереж зворотного поширення помилки для двох тестових запусків.

Параметри нейронної мережі	Запуск 1	Запуск 2
Швидкість навчання (η)	0.05	0.005

Продовження таблиці 3.2

Параметри нейронної мережі	Запуск 1	Запуск 2
Швидкість (α)	0.01	0.001
Кількість ітерацій (t)	1200	1000
Середня квадратична похибка (MSE)	0.05	0.005

Для експериментів і класів, як показано в Таблиці 3.2. 784 документа відносяться до 1 і більше класів. Щоб перейти до групи документів для навчання, ми випадковим чином вибрали 200 документів в якості позитивного навчального набору. Для негативних прикладів ще 200 документів вибираються випадковим чином з іншого набору документів, який не належить набору 11 класів. Загальна кількість навчальних документів - 400, в тому числі з негативними прикладами і позитивними прикладами.

4 ЕКСПЕРИМЕНТИ

Ми використовували той же тест, який включає байєсовські класифікатори та класифікатори TF-IDF. Ми також включаємо підхід WPCM, як порівняння з методами, які використовуються для перевірки застосовності системи класифікації. Опис WPCM було згадано в Розділі 3. TF-IDF і байєсовські методи тестів описані нижче.

4.1 TF-IDF measures

Класифікатор TF-IDF заснований на відповідному алгоритмі зворотного зв'язку Роккі з використанням моделі векторного простору. Алгоритм представляє документи у вигляді векторів, так що документи зі схожим змістом мають схожі вектори. Кожен компонент вектора відповідає терміну в документі, зазвичай слову.

Алгоритм Роккі заснований на методі зворотного зв'язку за релевантністю, виявлення в інформаційно-пошукових системах, який виник на основі системи пошуку інформації SMART, розробленої в 1960-1964 рр. Як і багато інших пошукових системи, підхід зі зворотним зв'язком Роккі був розроблений з використанням моделі векторного простору. Алгоритм заснований на припущенні, що більшість користувачів мають загальне уявлення про те, які документи слід позначати як релевантні або нерелевантні. Отже, пошуковий запит користувача змінений, щоб включити довільний відсоток релевантних і нерелевантних документів як засіб підвищення чуйності пошукової машини і, можливо, також точності.

Вага кожного компонента обчислюється з використанням схеми частотного зважування документа, зворотної частоти терміна (TF-IDF), яка намагається винагороджувати слова, які зустрічаються багато разів, але в декількох документах. Щоб класифікувати новий документ Doc, косинуси векторів-прототипів з відповідними векторами документа обчислюються для кожного класу. Doc присвоюється класу, вектор документа якого має найвищий косинус.

4.2 Класифікатор Баєса

Для статистичної класифікації ми використовували стандартний байесовский класифікатор. Використовуючи байесовский класифікатор, ми припустили, що поява терміна не залежить від інших термінів. Ми хочемо знайти клас cs, який дає найвищу умовну ймовірність для документа Doc. Нехай w_k^m = {w_1, w_2, … , w_m} - це слова, що представляють текстовий зміст документа Doc, і нехай k позначає номер терміна, де k = 1, … , m. Оцінка за класифікацію вимірюється

$$P(cs) = \prod_{k=1}^{m} P(w_k|cs)$$

де P (cs) - апріорна ймовірність класу cs, а P (w_k|cs) - ймовірність того, що слово w_k з класу cs оцінюється в маркірованому навчальному документі. Потім мережа класифікує документ сторінки в класі, який максимізує оцінку класифікації. Якщо оцінки для всіх класів в спортивній категорії менше заданого порогового значення, то документ вважається некласифікованим. Інструментарій Rainbow використовувався для класифікації навчальних і тестових новинних веб-сторінок з використанням методів Байеса і TF-IDF.

В Rainbow Framework організовані різні методи і процеси, які можна використовувати для моніторингу та оцінки. Діапазон завдань згрупований в сім кольорових кластерів: Manage, Define, Frame, Describe, Understand Causes, Synthesise, та Report & Support Use. Користувачі можуть використовувати платформу для планування оцінки, що охоплює всі необхідні завдання, або вибрати підхід, що містить заздалегідь підготовлену комбінацію варіантів завдань.

4.3 Результати експерименту

Результати окремих симуляцій з використанням байєсівського, TF-IDF і WPCM показані в таблиці 4.1. Точності з використанням байесовских методів, TF-IDF і

WPCM складають 81,00%, 83,94% і 84,10% відповідно. Також ми виявили, що якщо кількість навчальних і тестових документів невелика, тобто 10-80 сторінок, точність класифікації становить менше 85%. Точність класифікації уроків гольфу, регбі, лижного спорту, плавання і тенісу з використанням підходу WPCM краще в порівнянні з підходами Байеса і TF-IDF, які складають 95.90%, 75.26%, 89.90%, 85.90%, 96.02%. Відповідно як показано в таблиці 4.1.

Таблиця 4.1 - Відсоток точності класифікації стрічках веб-сторінок

Класи Docs	Баєс (%)	TF-IDF (%)	WPCM (%)
baseball	99.01	96.04	96.04
boxing	86.80	65.80	74.20
cycling	80.00	68.00	74.20
football	71.40	85.30	76.00
golf	96.67	95.00	95.90
hockey	80.21	85.50	78.86
motor-sports	85.65	89.90	82.60
rugby	63.56	72.60	75.26
skiing	63.60	84.08	89.90
swimming	75.21	88.00	85.90
tennis	82.21	94.30	96.02
Total	81.00	83.94	84.10

Процес вибору характеристик CPBF, застосовуваний в підході WPCM, заснований на найвищій ентропії ключових слів. Вибираються перші 50 ключових слів з найвищими значеннями ентропії і комбінуються з функціями, взятими з підходу PCA. Придатність вибору ключових слів, що належать до конкретних класів, була ретельно розглянута в підході WPCM при проведенні експериментів, описаних в Розділі 4. Наприклад, обрані ентропії для ключових слів 'famous', 'field', 'skills' і 'manager' з класу football, хоча ці ключові слова також існують в інших класах, таких як баскетбол і бейсбол. Однак однієї ідентифікації ключових слів недостатньо. Пов'язані ваги, які стосуються кожного ключового слова, також важливі для підвищення ефективності класифікації з використанням запропонованого підходу, оскільки одне і те ж слово може зустрічатися в різних класах. Це основна причина, по якій точність класифікації класів boxing і cycling (86,80% і 80,00%) краще при використанні байєсівського підходу в порівнянні з підходом WPCM, як показано в таблиці IV. Крім того, процеси зупинки і зупинки також знижують ефективність класифікації з використанням підходу WPCM в порівнянні з Байєсова підходом. Крім того, точність класифікації футбольних і хокейних класів (85,30% і 85,50%) краще при використанні підходу TF-IDF в порівнянні з підходом WPCM, як показано в таблиці 4.1. Це пов'язано з тим, що вміст навчальних документів відноситься до класів футболу, а хокей містить багато розріджених ключових слів. Для вибору документів-кандидатів з кожного класу необхідно використовувати більш досконалий підхід до вибору документів, щоб підвищити результати класифікації за допомогою підходу WPCM. Результати експериментів для параметрів нейронних мереж зі зворотним поширенням помилок, як в Таблиці 3.2, показані на Рис. 16 і 17. Для швидкості імпульсу $\alpha = 0{,}01$ ми виявили, що локальні мінімуми існують. Але якщо використовувати значення $\alpha = 0{,}001$, ми виявили, що MSE гладка. Для інших експериментів ми використовували параметр $\alpha = 0{,}001$, оскільки він вказує на стабільне значення MSE, яке буде використовуватися для нашого процесу класифікації.

У статистиці середньоквадратична помилка (MSE) або середньоквадратичне відхилення (MSD) оцінювача (процедури оцінки ненаблюдаемой величини) вимірює середнє значення квадратів помилок, тобто середньоквадратичне різницю між оціненими значеннями і фактичними значеннями. MSE - це функція ризику, відповідна очікуваному значенню квадрата втрати помилок. Той факт, що MSE майже завжди строго позитивна (а не нульова), відбувається через випадковість або через те, що оцінювач не враховує інформацію, яка могла б дати більш точну оцінку. MSE - це міра якості оцінки - вона завжди невід'ємна, а значення, близькі до нуля, краще.

MSE - це другий момент (при виникненні) помилки і, таким чином, включає в себе як дисперсію оцінки (наскільки широкий розкид оцінок від однієї вибірки даних до іншої), так і її зміщення (наскільки далеко середнє оцінене значення від істинного значення). Для незміщеної оцінки MSE - це дисперсія оцінки. Як і дисперсія, MSE має ті ж одиниці вимірювання, що і квадрат оцінюваної величини. За аналогією зі стандартним відхиленням витяг квадратного кореня з MSE дає середньоквадратичнепомилку або середньоквадратичне відхилення (RMSE або RMSD), яке має ті ж одиниці, що і оцінювана величина; для незміщеної оцінки RMSE це квадратний корінь з дисперсії, відомий як стандартна помилка.

Епоха - це термін, який використовується в машинному навчанні, який вказує кількість проходів за весь навчальний набір даних, виконаних алгоритмом машинного навчання. Набори даних зазвичай групуються в пакети (особливо, коли обсяг даних дуже великий). Деякі люди використовують термін ітерація в широкому сенсі і відносяться до проходження однієї партії через модель як до ітерації.

Якщо розмір пакета - це весь навчальний набір даних, то кількість епох - це кількість ітерацій. З практичних причин це звичайно не так. Багато моделей створені з більш ніж однієї епохою.

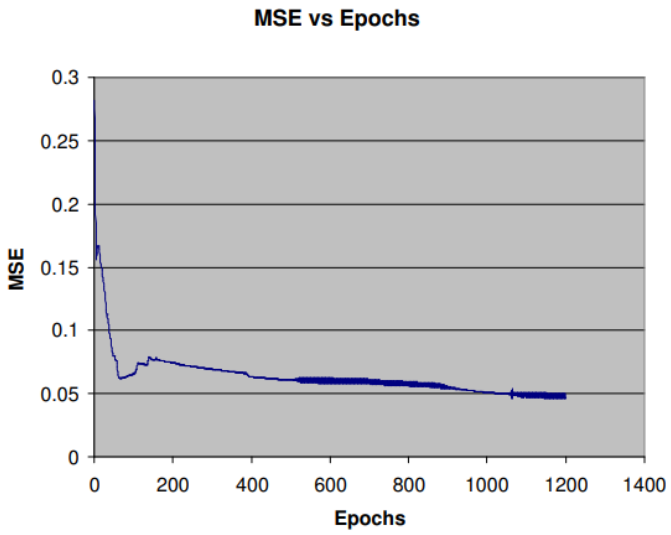

Рис. 4.1 - Результати першого запуску для MSE та Epochs для класифікації веб-сторінок з використанням нейронних мереж (швидкість імпульсу = 0.01)

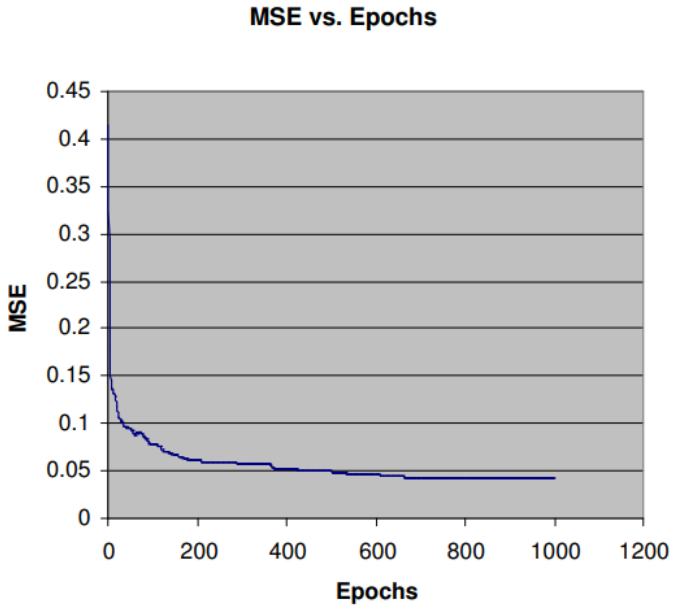

Рис. 4.2 - Результати другого запуску для MSE vs. Epochs для класифікації веб-сторінок з використанням нейронних мереж (швидкість імпульсу = 0.001).

ВИСНОВКИ

Метою магістерської роботи стало дослідження існуючих та створення методу для аналізу контенту та класифікації веб-сторінок по підгрупам обраної тематики (як спорт, фінанси або ін.) за допомогою нейронних мереж, з урахуванням підвищення точності або швидкості порівняно з аналогами.

В ході роботи було проаналізовано методи класифікації сайтів, котрі існують, що дозволило зробити висновок, що дані методи не можуть задовольнити сучасним умовам точності в різноманітті тематик класифікації.

Після проведення досліджень було з’ясовано, що ефективнішим буде метод на основі CPBF та PCA, де CPBF – це підхід функцій на основі профілю класу, а PCA - неконтрольований непараметрический статистичний метод, який використовується для зменшення розмірності. Слід взяти до уваги, що дуже важливим на практиці є обрання якісної вибірки даних, що не міститиме невірно відмічених або й взагалі порожніх веб-сторінок. Цей висновок було зроблено після того, як моделі були навчені на попередньо фільтрованій вибірці з вірно передбаченими даними.

На основі всіх виконаних досліджень і вивченої літератури була розроблена модель для аналізу контенту сайтів та їх класифікації за певною тематикою з оцінкою точності не менше 85%, низькою квадратичною похибкою (менше 0.05) та універсальною формою роботи з різними темами класифікацій.

Серед проблем можна виділити, що найскладніше класифікувати сайти, котрі містять багато медіаконтенту та мало текстової інформації. Таких сторінок в мережі досить багато, оскільки люди краще сприймають інформацію завдяки медіаконтенту, як зображення або відео. Систему в подальшому можна розширити додатковою нейромережею для розпізнання такого типу контенту, що покращить якість аналізу та класифікації. Тому одним з вірогідних напрямків подальшого розвитку є використання методів розпізнання медіаконтенту.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. E. Riloff, W. Lehnert, Information extraction as a basis for highprecision text classification, ACM Transactions on Information Systems 12 (3) (1994) 296–333.
2. S. Brin, L. Page, The anatomy of a large-scale hypertextual web search engine, Proceedings of the 7th International World Wide Web Conference, Brisbane, Australia, Apr 1998
3. В. Д. Чабаненко. Модифікації методу стохастичного градієнтного спуску для задач машинного навчання з великими обсягами даних. Master's thesis, Московський державний університет імені М.В. Ломоносова, 2016.
4. M. Chau, H. Chen, Personalized and focused Web spiders, in: N. Zhong, J. Liu, Y. Yao (Eds.), Web intelligence, Springer–Verlag, 2003b, pp. 197–217.
5. J. Turian et al. Word representations: A simple and general method for semi-supervised learning. Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, c. 384-394, 2010.
6. A. McCallum, K. Nigam, J. Rennie, K. Seymore, A machine learning approach to building domain-specific search engines, Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI-99), 1999, pp. 662–667.
7. Методы оптимизации нейронных сетей [Електронний ресурс] / Павел Садовников. Adam, 2017 — Режим доступу: https://habr.com/ru/post/318970/ — Дата доступу: жовтень 2020.
8. J. Cho, H. Garcia-Molina, L. Page, Efficient crawling through URL ordering, Proceedings of the 7th International World Wide Web Conference, Brisbane, Australia, Apr 1998.
9. Hochreiter S., Bengio Y., Frasconi P., Schmidhuber J. Gradient flow in recurrent nets: the difficulty of learning long-term dependencies. [Electronic resource] / A

Field Guide to Dynamical Recurrent Neural Networks. IEEE press, 2001 — Mode of access: ftp://ftp.idsia.ch/pub/juergen/gradientflow.pdf — Last access: 2020.

10. Машинне навчання [Електронний ресурс] – Режим доступу: http://www.machinelearning.ru/wiki/index.php?title=Машинное_обучение – Дата доступу: жовтень 2020.

11. Michael Chau, Hsinchun Chen, A machine learning approach to web page filtering using content and structure analysis [Electronic resourse] / ScienceDirect, 2007 – Mode of access: https://www.sciencedirect.com/science/article/abs/pii/S0167923607000875 - Last access: 2020.

12. S. Chakrabarti, B. Dom, P. Indyk, Enhanced hypertext categorization using hyperlink, Proceedings of ACM SIGMOD International Conference on Management of Data, Seattle, Washington, USA, Jun 1998.
13. T. Joachims, Text categorization with support vector machines: learning with many relevant features, Proceedings of the European Conference on Machine Learning, Berlin, 1998, pp. 137–142.
14. Assessment of extended aggregated association rules Sitnikov, D., Ryabov, O., Titova, O., Kovalenko, A.Proceedings of 2018 IEEE 9th International Conference on Dependable Systems, Services and Technologies, DESSERT 2018, 2018, pp. 93–97