Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)
Кафедра комп'ютерних систем та мереж
(повна назва кафедри)

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня

бакалавр
(назва освітнього ступеня)

на тему: *Комп'ютеризована система тематичної рубрикації документів*

Виконав: студент *IV* курсу, групи *СІс-44*
спеціальності *123 «Комп'ютерна інженерія»*
(шифр і назва спеціальності)

(підпис) *Григораш В.С.* (прізвище та ініціали)

Керівник (підпис) *Луцків А.М.* (прізвище та ініціали)

Нормоконтроль (підпис) *Луцик Н.С.* (прізвище та ініціали)

Завідувач кафедри (підпис) *Осухівська Г.М.* (прізвище та ініціали)

Рецензент (підпис) *Млинко Б.Б.* (прізвище та ініціали)

Тернопіль
2021

Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет *комп'ютерно-інформаційних систем і програмної інженерії*
(повна назва факультету)

Кафедра *комп'ютерних систем та мереж*
(повна назва кафедри)

ЗАТВЕРДЖУЮ
Завідувач кафедри
Осухівська Г.М.
(підпис) (прізвище та ініціали)
« » 2021 р.

З А В Д А Н Н Я

НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття освітнього ступеня *бакалавр*
(назва освітнього ступеня)

за спеціальністю *123 «Комп'ютерна інженерія»*
(шифр і назва спеціальності)

студенту *Григорашу Вадиму Святославовичу*
(прізвище, ім'я, по батькові)

1. Тема роботи *Комп'ютеризована система тематичної рубрикації документів*

Керівник роботи *Луцків Андрій Мирославович, к.т.н., доцент*
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Затверджені наказом ректора від «*10*» *лютого* 2021 року № *4.7-97*

2. Термін подання студентом завершеної роботи *25.06.2021 р.*

3. Вихідні дані до роботи *Методи машинного аналізу текстів, відкриті бібліотеки для опрацювання природної мови, методи класифікації текстових документів, принципи мови програмування Python*

4. Зміст роботи (перелік питань, які потрібно розробити)
Вступ. 1. Аналіз технічного завдання і сфери застосування комп'ютеризованої системи тематичної рубрикації документів 2. Проектування структури і компонентів комп'ютеризованої системи тематичної рубрикації документів. 3. Програмна реалізація комп'ютеризованої системи тематичної рубрикації документів. 4. Безпека життєдіяльності, основи охорони праці. Висновки

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, слайдів)

1. *Кластеризація, тематичне моделювання і класифікація текстової інформації.*
2. *Архітектура комп'ютеризованої системи тематичної рубрикації документів.*
3. *Схема препроцесингу текстових документів*
4. *Архітектура комп'ютеризованої системи тематичної рубрикації документів*
5. *Алгоритм тематичної рубрикації документів.*
6. *Результати тематичної рубрикації текстових документів*

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Безпека життєдіяльності,	*Пилипець М.І., д.т.н., проф. каф. МТ*		
основи охорони праці			

7. Дата видачі завдання _____

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1	*Розробка та аналіз технічного завдання*	*10.02-19.02.2021*	
2	*Аналіз моделей і сфер застосування тематичного моделювання*	*19.02-05.03.2021*	
3	*Проектування структури і компонентів комп'ютеризованої системи тематичної рубрикації документів*	*05.03-26.03.2021*	
4	*Методи та інструменти реалізації систем тематичної рубрикації документів*	*26.03-01.04.2021*	
5	*Програмна реалізація комп'ютеризованої системи тематичної рубрикації документів*	*01.04-22.04.2021*	
6	*Розробка інструкцій із встановлення та налаштування параметрів комп'ютеризованої системи*	*22.04-10.05.2021*	
7	*Безпека життєдіяльності, основи охорони праці*	*10.05-18.05.2021*	
8	*Оформлення кваліфікаційної роботи*	*18.05-06.06.2021*	
9	*Попередній захист кваліфікаційної роботи*	*06.06-18.06.2021*	
10	*Захист кваліфікаційної роботи*	*22.06-27.06.2021*	

Студент _____ *Григораш Вадим Святославович*
(підпис) (прізвище та ініціали)

Керівник роботи _____ *Луцків Андрій Мирославович*
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Комп’ютеризована система тематичної рубрикації документів // Кваліфікаційна робота на здобуття освітнього ступеня бакалавр // Григораш Вадим Святославович // ТНТУ, спеціальність 123 «Комп’ютерна інженерія»// Тернопіль, 2021 // с.– 74, рис. – 25 , табл. – 3, аркушів А1 – 6, бібліогр. – 22.

Ключові слова: система, текст, тема, документ, класифікація, рубрикація.

У кваліфікаційній роботі спроектовано та реалізовано програмний прототип комп’ютеризованої системи тематичної рубрикації текстових документів. Під документом, у даному випадку, може виступати звичний для пересічного користувача текстовий файл, або, наприклад, повідомлення із соціальних мереж чи лист з електронної пошти.

До складу системи входять сховище документів та компонент, що відповідає з рубрикацію документів на основі аналізу їх вмісту.

Компонент рубрикації документів складається з наступних модулів: модуль попереднього опрацювання тексту; модуль виявлення ознак тексту; модуль класифікації документів.

В якості методів для виявлення ознак тексту у документі запропоновано використати різновиди статистичних ознак алгоритму TF-IDF, а також семантична векторизації.

Експериментальні дослідження щодо рубрикації документів виконано на основі моделей Баєсівського класифікатора, методу опорних векторів, логістичної регресії і методів, що базуються на використанні простих і глибоких нейронних мереж. Результати, одержані у процесі експерименту на наборі неструктурованих даних без виконання попереднього опрацювання тексту показали, що найбільшу точність можна досягти при використанні логістичної регресії, яка становить трохи більше 70%.

ABSTRACT

Computer-aided system of documents topical classification // Bachelor's thesis // Hryhorash Vadym Sviatoslavovych// TNTU, speciality 123 «Computer engineering»// Ternopil, 2021 // p.– 74 , fig. – 25 , tab. – 3, posters A1 – 6, ref. – 22.

Keywords: system, text, topic, document, classification, rubrication.

In the qualification work the program prototype of the computerized system of thematic rubrication of text documents is designed and realized. The document, in this case, may be a text file familiar to the average user, or, for example, a message from social networks or an e-mail.

The system includes a document repository and a component that corresponds to the rubrication of documents based on the analysis of their content. The component of rubrication of documents consists of the following modules: the module of preliminary processing of the text; text feature detection module; document classification module.

As methods for detecting the features of the text in the document, it is proposed to use a variety of statistical features of the TF-IDF algorithm, as well as semantic vectorization. Experimental studies on the rubrication of documents were performed on the basis of Bayesian classifier models, the method of reference vectors, logistic regression and methods based on the use of simple and deep neural networks.

The results obtained during the experiment on a set of unstructured data without pre-processing the text showed that the highest accuracy can be achieved using logistic regression, which is slightly more than 70%.

ЗМІСТ

ПЕРЕЛІК ОСНОВНИХ УМОВНИХ ПОЗНАЧЕНЬ,
СИМВОЛІВ І СКОРОЧЕНЬ

БД	База даних
БЗ	База знань
КС	Комп’ютеризована система
ПЗ	Програмне забезпечення
НМ	Нейронна мережа
CNN	Convolutional Neural Networks
GRU	Gated Recurrent Unit
LSTM	Long Short Term Models
LDA	Lantent Dirichlet Allocation
NLP	Natural Language Processing
NLG	Natural Language Generation
RNN	Recurrent Neural Networks
RCNN	Recurrent Convolutional Neural Networks

ВСТУП

У результаті еволюції інформаційних технологій на сьогодні накопичилась величезна кількість даних різної природи, які потребують автоматизованого опрацювання. Враховуючи, що сховища даних дозволяють зберігати такі об’єми інформації і на сьогодні спостерігається значний розвиток теоретичних і прикладних аспектів алгоритмів машинного навчання, то побудова комп’ютеризованих інтелектуальних систем є актуальною задачею на даному етапі розвитку інформаційних технологій.

Важливими напрямами розвитку інтелектуальних комп’ютеризованих систем є сфери, де необхідно проводити прогнозування фізичних величин у часі, автоматичну класифікація і кластеризацію об’єктів різної природи, створення «розумних систем», а також наближення машинного інтелекту та біологічного інтелекту людини.

У кваліфікаційній роботі необхідно організувати комп’ютеризовану систему тематичної рубрикації текстових документів із застосуванням сучасних досягнень у галузях комп’ютерної і програмної інженерії, а також галузі інтелектуального опрацювання даних. Рубрикація документів відноситься до задач класифікації та опрацювання природної мови, тому доцільно використовувати моделі, методи та інструментальні засоби NLP.

При проектуванні комп’ютеризованої системи необхідно визначити вимоги до апаратного забезпечення, побудувати архітектуру системи, а також реалізувати програмну інтелектуальну її складову. Початковим етапом створення системи тематичної рубрикації є формування технічного завдання, яке деталізує вимоги до системи, а також проведення аналітичного огляду наукових і практичних напрацювань цієї сфери. Згідно з обмеженнями, які накладаються на програмну реалізацію системи повинні бути використанні відкриті бібліотеки, що підтримуються мовою програмування Python. Сферою застосування комп’ютеризованої системи тематичної рубрикації документів є автоматична класифікація листі електронної пошти, архіви документів та ін.

РОЗДІЛ 1 АНАЛІЗ ТЕХНІЧНОГО ЗАВДАННЯ І СФЕРИ ЗАСТОСУВАННЯ КОМП'ЮТЕРИЗОВАНОЇ СИСТЕМИ ТЕМАТИЧНОЇ РУБРИКАЦІЇ ДОКУМЕНТІВ

1.1 Аналіз вимог технічного завдання при проектуванні комп'ютеризованої системи тематичної рубрикації документів

Основне призначення комп'ютеризованої системи тематичної рубрикації документів полягає в автоматизації процесу визначення вмісту документу і класифікації або кластеризації його у відповідності до теми. Дана система може бути різновидом або частиною більш складних комп'ютерно-інформаційних комплексів з опрацювання текстової інформації.

Застосування комп'ютеризованих систем такого класу стосуються побудови тематичних моделей для сфери добування тексту та пошуку інформації. Проектовану комп'ютеризовану систему можна використовувати при вирішенні задач класифікації, категоризації, узагальнення та сегментації документів.

Комп'ютеризована система тематичної рубрикації документів може також бути ефективним інструментом у галузі комп'ютерного зору, популяційної генетики та соціальних мереж. При пошуку інформації, тематичне моделювання допомагає розширювати критерії запитів, а також персоналізувати результати пошуку або побудови рекомендацій шляхом відображення тих тем і тих документів, які відповідають уподобанням користувачів.

У соціальних науках застосування комп'ютеризованої системи рубрикації документів дає змогу якісно аналізувати настрій користувачів, його емоції та

будувати психологічний портрет людини на основі приналежності документів до певної рубрики.

Окрім цього, комп'ютеризована система рубрикації документів може використовуватись у сфері програмної і комп'ютерної інженерії, що дозволить на основі моделювання тем проводити аналіз вихідного коду, змін журналів, стану баз даних та ін.

Метою побудови комп'ютеризованої системи тематичної рубрикації документів є створення автоматизованого інтелектуального засобу для класифікації текстових документів, які дозволять формувати релевантні та ранжовані списки у відповідності до запиту користувача. Проектована комп'ютеризована система повинна забезпечити ефективність процесу пошуку і рубрикації документів за рахунок зменшення часових затрат, у випадку виконання таких дій людиною.

Мета роботи щодо створення комп'ютеризованої системи рубрикації документів передбачає виконання і розв'язок ряду задач, основними з яких є:

– дослідження сучасних методів та інструментів опрацювання природної мови автоматизованими засобами;

– обґрунтування і побудова моделі розпізнавання контексту у текстових документах;

– реалізація моделі аналізу текстових даних;

– реалізація інтелектуалізованих алгоритмів класифікації і рубрикації документів;

– обґрунтоване застосування метрик для вимірювання подібності документів між собою та приналежності до певної рубрики;

– проведення експериментальних досліджень щодо рубрикації документів;

– скорочення часових ресурсів формування рубрик і категорій документів.

Основними задачами комп'ютеризованої системи тематичної рубрикації документів є:

– опрацювання корпусу текстових документів засобами автоматизованого аналізу;

– препроцесинг текстової інформації;

– виявлення інформативних ознак документів для встановлення їх приналежності до певної рубрики або категорії;

– можливість навчання і проведення процедур тестування тематичної рубрикації документів;

– класифікація та рубрикація документів;

– ранжування документів у межах заданих рубрик;

– формування адекватних відповідей у вигляді ранжованого списку документів на запит користувача, що стосується певної тематики корпусу документів;

– формування анотацій документів;

– можливість формування ключових слів за документами;

– формування додаткових рекомендацій документів відносно запиту користувача;

– забезпечення точності рубрикації документів з точністю не нижче за 70%;

– формування кількісних критеріїв щодо якості та ефективності тематичної рубрикації корпусу документів;

– можливість формування розподілу за рубриками у наявному корпусі документів.

До найбільш важливих функцій комп’ютеризованої системи тематичної рубрикації документів належить автоматизація процесу класифікації документів на основі аналізу їх вмісту із застосуванням алгоритмів машинного навчання.

Загальні вимоги до комп’ютеризованої системи тематичної рубрикації документів пов’язані із поставленими у кваліфікаційній роботі задачами і передбачають виконання усіх етапів, характерних для опрацювання природної мови.

Комп’ютеризована система в цілому повинна розв’язувати задачі класифікації документів за наперед визначеними категоріями, або як альтернативну виконувати їхню кластеризацію. Окрім цього, важливою вимогою до системи є формування анотацій документів у зрозумілій для людини формі на основі виявлених ключових слів або фраз.

Рубрикація за темами документів повинна відбуватись на відкритому наборі даних, який може бути попередньо розмічений. Розбиття на навчальну і тестові вибірки повинні бути або рівномірними, або складати максимальне співвідношення 80:20.

Архітектурна композиція комп’ютеризованої системи автоматичної тематичної рубрикації текстових документів повинна передбачати використання технології клієнт-сервер, як на рівні апаратного забезпечення, так і на рівні програмного. Це забезпечить комунікацію між програмним забезпечення користувача, яка в даному випадку виступає в якості терміналу, та сервером, що зберігає корпус документів.

При програмній реалізації системи автоматичної тематичної рубрикації доцільно використовувати середовища з підтримкою мов програмування у галузі інтелектуального аналізу даних, зокрема Python.

В цілому, набір основних вимог до комп’ютеризованої системи можна сформулювати наступним чином:

– можливість зчитування тексту з корпусу документів;

– забезпечення можливості векторного представлення документу;

– можливість застосування метрик для виявлення подібності між документами;

– здатність аналізувати контекст документу та визначати ключові слова;

– можливість визначення теми документу на основі навчальної вибірки;

– здатність одночасного віднесення документу до кілько категорій;

– здатність комп’ютеризованої системи взаємодіяти з іншими системами;

– забезпечення точності рубрикації документів на рівні не менше, ніж 70%.

Вимогами щодо функціонування комп’ютеризованої системи тематичної рубрикації документів, з врахуванням особливостей структурних компонентів, є забезпечення зв’язку прикладного додатку з джерелом даних, що містить корпус документів, які необхідно прокласифікувати, або вказати приналежність до певної наперед визначеної категорії.

Кожен документ повинен бути доступним для зчитування його вмісту, а також мати унікальний ідентифікатор та володіти інформативним метаописом. В метаописі можлива наявність інформації про кількість слів, розмір документу, автора, дати створення та інших, що є важливими з позиції його комплексного опису.

Працездатність комп’ютеризованої системи повинна забезпечуватись двома важливими функціональними блоками:

– сховище корпусу документів даних, яка може бути сформована у вигляді база даних текстової інформації;

– інтелектуальна компонента, що безпосередньо виконує тематичну рубрикацію документів.

Інтелектуальну складову комп’ютеризованої системи рекомендовано реалізувати на основі таких відкритих бібліотек:

– Scikit-learn – бібліотека з відкритим кодом, що володіє простими та ефективними інструментами для прогнозування і аналізу даних різної природи D;

– TextBlob – це бібліотека для обробки текстових даних, що забезпечує простий API для опрацювання природної мови, зокрема визначення частини мови, виділення іменникових фраз, аналіз настроїв, класифікації, перекладу тощо.

– Pandas – швидкий, потужний, гнучкий та простий у використанні інструмент аналізу та маніпулювання даними з відкритим кодом, який побудований поверх мови програмування Python;

– Keras – бібліотека, що пропонує послідовні та прості API, мінімізує кількість дій користувача, необхідних для загальних випадків використання, а також забезпечує чіткі повідомлення про помилки;

– XGBoost – це оптимізована, розподілена градієнтно-підсилювальна бібліотека, розроблена для забезпечення високої ефективності, гнучкості та портативності та реалізації алгоритмів машинного навчання в рамках Gradient Boosting.

Система тематичної рубрикації документів має показувати стійкі результати щодо визначення приналежності документу до певної категорії, а також формувати анотацію з використанням ключових слів.

Способи і засоби зв’язку між компонентами комп’ютеризованої системи тематичної рубрикації документів можна поділити на два типи: локальне навчання моделі тематичного моделювання та віддалене тестування працездатності системи.

При локальному тематичному моделюванні використовується частина корпусу текстових документів для проведення експериментальних досліджень щодо вибору оптимальної моделі рубрикації документів.

У випадку віддаленого тестування та експлуатації комп’ютеризованої системи корпус документів може бути розміщений на серверах мережі Інтернет або у хмарних сховищах. Протокол передачі та обміну даними, який при цьому використовується – HTTP/HTTPS.

Особливих вимог щодо проведення діагностики комп’ютеризованої системи тематичної рубрикації документів не висувається, однак вона повинна бути виконана у відповідності до затвердженого розкладу. Крім того, має проводитись також регулярна перевірка наповнення сховища даних текстовими документами, фіксації зміни їх локації, а також тестування зв’язку між клієнтом і сервером.

При виникненні збоїв, у випадку віддаленого зв’язку між інтелектуальним сервісом і сховищем документів, необхідно провести діагностику каналів передачі даних та усунути проблему.

До перспектив розвитку комп'ютеризованої системи тематичної рубрикації документів належить можливість її адаптації та інтеграції із пошуковими інтелектуальними системами, системами управління архівами документів у різних сферах, зокрема електронних бібліотек та ряду інших. Окрім цього, до системи автоматизованої рубрикації можна додавати інтерфейси користувача в залежності від його потреб, тобто сама система повинна реалізовувати так звану back end логіку.

Шляхами модернізації проектованої системи є можливість внесення змін для підвищення продуктивності виконання операцій щодо класифікації і кластеризації текстових документів, формування контейнерів із відповідними залежностями між бібліотеками у вигляді контейнерів, що забезпечить її кросплатформність та стійкість функціонування у середовищі кінцевого користувача.

Комп'ютеризована система тематичної рубрикації документів повинна відповідати вимогам надійності, які висуваються до такого класу систем, володіти інструментами авторизації користувачів на рівні програмного забезпечення клієнта і сервера, на якому розміщено сховище документів, а також характеризуватися здатністю до відновлюваності при виникнення аварійних ситуацій.

Надійність комп'ютеризованої системи тематичної рубрикації повинна відповідати наступним критеріям:

– сталість функціонування та формування результатів класифікації документів протягом визначеного періоду часу, зазвичай 8-12 год./день (період робочого дня підприємства);

– робастність алгоритмів машинного навчання при рубрикації документів;

– відповідність вимогам часу напрацювання на відмови, тобто час безперервного функціонування не менше, ніж 100000 год.;

– можливість швидкого усунення збоїв у роботи комп'ютеризованої системи рубрикації документів;

– наявність механізмів забезпечення захисту даних на програмному та апаратного рівні;

– наявність інструментів управління та контролю за виконанням запитів користувачів;

– здатність підтримувати стабільність зв’язку із зовнішніми системами.

Вимоги і задачі відносно проектування функцій комп’ютеризованої системи тематичної рубрикації документів стосуються :

– можливості аналізу вмісту документа та проведення токенізації тексту;

– здатності виявлення і видалення шумів з документу;

– можливість проведення лематизації і стеммінгу;

– здатності проведення процедур стандартизації текстових даних;

– можливість виявлення сутностей у документів;

– здатності формувати контекст документу;

– можливості формування статистичних показників документу;

– здатності визначення інформативних ознак тексту у корпусі документу;

– можливості класифікації документів із визначеною точністю і достовірністю приналежності до визначених категорій.

При моделюванні та експлуатації комп’ютеризованої системи тематичної рубрикації документів до апаратного забезпечення висуваються такі рекомендовані вимоги:

– процесор – Intel Core i5 4300M з частотою 2,2 ГГц або 2,3 ГГц (1 сокет, 8 ядер, 2 потоки на ядро);

– об’єм оперативної пам’яті – 16 ГБ.

Альтернативними конфігураціями при реалізації рекомендаційної системи є:

– Xeon E5-2498 v3 з частотою 1,8 ГГц (1 сокет, 10 ядер кожен, 2 потік на ядро, або Xeon Phi 7210 з частотою 1,3 ГГц (1 сокет, 64 ядра, 4 потоки на ядро);

– 32 ГБ або 64 ГБ об’єм оперативної пам’яті;

– накопичувач (жорсткий диск) з наявністю не менше, ніж 2-3 ГБ простору.

Операційні системи для ефективного роботи системи тематичної рубрикації можуть бути будь-якого типу (Windows, Linux, MacOS) однак повинні підтримувати мову програмування Python відповідної версії, а також перелік бібліотек, які описані у пункті структури та функціонування комп’ютеризованої системи.

1.2 Аналіз моделей і сфер застосування тематичного моделювання та рубрикації документів

Із все більшим зростанням і накопиченням різного роду інформації дедалі складнішим для людини стає їхнє опрацювання. Більшість інформації зберігається у вигляді текстів, висловлених природною для людини мовою, тому задачі перевірки електронної пошти, розуміння газет за десять попередніх років або характеристика наукових досліджень вимагають вже залучення засобів їхньої автоматизації, зокрема для виявлення короткого змісту повідомлень або суті великих документів.

Тому тематичне моделювання, що представляє собою статистичну базу даних і допомагає користувачам зрозуміти великі колекції документів: не просто знайти окремі документи, а й зрозуміти загальні теми, присутні в колекції, є на сьогодні дуже перспективним напрямом, що інтенсивно розвивається.

Тематичне моделювання – це потужний напрямок досліджень в області автоматичного опрацювання текстів. Тематична модель колекції текстових документів визначає, до яких тем відноситься кожен документ і з яких слів складається кожна тема.

На відміну від звичайних методів кластеризації, тематична модель може відносити документ не до одного кластеру-теми, а до декількох, тобто використовується «м'яка кластеризацію», причому не тільки документів, але і

слів. Зазвичай тематичні моделі відносять до методів машинного навчання «без вчителя», оскільки вони не вимагають розмічених навчальних вибірок. Це дозволяє використовувати тематичне моделювання в тих випадках, коли немає жодних додаткових даних, крім власне текстової колекції. Наприклад, тематичне моделювання застосовується для інформаційного пошуку у великих текстових масивах, для аналізу спеціалізованих текстів чи текстів на рідкісних мовах, для аналізу великих масивів тексто подібних даних, таких як програмний код, тексти пісень, банківські транзакції, географічні дані, музичні товари.

Результатом імовірнісного тематичного моделювання є кінцева множина тем, кожна з яких описується імовірнісним розподілом на множині слів. Важливою властивістю теми є її інтерпретованість. Слова, які мають високу ймовірність у даній темі, повинні належати до однієї предметної області і бути семантично пов'язаними. Тема вважається інтерпретованою, якщо, розглядаючи найбільш частотні слова, експерт може сказати, про що йдеться у документі, і дати їй певну назву [1].Якщо усі теми (або майже всІ) інтерпретуються, то про таку модель кажуть, що вона в цілому є інтерпретованою. В такому випадку модель може бути корисна для розуміння тематичної структури колекції. Інтепретованість є важко формалізованою характеристикою. Існують різні експертні та обчислювальні методики її кількісного оцінювання [2]. Розвиток імовірнісного тематичного моделювання почався з роботи Т.Хофманна [3], в якій була запропонована модель імовірнісного латентного семантичного аналізу («Probabilistic Latent Semantic Analysis, PLSA»). Побудова тематичної моделі є некоректно поставленим завданням стохастичного матричного розкладання, яка має безліч рішень. Для довизначення постановки задачі і вибору найбільш підходящого рішення необхідно вводити додаткові обмеження на модель. Наступною важливою моделлю тематичного моделювання стала модель латентного розміщення Діріхле («Latent Dirichlet Allocation, LDA») [4], заснована на Байєсівській регуляризації шуканих дискретних розподілів за допомогою апріорних розподілів Діріхле. У наступні роки на основі PLSA і LDA

були розроблені сотні спеціалізованих моделей, що відрізняються способами регуляризації, структурою вихідних даних матричної декомпозиції [5-7].

Класифікація тексту відноситься до класу задач «навчання із вчителем» та полягає у тренуванні класифікатора на основі навчальних даних. Моделювання тем (тематичне моделювання) належить до задач «навчання без вчителя», де теми документів або тексті наперед невідомі. У цьому випадку теми формуються на основі фактичних даних.

Кластеризація текстових даних і тематичне моделювання схожі в тому сенсі, що обидві задачі відносяться до задач «навчання без вчителя» (рис. 1.1).

Рисунок 1.1 – Кластеризація і тематичне моделювання

Як кластеризація так і тематичне моделювання намагаються упорядкувати документи для кращого пошуку та перегляду інформації. Однак між ними є різниця.

Кластеризація тексту розглядає схожість між документами та спроби сформувати подібні кластери цих документів. Ці заходи подібності можуть базуватися на статистичній оцінці TF-IDF. У моделюванні тем не розв’язується задача визначення подібності документів. Натомість розглядається документ як суміш тем, у яких тема є розподілом ймовірності слів. М'яка кластеризація (де документ може належати до кількох кластерів) може розглядатися як подібна до моделювання тем, хоча підходи все одно відрізняються. Таким чином, кластери тексту не зовсім однакові з темами при їх моделюванні.

У кваліфікаційній роботі необхідно розробити комп’ютеризовану систему тематичної рубрикації документів, що швидше відповідає задачі класифікації текстових документів, як показано на рис. 1.2.

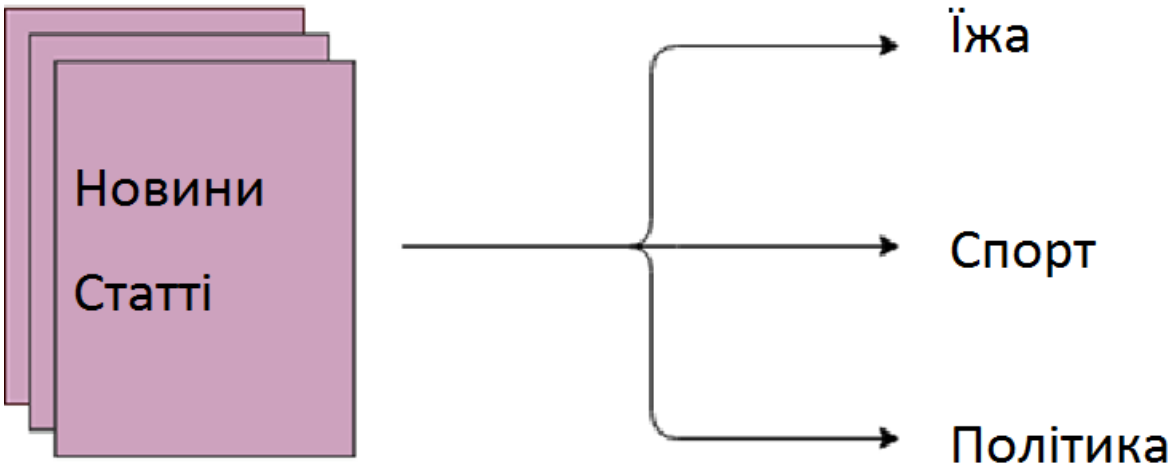


Рисунок 1.2 – Задача тематичної рубрикації документів

У даному випаду рубрикація документів є прикладом контрольованого машинного навчання, оскільки маркований набір даних, що містить текстові документи та їх мітки, використовується для підготовки класифікатора.

Наскрізний конвеєр класифікації тексту складається з трьох основних компонентів:

1. Підготовка набору даних: Першим кроком є етап підготовки набору даних, який включає процес його завантаження та виконання базової

попередньої обробки. Потім набір даних поділяється на навчальну вибірку і тестову.

2. Інженерія ознак («Features engineering»). Наступним кроком є виявлення ознак тексту при якому необроблений набір даних трансформується у такі ознаки, які можна використовувати у моделях машинного навчання. Цей крок також включає процес створення нових ознак на основі наявних даних.

3. Навчання моделі. Заключним кроком є етап побудови моделі при якому модель машинного навчання навчається на маркованому наборі даних.

4. Поліпшення ефективності текстового класифікатора. Передбачається налаштування гіперпараметрів та оцінювання моделей прогнозування приналежності документу до певної категорії.

РОЗДІЛ 2 ПРОЕКТУВАННЯ СТРУКТУРИ І КОМПОНЕНТІВ КОМП'ЮТЕРИЗОВАНОЇ СИСТЕМИ ТЕМАТИЧНОЇ РУБРИКАЦІЇ ДОКУМЕНТІВ

2.1 Побудова структури компонентів комп'ютеризованої системи тематичної рубрикації документів

Визначення компонентів комп'ютеризованої системи тематичної рубрикації документів та взаємозв'язків між ними формують процес побудови та відображення її архітектури і дозволяють забезпечити досягнення поставлених у роботі цілей.

Проводячи аналіз предметної області виявлено, що основними компонентами системи є інформація щодо документів, їхній опис і власне програмна реалізація модуля класифікації текстових документів.

Як було зазначено у попередньому розділі, задача тематичної рубрикації документів полягає в тому, щоб на основі навчальної вибірки документів, які містять маркери категорій (класів), забезпечити навчання системи таким чином, що при появі нового документу, інтелектуальний сервіс автоматично формував мітку цього документу.

Враховуючи вище викладений матеріал, запропоновано архітектуру комп'ютеризованої системи тематичної рубрикації документів, яка представлена на рис. 2.1. До її складу входять:

– сховище документів;
– підсистема аналізу текстової інформації;
– підсистема виявлення ознак документу;
– підсистема класифікації документу.

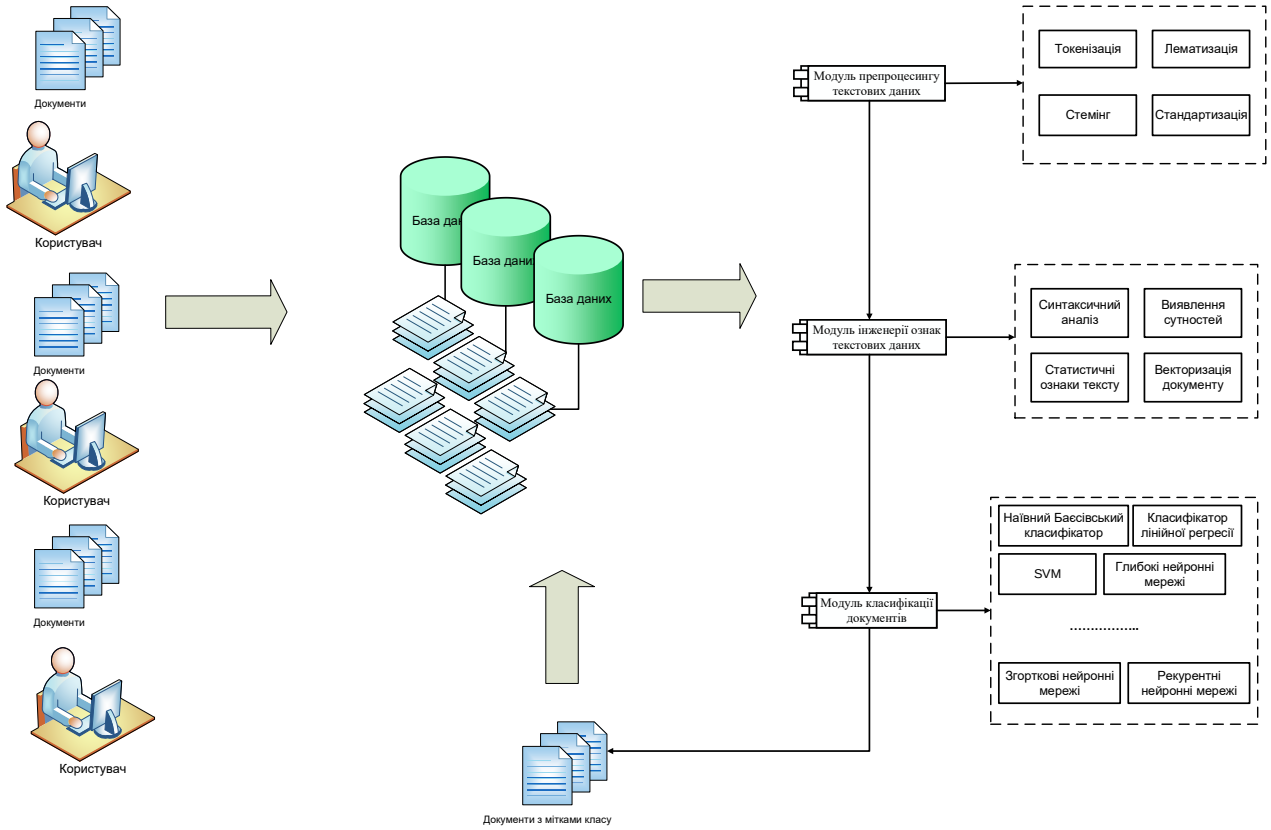


Рисунок 2.1 – Компоненти і зв’язки між ними комп’ютеризованої системи тематичної рубрикації текстових документів

Процес функціонування комп’ютеризованої системи тематичної рубрикації документів передбачає:

– наповнення бази даних документів із мітками категорій, до яких вони належать;

– зберігання та опрацювання документів;

– аналіз вмісту текстового документу;

– повернення мітки класу до якого належить документ.

З рис. 2.1 видно, шо після того, як документ потрапив у базу даних, він передається на опрацювання модулем препроцесингу тексту і над ним повинні виконуватись такі дії як:

– токенізація;

– лематизація;

– стемінг;

– стандартизація.

Після того, як виконано препроцесинг текстових даних, керування комп’ютеризованої системи передається до модуля інженерії ознак тексту. Даний модуль реалізує наступні функції:

– синтаксичний аналіз тексту;

– виявлення сутностей;

– формування статистичних ознак документу;

– створення векторного представлення текстового документу.

Маючи важливі ознаки тексту в контексті їхньої приналежності до певних категорій, завершальним кроком є безпосередня класифікація документів, або по іншому формування мітки документу. Класифікація текстових документів може здійснювались на основі наступних моделей та алгоритмів:

– наївний Баєсівкий класифікатор;

– класифікатор на основі лінійної регресії;

– класифікатор на базі багатошарового прецептрону;

– модель глибоких нейронних мереж;

– класифікатор на основі опорних векторів («SVM»);

– класифікатор на основі підходу Boosting.

Варто відміти, що до класу глибоких нейронних мереж належать:

– модель згорткових нейронних мереж з класифікатором (CNN);

– модель рекурентних нейронних мереж («RNN»);

– моделі з корокто- довго- тривалою пам’яттю («Long Short Term Models LSTM»);

– моделі нейронних мереж з блоками «Gated Recurrent Unit (GRU)»;

– двонаправлені рекурентні нейронні мережі «Bidirectional RNN»;

– рекупентні згорткові нейронні мережі «Recurrent Convolutional Neural Network (RCNN)»;

– інші різновиди нейронних мереж з великою кількістю прихованих шарів та складної архітектури.

2.2 Методи та інструменти препроцесингу текстових даних

Оскільки текст є найбільш неструктурованою формою з усіх наявних даних, тому у ньому присутні різні типи шумів, і дані неможливо аналізувати без попереднього опрацювання. Весь процес очищення і стандартизації тексту, що покликаний прибрати шуми і забезпечити здатність до аналізу прийнято називати препроцесингом. Зазвичай, препроцесинг передбачає виконання трьох основних етапів:

– виявлення і видалення шумів у тексті
– нормалізація текстового набору
– стандартизація тексту.

На рис. 2.2 показано архітектуру конвеєру попередньої обробки тексту.

Рисунок 2.2 – Схема препроцесингу текстових даних

На рис. 2.2 вхідними даними є неструктурований текст, що міститься у документах. Після цього потрібно виконати утилізацію шумів, що передбачає видалення стоп слів, не важливих знаків пунктуації, URL-адрес та посилань на

літературні джерела, якщо такі наявні у документі. По-іншому до шуму належить будь-який фрагмент тексту, який не має відношення до контексту даних та кінцевого результату. До стоп-слів в англійській мові належать загальновживані слова по типу «am, the, of, in», URL-адреси або посилання, сутності соціальних мереж (згадки, хештеги), знаки пунктуації та галузеві слова. Цей крок стосується видалення всіх типів непотрібних сутностей, які присутні у тексті. Загальним підходом до видалення шуму є підготовка словника сутностей та ітерація текстового об'єкта за допомогою лексем (або за допомогою слів), усуваючи ті лексеми, які присутні у довіднику шумів.

Приклад реалізації видалення шумів мовою Python наведено у лістингу 2.1.

Лістинг 2.1 – Видалення шумів з тексту

```
# Приклад видалення шумових слів з тексту

noise_list = ["is", "a", "this", "..."]
def _remove_noise(input_text):
    words = input_text.split()
    noise_free_words = [word for word in words if word not in
noise_list]
    noise_free_text = " ".join(noise_free_words)
    return noise_free_text

_remove_noise("this is a sample text")
>>> "sample text"
```

Після видалення шумів з документу виконуються наступні кроки щодо токенізації, лематизації і стемінгу текстових даних. Токенізація представляє обою процес перетворення тексту на лексеми. Лексеми – слова або сутності, які наявні у тексті. Текстовий об'єкт – речення або фраза, слово або стаття. Один з принціипів організації проуесу токенізації показано на рис. 2.3.

Інший тип текстового шуму стосується багатьох представлень, що описують одне і те ж слово. Наприклад, "грати", "гравець", "грали", "гравці" і

"гравцями" – це різні варіанти слова – "грати", хоча вони пишуться по різному, але вживаються у спільному контексті. Етап нормалізації перетворює всі відмінки слова або його види у нормалізовану форму (також відому як лема). Нормалізація є ключовим кроком при препроцесингу текстових даних, оскільки вона виконує перехід елементів з більшого простору даних (N різних елементів) у простір з низькими розмірностями (1 елемент), що є ідеальним варіантом для будь-якої моделі машинного навчання. Найпоширенішими практиками лексичної нормалізації є: стемінг і лематизація

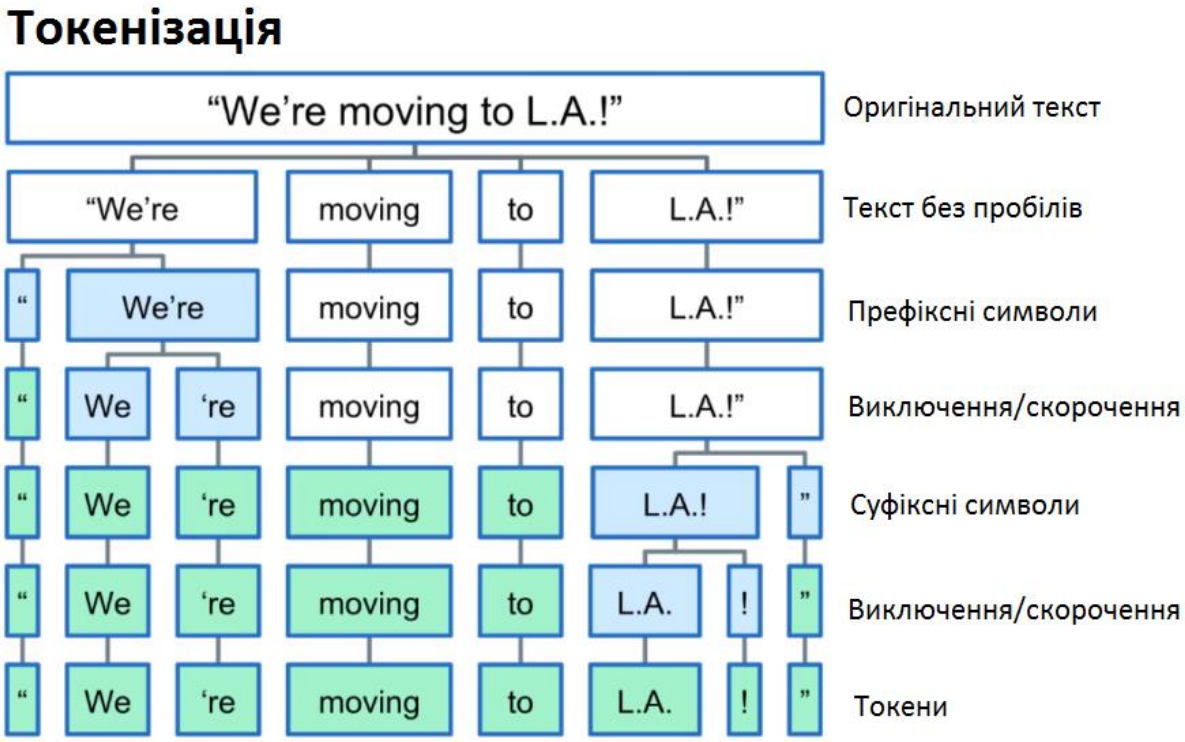


Рисунок 2.3 – Алгоритм токенізації тексту

Стемінг – це елементарний процес, заснований на правилах очищення суфіксів („ing“, „ly“, „es“, „s“ тощо) із слова.

Лемматизація, з іншого боку, є організованою і поетапною процедурою отримання кореневої форми слова, вона використовує лексику (значення словника) та морфологічний аналіз (структура слова та граматичні залежності). На рис. 2.4 показано приклад лематизації і стемінгу, а методи реалізації стемінгу показано на рис. 2.5.

Препроцесинг даних, реалізований на основі токенізації, лематизації і стемінгу забезпечує більш структуроване представлення текстового документу, що дозволяє вже виконувати певні операції на реченнями, однак для забезпечення повноти попереднього опрацювання природної мови потрібно провести ще стандартизацію документу.

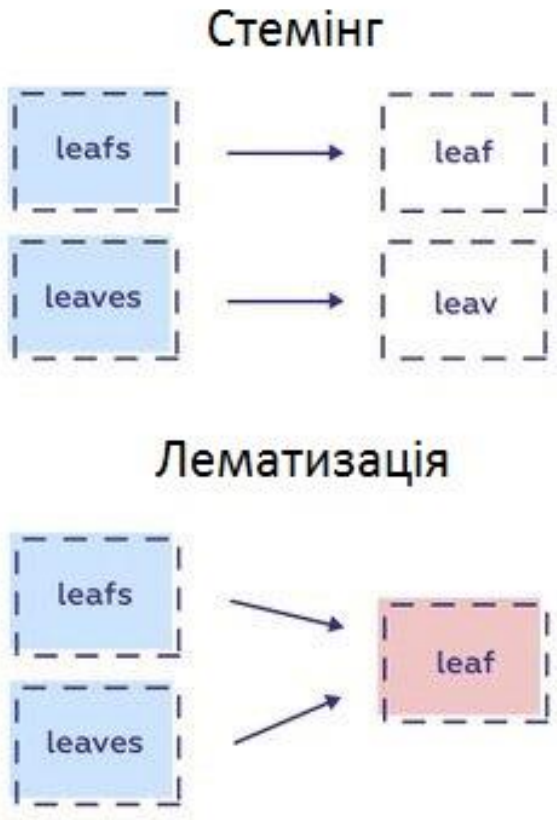


Рисунок 2.4 – Приклад стемінгу і лематизації

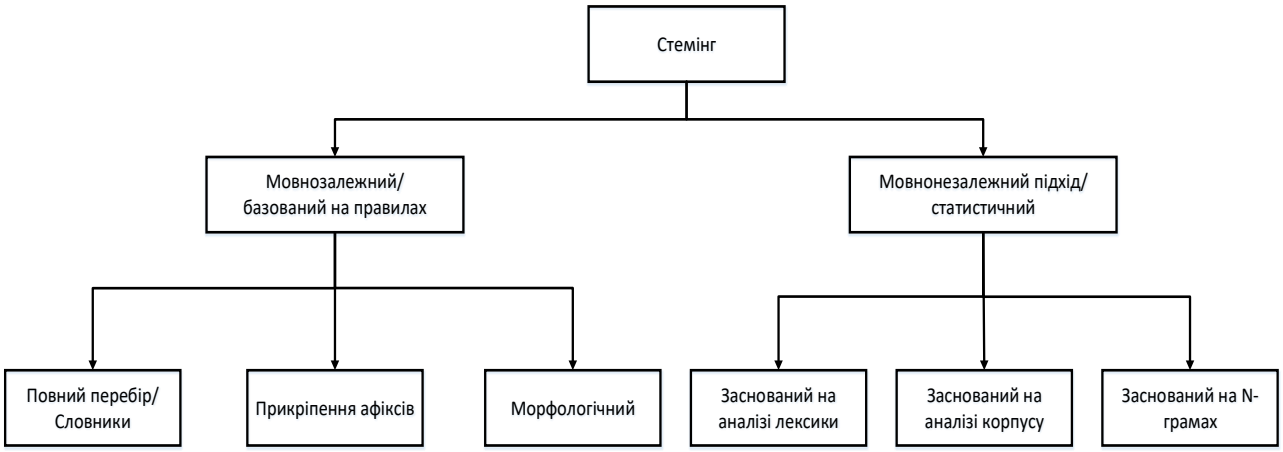


Рисунок 2.5 – Методи реалізації процедури стемінгу

Приклад програмного коду, які реалізує лематизацію і стемінг на основі бібліотеки NLTK показано у лістингу 2.2.

Лістинг 2.2 – Приклад реалізації процедури лематизації і стемінгу

```
from nltk.stem.wordnet import WordNetLemmatizer
lem = WordNetLemmatizer()

from nltk.stem.porter import PorterStemmer
stem = PorterStemmer()

word = "multiplying"
lem.lemmatize(word, "v")
>> "multiply"
stem.stem(word)
>> "multipli"
```

Текстові дані часто містять слова або фрази, яких немає в жодному стандартному лексичному словнику. Ці частини не визнаються пошуковими системами та моделями. До таких слів і фраз належать абревіатури, хештеги з прикріпленими словами та розмовні сленги. За допомогою регулярних виразів та підготовлених вручну словників даних такий тип шуму можна виправити. Приклад реалізації пошуку слів на основі словника для заміни сленгу соціальних медіа з тексту наведено у лістингу 2.3.

Лістинг 2.3 – Стандартизація сленгових виразів

```
lookup_dict = {'rt':'Retweet', 'dm':'direct message', "awsm" :
"awesome", "luv" :"love", "..."}
def _lookup_words(input_text):
    words = input_text.split()
    new_words = []
    for word in words:
        if word.lower() in lookup_dict:
```

```
            word = lookup_dict[word.lower()]
        new_words.append(word) new_text = " ".join(new_words)
        return new_text

_lookup_words("RT this is a retweeted tweet by Shivam Bansal")
>> "Retweet this is a retweeted tweet by Shivam Bansal"
```

Таким чином, проведено аналіз методів попереднього опрацювання текстової інформації, визначено алгоритми їх функціонування, що в подальшому буде використано при реалізації комп'ютеризованої системи тематичної рубрикації документів.

2.3 Методи інженерії ознак при опрацюванні текстової інформації

Для того, щоб проводити аналіз попередньо опрацьованих текстів їх потрібно перетворити на ознаки. Залежно від використання, ознаки тексту можуть бути визначені на основі використання різних методів:

- синтаксичний аналіз;
- на основі аналізу слів, які визначають сутності або N-грами;
- статистичних функцій та векторизації документу («word embedding»).

2.3.1 Синтаксичний аналіз слів і речень

Синтаксичний розбір включає в себе аналіз слів у реченні щодо граматики та їх розташування таким чином, щоб показати взаємозв'язок між словами. Граматика залежності і теги частин мови («Part Of Speech») є важливими атрибутами тексту.

Дерева залежності – речення складаються з декількох пов'язаних слів. Зв'язок між словами у реченні визначається граматикою базової залежності. Граматика залежностей – це клас синтаксичного аналізу тексту, який має справу з (позначеними) асиметричними двійковими відношеннями між двома лексичними елементами (словами).

Кожне відношення може бути представлене у формі триплету (відношення, головний, залежний). Для прикладу, «Bills on ports and immigration were submitted by Senator Brownback, Republican of Kansas». Зв'язок між словами можна спостерігати у вигляді дерева, як показано на рис. 2.6.

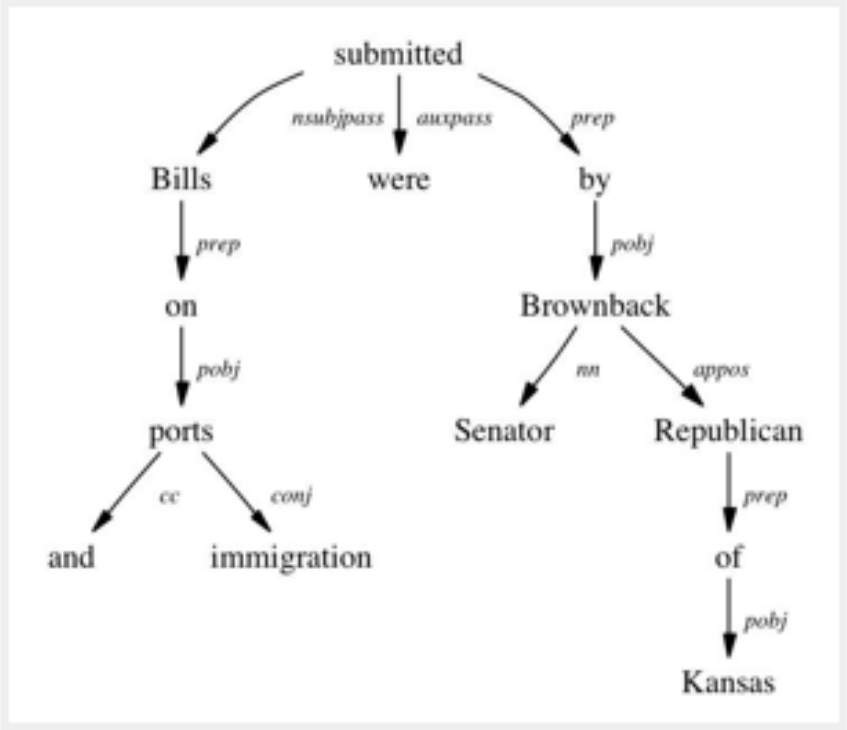


Рисунок 2.6 – Дерево залежності між словами у реченні

Дерево показує, що “ submitted ” є головним словом цього речення і пов’язаний двома піддеревами (субдерева суб’єкта та об’єкта). Кожне піддерево є також деревом залежностей із залежностями, такими як:

– «“Bills” <-> “ports” <by> “proposition”»;
– «“ports” <-> “immigration” <by> “conjugation”».

Цей тип дерева при рекурсивному аналізі зверху вниз дає в якості вихідних даних триплети відношення граматик, які можуть бути використані як ознакт для багатьох задач nlp, таких як аналіз настроїв, ідентифікація акторів та сутностей, класифікація тексту. Для створення дерев залежностей можна використовувати StanfordCoreNLP та граматики залежностей NLTK.

Окрім граматичних залежностей, кожне слово у реченні також асоціюється з тегом, який вказує на частину мови («pos») (іменники, дієслова, прикметники, прислівники тощо). Теги «pos» визначають використання та функції слова у реченні. Список усіх можливих pos-тегів, визначених Пенсильванським університетом можна знайти за посиланням: «https://www.sketchengine.eu/penn-treebank-tagset/». На рис. 2.7 показано приклад визначення частин мови.

Рисунок 2.7 – Приклад результату визначення частин мови у тексті

Тегування частин мови використовується для багатьох важливих цілей при опрацюванні природної мови, зокрема, при:

– аналізі омонімічних слів;
– покращенні процесу виявлення ознак у тексті;
– лематизації і нормалізації;
– ефективному видалення стоп-слів.

Деякі слова у мові можуть мати кілька значень у відповідності до контексту їхнього використання. Наприклад, у двох реченнях нижче:

– «Please book my flight for Delhi»;
– «I am going to read this book in the flight».

У цих реченнях слово «book» використовується у різному контексті, однак тег частини мови у них відрізняється. У першому реченні слово "book" вживається як дієслово, тоді як у дркгому – воно вживається як іменник.

Модель навчання може визначати різні контексти слова, коли воно використовується як ознака, однак якщо тег частини мови пов’язаний з ними, контекст зберігається, таким чином підвищується ефективність розпізнавання контексту слів у реченні.

POS-теги є основою процесу лематизації для приведення слова до нормальної форми (лема), а також вони корисні при видаленні стоп-слів. Наприклад, є деякі теги, які завжди визначають низьку частоту/менш важливі слова мови. Наприклад: (IN - "всередині", "після", "крім"), (CD - "один", "два", "сотня"), (MD - "може", "must" тощо).

2.3.2 Розпізнавання сутностей і тематичне моделювання

Сутності визначаються як найважливіші фрагменти речення – іменні фрази, дієслівні фрази або їх комплекс. Алгоритми виявлення та розпізнавання сутності – це, як правило, ансамблеві моделі синтаксичного аналізу на основі асоціативних правил, пошуку у словнику, позначення позицій та аналізу залежностей. Сферою застосування алгоритмів розпізнавання сутностей є автоматизовані чат-боти, аналізатори вмісту документів та емоцій користувачів.

Приклад виявлення і розпізнавання сутностей приведено на рис. 2.8.

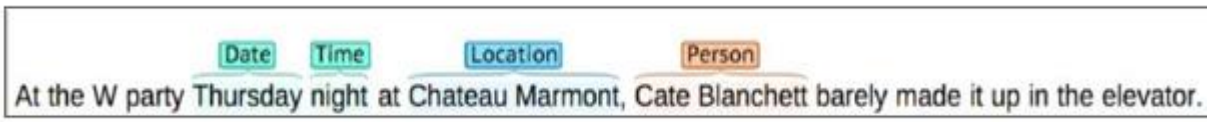


Рисунок 2.8 – Приклад виявлення та розпізнавання сутностей

Моделювання тем та розпізнавання іменованих сутностей - два ключові методи виявлення сутності при опрацюванні природної мови.

Процес виявлення з тексту таких сутностей, як імена осіб, назви локацій, компаній тощо, називають розпізнаванням іменованих сутностей «Name Entity Recognition (NER)». Наприклад: «Григораш Вадим, студент Тернопільського національного технічного університету, зараз перебуває у Львові». У даному випадку має наступні іменовані сутності:

– «особа»: «Григораш Вадим»;

– «організація»: «Тернопільський національний технічний уніерситет»;

– «місцезнаходження»: «Тернопіль».

Типова модель NER складається з трьох блоків: ідентифікація менованих сутностей, їх класифікація та усунення неоднозначності.

Ідентифікація іменованих фраз передбачає добування всіх іменникових фраз із тексту за допомогою синтаксичного аналізу залежностей та позначення частини мови.

Класифікація фраз є етапом, на якому всі добуті іменні фрази класифікуються за відповідними категоріями (місцезнаходження, імена тощо). API Google карт надає хороший механізм для розрізнення місць розташування. Окрім цього, відкриті бази даних з dbpedia, wikipedia можна використовувати для ідентифікації імен осіб або назв компаній. При класифікації можна керувати пошуковими таблицями і словниками, комбінуючи інформацію з різних джерел.

Іноді можлива така ситуація, що сутності неправильно класифіковані, отже, корисно створити рівень перевірки поверх результатів. Для цього можна використати графи знань. До популярних графів знань належать: «Google Knowledge Graph», «IBM Watson» та «Wikipedia».

Тематичне моделювання – це процес автоматичного визначення тем, які присутні у текстовому корпусі, що враховує приховані закономірності серед слів у корпусі без вчителя. Теми визначаються як «повторюваний патерн термінів, що повторюються у корпусі». Хороша тематична модель дає результат - „здоров’я”, „лікар”, „пацієнт”, „лікарня” для теми «Охорона здоров’я» та „ферма”, „врожаї”, „пшениця” для теми «Сільське господарство». Прихований розподіл Діріхле (LDA) – найпопулярніша техніка моделювання тем.

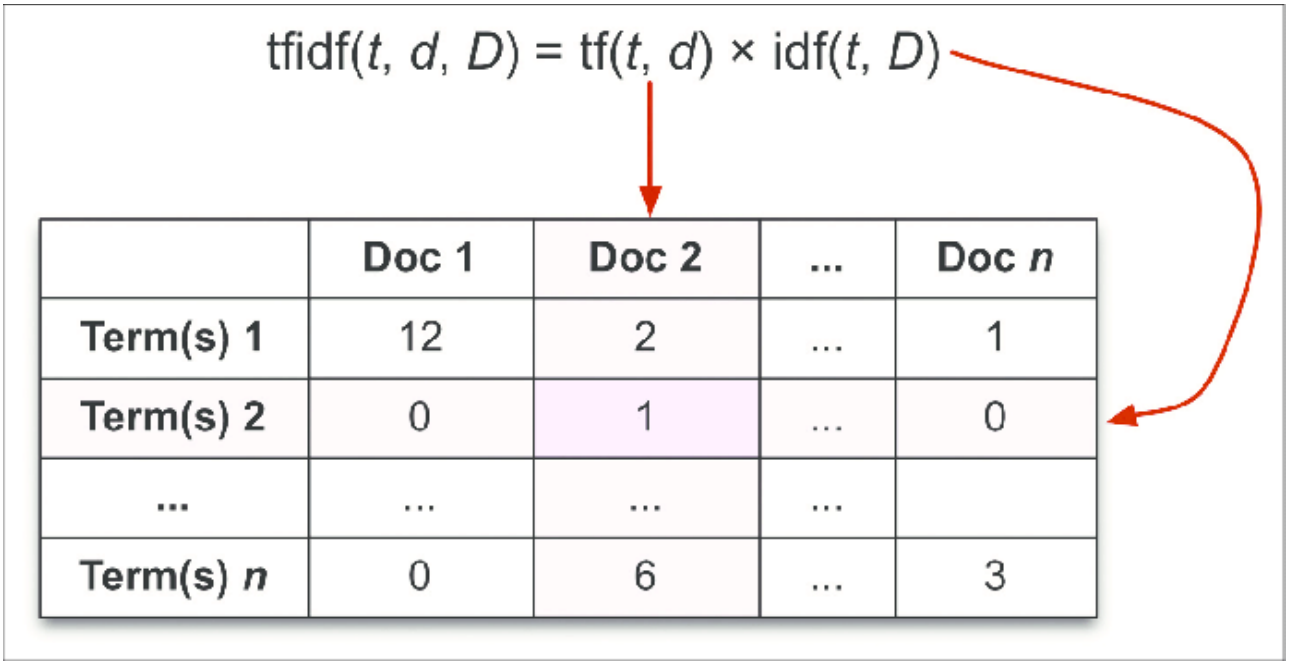
Поєднання N слів разом називаються N-грамами. N грами (N> 1), як правило, є більш інформативними порівняно зі словами (уніграмами) як ознаками. Крім того, біграми (N = 2) розглядаються як найважливіші ознаки серед всіх інших ознак тексту.

2.3.3 Статистичні ознаки тексту

Текстові дані також можна відображати у кількісних показниках. Одним з них є показникTF-IDF, що базується на частоті вживання термінів і представляє собою зважена модель, яка зазвичай використовується для вирішення проблем пошуку інформації. Він орієнтований на перетворення текстових документів у векторні моделі на основі частоти вживання слів у документах, не беручи до уваги точне впорядкування.

Наприклад, є набір даних із N текстових документів. У будь-якому документі «D», буде визначений частковий показник TF для терма «t», як кількість його входжень у документі "D". Обернена частота документів (IDF) - IDF для цього терма визначається як логарифм співвідношення загальної кількості документів, що є в корпусі, і кількості документів, що містять терм «t».

TF- IDF відображає відносну важливість терма у корпусі текстових документів і схематично показаний на рис. 2.9.



	Doc 1	Doc 2	...	Doc *n*
Term(s) 1	12	2	...	1
Term(s) 2	0	1	...	0
...	...	...	...	
Term(s) *n*	0	6	...	3

Рисунок 2.9 – Приклад статистичного показника TF-IDF

Модель створює словник і присвоює індекс кожному слову. Кожен рядок на виході містить кортеж (i, j) і значення tf-idf слова з індексом j у документі i.

Ознаки тексту засновані на підрахунку кількості слів або їх щільності, також можуть бути використані в моделях та аналізі документів. Ці особливості можуть здатися дріб'язковими, але вони мають сильний вплив на моделі навчання. Деякі особливості: кількість слів, кількість речень, пунктуація та кількість слів, визначених галуззю. Інші типи заходів включають такі показники читабельності, як кількість складів, показник спаму і простота читання.

2.3.4 Семантична векторизація текстових документів

Векторизація слів (Word embedding) – це сучасний спосіб представити слова у вигляді векторів. Метою векторизації слів є перевизначення високопросторових ознак слова у вектори менших розмірностей шляхом збереження контекстної подібності у корпусі документів. Вони широко використовуються у моделях глибокого навчання, таких як згорткові нейронні мережі та рекурентні нейронні мережі. Word2Vec та GloVe – дві популярні моделі для векторизації тексту. Ці моделі беруть корпус текстів як вхідні дані і створюють вектори слів як вихідні дані. Модель Word2Vec складається з модуля попередньої обробки, моделі неглибокої нейронної мережі, що називається Continuous Bag of Words, і іншої моделі «shallow» нейронної мережі, яка називається skip-gram. Спочатку формується словниковий запас з навчального корпусу, а потім проводиться їхня семантична векторизація. На рис. 2.10 показано модель Word2Vec

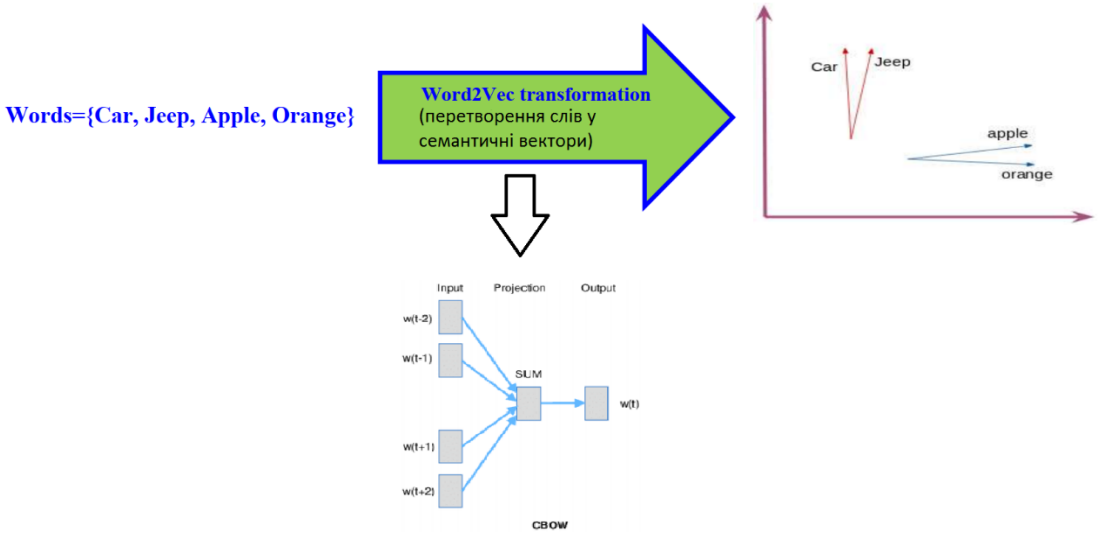


Рисунок 2.10 – Модель та архітектура Word2Vec

2.4 Алгоритм класифікації текстової інформації

Класифікація тексту - одна з класичних проблем в області опрацювання природної мови. До відомих прикладів належать такі задачі як:

– ідентифікація спаму електронної пошти;

– класифікація тем за новинами;

– класифікація настроїв людини;

– організація пошукових систем.

Класифікація тексту, або по-іншому рубрикація документів, загальними словами визначається як методика систематичної класифікації текстового об'єкта (документа чи речення) щодо його приналежності до однієї з фіксованих категорій. Це дійсно корисно, коли обсяг даних занадто великий, особливо для організації, фільтрації інформації та її зберігання.

Типовий класифікатор, що використовується при опрацюванні природних мов складається з двох частин (рис. 2.11):

– навчання;

– прогнозування.

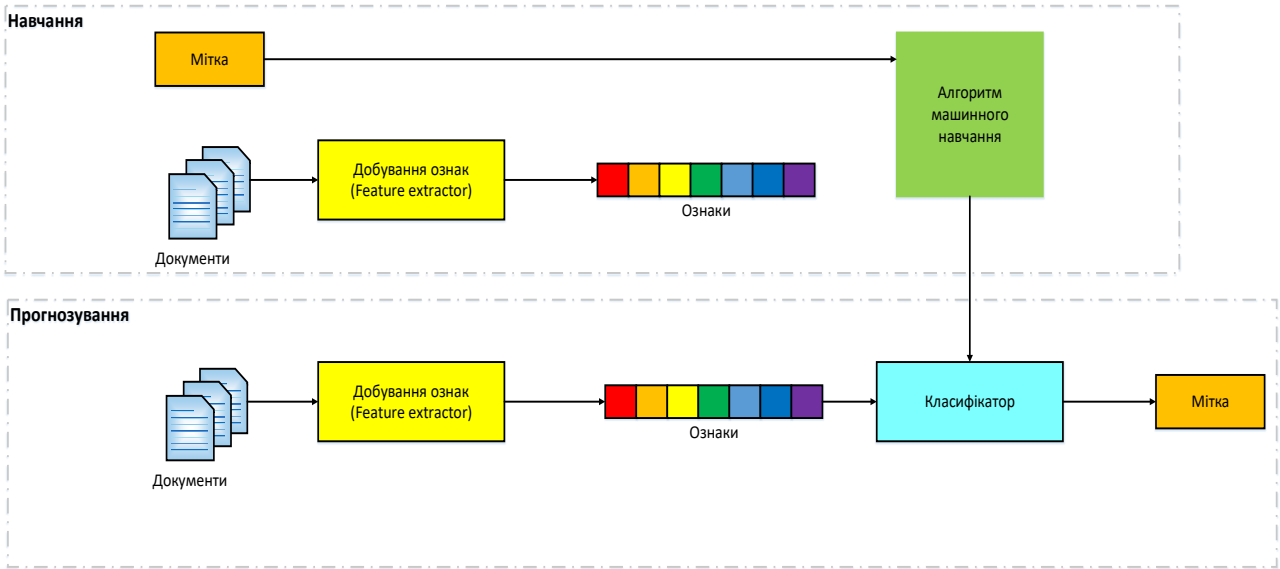


Рисунок 2.11 – Алгоритм класифікації текстових документів

Модель класифікації тексту сильно залежить від якості та кількості ознак, тому при застосуванні будь-якої моделі машинного навчання, завжди доцільно включати якомога більше навчальних даних.

Наївний Байєсівський класифікатор – це класифікаційний прийом, заснований на теоремі Байєса з припущенням незалежності серед існуючих ознак тексту, тобто це означає, що наявність певної ознаки в класі не пов'язана з наявністю будь-якої іншої ознаки тут. На рис. 2.12 показано приклад результату застосування алгоритму функціонування цього класифікатора.

Прогноз

	Так	Ні	P(так)	P (ні)
Сонячно	2	3	2/9	3/5
Похмуро	4	0	4/9	0/5
Дощ	3	2	3/9	2/5
Всього	9	5	100%	100%

Температура

	Так	Ні	P(так)	P (ні)
Жарко	2	2	2/9	2/5
Норм	4	2	4/9	2/5
Холод	3	1	3/9	1/5
Всього	9	5	100%	100%

Вологість

	Так	Ні	P(так)	P (ні)
Висока	3	4	3/9	4/5
Норм	6	1	6/9	1/5
Всього	9	5	100%	100%

Вітер

	Так	Ні	P(так)	P (ні)
Є	6	2	6/9	2/5
Немає	3	3	3/9	3/5
Всього	9	5	100%	100%

Play		P(так)/P(ні)
Так	9	9/14
Ні	5	5/14
Всього	14	100%

Рисунок 2.12 – Результат роботи Наївного Баєсівського класифікатора

Ще один класифікатор, який може застосовуватись при рубрикації текстових документів – це логістична регресія. Вона представляє собою алгоритм класифікації машинного навчання, що за допомогою логістичних функцій знаходить імовірність приналежності змінної до певної категорії. Результатом логістичної регресії є будь-яке двійкове значення, то виконється бінарна класифікація: належить чи не належить. Зараз даний алгоритм широко використовується для задач класифікації, оскільки існує багато прикладів, які свідчать про те, що модель логістичної регресії допомагає приймати рішення швидше після того, як вона пройде навчання на відповідній вибірці. На рис. 2.13 показано принцип функціонування логістичної регресії.

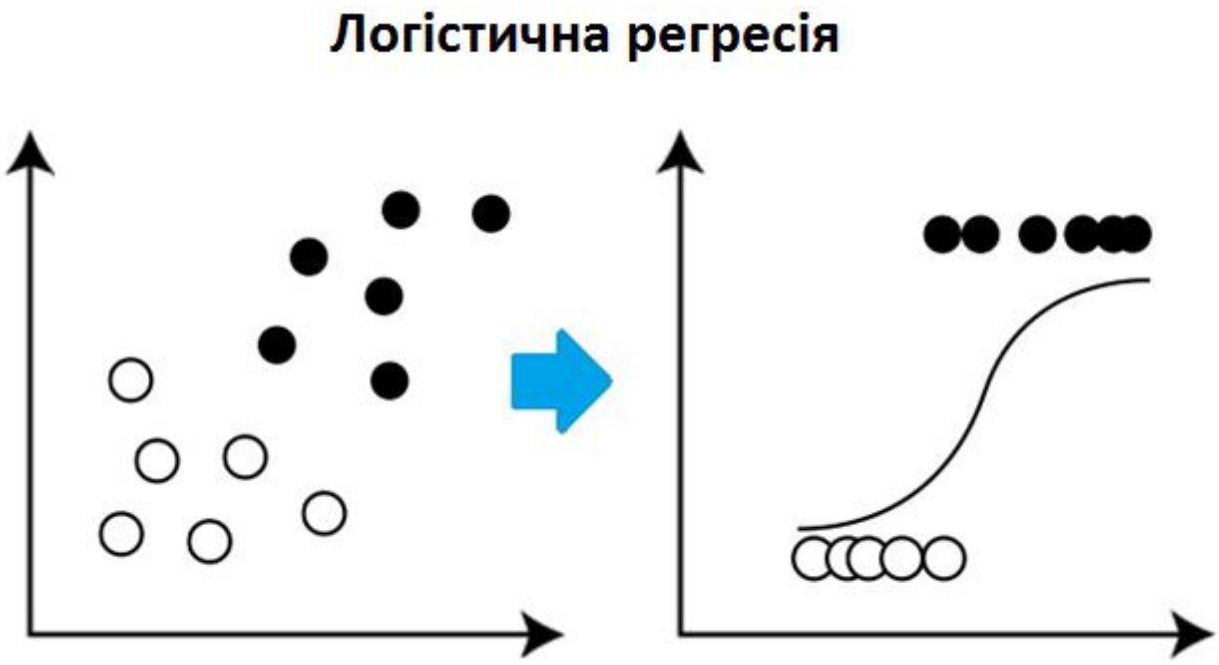


Рисунок 2.13 – Принцип роботи алгоритму логістичної регресії

Метод опорних векторів або «SVM» належить до лінійних алгоритмів, які використовуються в задачах класифікації та регресії. Даний алгоритм надає широкий спектр практичного застосування і може розв’язувати як лінійні так і нелінійні завдання. Суть алгоритму роботи на основі опорних векторів доволі проста: алгоритм формує лінію або гіперплощину, що розділяє дані на класи.

Основним завданням алгоритму є знаходження найбільш правильної лінію, або гіперплощини, що розділяє дані на два класи. SVM це алгоритм, який отримує на вході дані, і повертає лінію поділу на класи. На рис. 2.14 показаний приклад застосування алгоритму опорних векторів.

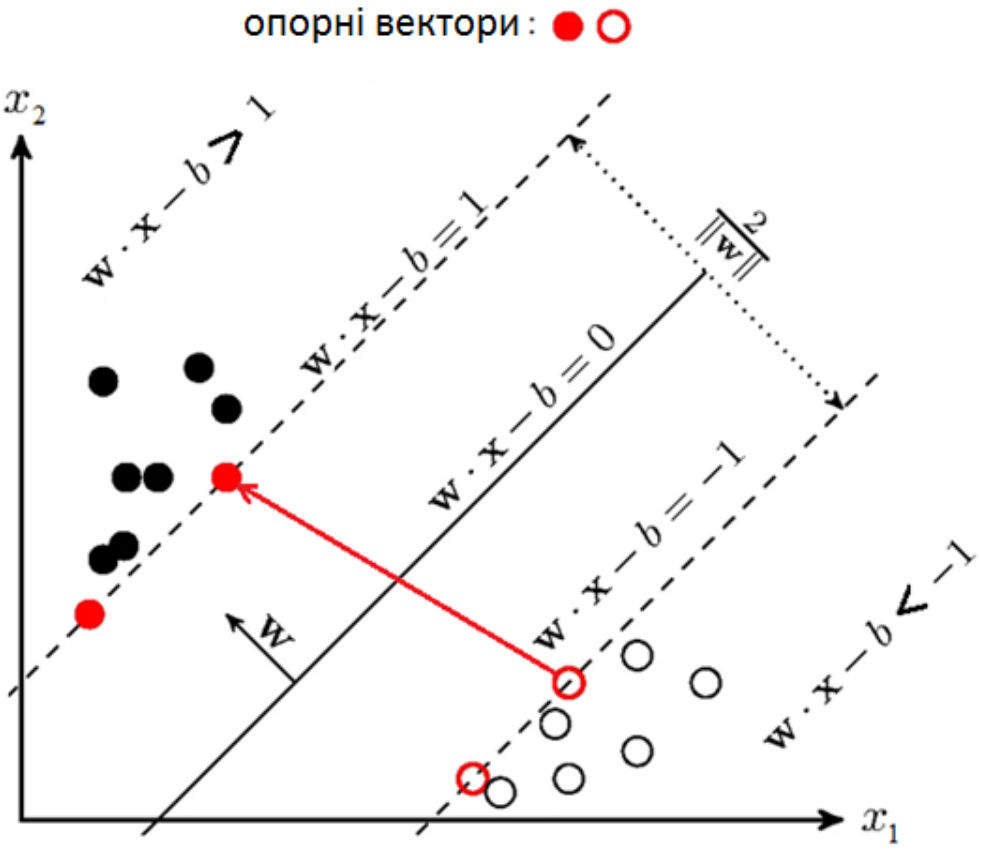


Рисунок 2.14 – Приклад застосування методу опорних векторів

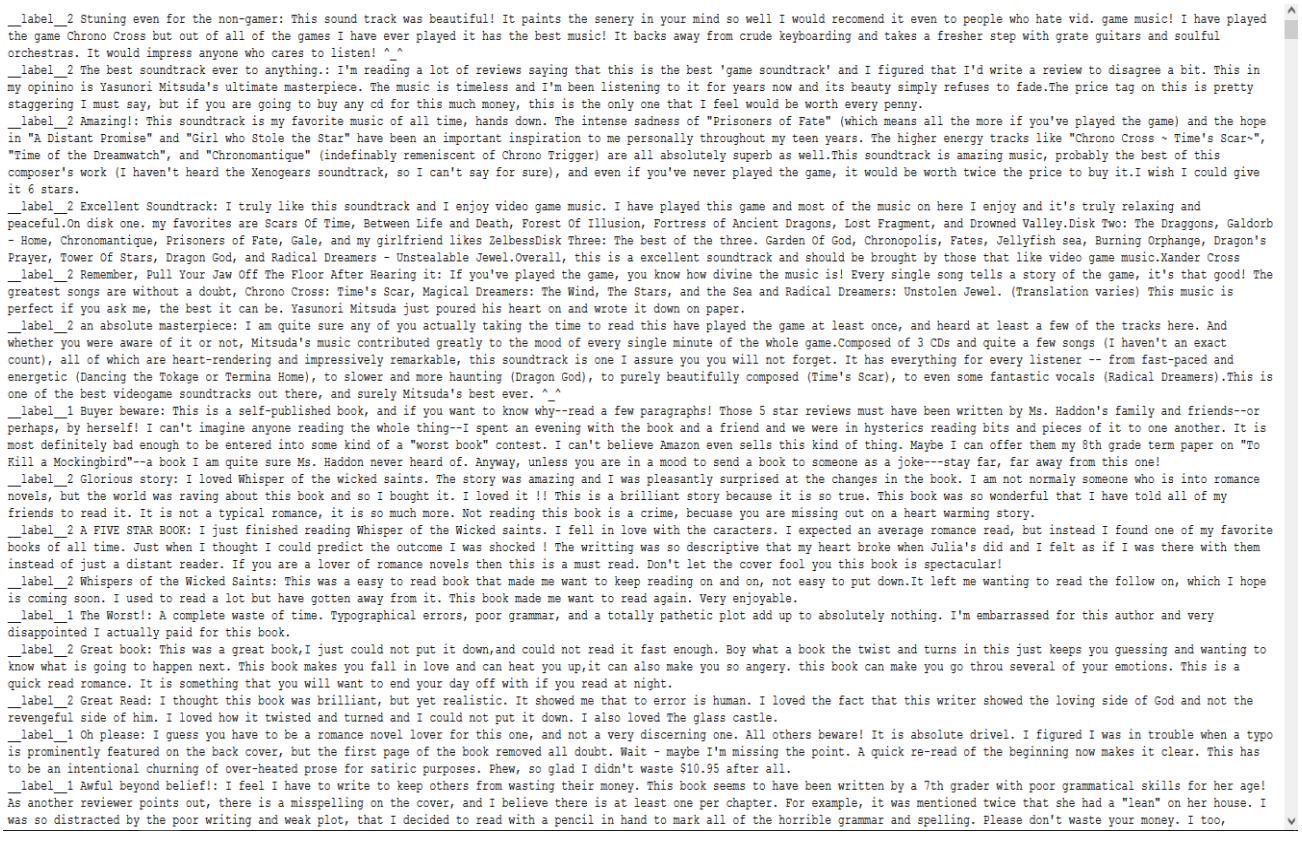
Алгоритм SVM влаштований таким чином, що він шукає точки на графіку, які розташовані безпосередньо до лінії поділу найближче. Ці точки називаються опорними векторами. Потім, алгоритм обчислює відстань між опорними векторами і розділяє їх площиною. Ця відстань називається зазором (margin). Основна мета алгоритму – максимізувати відстань зазору. Кращою гіперплощиною вважається така, для якої цей зазор є максимальним.

Окрім наведених вище моделей і алгоритмів широко застосовуються моделі глибоких нейронних мереж, практичну реалізацію яких наведено у наступному розділі.

РОЗДІЛ 3 ПРОГРАМНА РЕАЛІЗАЦІЯ КОМП'ЮТЕРИЗОВАНОЇ СИСТЕМИ ТЕМАТИЧНОЇ РУБРИКАЦІЇ ДОКУМЕНТІВ

3.1 Аналіз вхідного набору даних і його попереднє опрацювання

В якості вхідного набору даних у кваліфікаційній роботі використовується дата сет текстів щодо огляду Amazon. Набір даних складається з 3,6 млн текстових оглядів та їх міток, однак при реалізації комп'ютеризованої системи тематичної рубрикації буде використано лише частину дату сету, що пов'язано з обмеженістю наявних апаратних ресурсів. Вміст розміченого документу показано на рис. 3.1.



__label__2 Stuning even for the non-gamer: This sound track was beautiful! It paints the senery in your mind so well I would recomend it even to people who hate vid. game music! I have played the game Chrono Cross but out of all of the games I have ever played it has the best music! It backs away from crude keyboarding and takes a fresher step with grate guitars and soulful orchestras. It would impress anyone who cares to listen! ^_^
__label__2 The best soundtrack ever to anything.: I'm reading a lot of reviews saying that this is the best 'game soundtrack' and I figured that I'd write a review to disagree a bit. This in my opinino is Yasunori Mitsuda's ultimate masterpiece. The music is timeless and I'm been listening to it for years now and its beauty simply refuses to fade.The price tag on this is pretty staggering I must say, but if you are going to buy any cd for this much money, this is the only one that I feel would be worth every penny.
__label__2 Amazing!: This soundtrack is my favorite music of all time, hands down. The intense sadness of "Prisoners of Fate" (which means all the more if you've played the game) and the hope in "A Distant Promise" and "Girl who Stole the Star" have been an important inspiration to me personally throughout my teen years. The higher energy tracks like "Chrono Cross ~ Time's Scar~", "Time of the Dreamwatch", and "Chronomantique" (indefinably remeniscent of Chrono Trigger) are all absolutely superb as well.This soundtrack is amazing music, probably the best of this composer's work (I haven't heard the Xenogears soundtrack, so I can't say for sure), and even if you've never played the game, it would be worth twice the price to buy it.I wish I could give it 6 stars.
__label__2 Excellent Soundtrack: I truly like this soundtrack and I enjoy video game music. I have played this game and most of the music on here I enjoy and it's truly relaxing and peaceful.On disk one. my favorites are Scars Of Time, Between Life and Death, Forest Of Illusion, Fortress of Ancient Dragons, Lost Fragment, and Drowned Valley.Disk Two: The Draggons, Galdorb - Home, Chronomantique, Prisoners of Fate, Gale, and my girlfriend likes ZelbessDisk Three: The best of the three. Garden Of God, Chronopolis, Fates, Jellyfish sea, Burning Orphange, Dragon's Prayer, Tower Of Stars, Dragon God, and Radical Dreamers - Unstealable Jewel.Overall, this is a excellent soundtrack and should be brought by those that like video game music.Xander Cross
__label__2 Remember, Pull Your Jaw Off The Floor After Hearing it: If you've played the game, you know how divine the music is! Every single song tells a story of the game, it's that good! The greatest songs are without a doubt, Chrono Cross: Time's Scar, Magical Dreamers: The Wind, The Stars, and the Sea and Radical Dreamers: Unstolen Jewel. (Translation varies) This music is perfect if you ask me, the best it can be. Yasunori Mitsuda just poured his heart on and wrote it down on paper.
__label__2 an absolute masterpiece: I am quite sure any of you actually taking the time to read this have played the game at least once, and heard at least a few of the tracks here. And whether you were aware of it or not, Mitsuda's music contributed greatly to the mood of every single minute of the whole game.Composed of 3 CDs and quite a few songs (I haven't an exact count), all of which are heart-rendering and impressively remarkable, this soundtrack is one I assure you you will not forget. It has everything for every listener -- from fast-paced and energetic (Dancing the Tokage or Termina Home), to slower and more haunting (Dragon God), to purely beautifully composed (Time's Scar), to even some fantastic vocals (Radical Dreamers).This is one of the best videogame soundtracks out there, and surely Mitsuda's best ever. ^_^
__label__1 Buyer beware: This is a self-published book, and if you want to know why--read a few paragraphs! Those 5 star reviews must have been written by Ms. Haddon's family and friends--or perhaps, by herself! I can't imagine anyone reading the whole thing--I spent an evening with the book and a friend and we were in hysterics reading bits and pieces of it to one another. It is most definitely bad enough to be entered into some kind of a "worst book" contest. I can't believe Amazon even sells this kind of thing. Maybe I can offer them my 8th grade term paper on "To Kill a Mockingbird"--a book I am quite sure Ms. Haddon never heard of. Anyway, unless you are in a mood to send a book to someone as a joke---stay far, far away from this one!
__label__2 Glorious story: I loved Whisper of the wicked saints. The story was amazing and I was pleasantly surprised at the changes in the book. I am not normaly someone who is into romance novels, but the world was raving about this book and so I bought it. I loved it !! This is a brilliant story because it is so true. This book was so wonderful that I have told all of my friends to read it. It is not a typical romance, it is so much more. Not reading this book is a crime, becuase you are missing out on a heart warming story.
__label__2 A FIVE STAR BOOK: I just finished reading Whisper of the Wicked saints. I fell in love with the caracters. I expected an average romance read, but instead I found one of my favorite books of all time. Just when I thought I could predict the outcome I was shocked ! The writting was so descriptive that my heart broke when Julia's did and I felt as if I was there with them instead of just a distant reader. If you are a lover of romance novels then this is a must read. Don't let the cover fool you this book is spectacular!
__label__2 Whispers of the Wicked Saints: This was a easy to read book that made me want to keep reading on and on, not easy to put down.It left me wanting to read the follow on, which I hope is coming soon. I used to read a lot but have gotten away from it. This book made me want to read again. Very enjoyable.
__label__1 The Worst!: A complete waste of time. Typographical errors, poor grammar, and a totally pathetic plot add up to absolutely nothing. I'm embarrassed for this author and very disappointed I actually paid for this book.
__label__2 Great book: This was a great book,I just could not put it down,and could not read it fast enough. Boy what a book the twist and turns in this just keeps you guessing and wanting to know what is going to happen next. This book makes you fall in love and can heat you up,it can also make you so angery. this book can make you go throu several of your emotions. This is a quick read romance. It is something that you will want to end your day off with if you read at night.
__label__2 Great Read: I thought this book was brilliant, but yet realistic. It showed me that to error is human. I loved the fact that this writer showed the loving side of God and not the revengeful side of him. I loved how it twisted and turned and I could not put it down. I also loved The glass castle.
__label__1 Oh please: I guess you have to be a romance novel lover for this one, and not a very discerning one. All others beware! It is absolute drivel. I figured I was in trouble when a typo is prominently featured on the back cover, but the first page of the book removed all doubt. Wait - maybe I'm missing the point. A quick re-read of the beginning now makes it clear. This has to be an intentional churning of over-heated prose for satiric purposes. Phew, so glad I didn't waste $10.95 after all.
__label__1 Awful beyond belief!: I feel I have to write to keep others from wasting their money. This book seems to have been written by a 7th grader with poor grammatical skills for her age! As another reviewer points out, there is a misspelling on the cover, and I believe there is at least one per chapter. For example, it was mentioned twice that she had a "lean" on her house. I was so distracted by the poor writing and weak plot, that I decided to read with a pencil in hand to mark all of the horrible grammar and spelling. Please don't waste your money. I too,

Рисунок 3.1 – Вміст і структура вхідного набору даних

Після того, як завантажено вхідний набір даних потрібно поетапно виконати наступні операції. Перш за все необхідно імпортувати бібліотеки, які необхідні для проведення структурної рубрикації документів. Під документом, у даному випадку слід розуміти огляд Amazon з відповідною міткою. Імпорт необхідних бібліотек наведено у лістингу 3.1.

Лістинг 3.1 – Імпорт бібліотек, необхідних для рубрикації документів

```
from sklearn import model_selection, preprocessing,
linear_model, naive_bayes, metrics, svm
from sklearn.feature_extraction.text import TfidfVectorizer,
CountVectorizer
from sklearn import decomposition, ensemble

import pandas, xgboost, numpy, textblob, string
from keras.preprocessing import text, sequence
from keras import layers, models, optimizers
```

Об’єкти, що імпортуються з бібліотеки sklearn потрібні для виконання препроцесингу даних, побудови моделей класифікації на основі Баєсівського класифікатора, методу опорних векторів, лінійної моделі (логістичної регресії), а також для семантичної векторизації і статистичних показників вмісту документів.

Бібліотеки pandas, xgboost, numpy, textblob і string необхідні для виконання перетворень над текстовими і числовими даними. Бібліотека keras буде використовуватись для реалізації моделей класифікації на основі глибоких нейронних мереж.

Щоб підготувати набір даних, перш за все потрібно завантажити дані у фрейм pandas, що буде складатись з двох стовпців – тексту та мітки. Для цього необхідно виконати програмний код, який наведений у лістингу 3.2.

Змінна data буде містити в собі повністю вміст вхідного набору даних, який зчитується з місця розташування: «data/corpus».

Після цього створюються дві змінні, які спочатку представляють собою порожні списки: labels і texts. Змінна labels відповідає за мітки тексту, а text – власне за вмістиме документу (огляду). Після цього в циклі формується записи у попередньо створені списки з додавання розділювача по типу нової стрічки. У результаті створення об’єкту дата фрейм, одержуємо в одному стовпці «text» - список вмісту документів, а в іншому «label» – список відповідних міток текстів.

Лістинг 3.2 – Завантаження і препроцесинг даних

```
# завантаження дата сету
data = open('data/corpus').read()
labels, texts = [], []
for i, line in enumerate(data.split("\n")):
    content = line.split()
    labels.append(content[0])
    texts.append(" ".join(content[1:]))

# створення фрейму текстів і міток
trainDF = pandas.DataFrame()
trainDF['text'] = texts
trainDF['label'] = labels
```

Далі потрібно виконати декомпозицію набору даних на навчальні і тестові вибірки для того, щоб можна було в подальшому оцінити якість класифікації. Окрім цього, доцільно провести кодування цільової змінної, щоб забезпечити ефективність і продуктивність моделей машинного навчання. У лістингу 3.3 приведено програмний код поділу на навчальну і тестову вибірки, а також кодування змінної, яка відповідає стовпцю «label» у дата фреймі trainDF.

Лістинг 3.3 – Кодування цільової змінної і поділ набору даних

```
# розбиття набору даних на навчальну і тестову вибірки
train_x, valid_x, train_y, valid_y =
model_selection.train_test_split(trainDF['text'],
trainDF['label'])
# кодування цільової змінної
encoder = preprocessing.LabelEncoder()
train_y = encoder.fit_transform(train_y)
valid_y = encoder.fit_transform(valid_y)
```

У результаті проведення маніпуляцій, які наведені у лістингах вище одержано набори даних на яких можна проводити вже інженерію ознак та будувати моделі класифікації текстових документів.

3.2 Виявлення ознак текстових документів

На цьому кроці неструктуровані текстові дані будуть перетворені у вектори об’єктів, а нові ознаки будуть створені за допомогою наявного набору даних.

Для виявлення ознак текстових документів пропонується використати наступні методи і підходи:

1. Count Vectorizer – перетворення колекції текстових документів на матрицю з розподілом токенів слів.
2. Векторизація на основі TF-IDF як ознаки:

– рівень слова;

– півень N-грам;

– рівень сутностей;

– вбудовані слова як ознаки (word embedding);

3. Ознаки на основі NLP.
4. Моделі тем як ознаки текстових докуентів.

Count Vectorizer – це матрична нотація набору даних, у якій кожен рядок представляє документ із корпусу, кожен стовпець – термін із корпусу, а кожна комірка – підрахунок частоти певного терміну в конкретному документі. Приклад матриці, одержаної при виконанні такого методу показана на рис. 3.2.

Training examples	Features		
	love	programming	also
1 → I love programming	1	1	0
2 → Programming also loves me	1	1	1

Рисунок 3.2 – Матриця, що утворюється внаслідок векторизації документу

У лістингу 3.4 показано програмний код виявлення ознак на основі підходу Count Vectorizer.

Лістинг 3.4 – Програмна реалізація методу Count Vectorizer

```
# створення count vectorizer об’єкту
count_vect = CountVectorizer(analyzer='word',
token_pattern=r'\w{1,}')
count_vect.fit(trainDF['text'])

# перетворення навчальної і тестової вибірки
# з використанням count vectorizer об’єкту
xtrain_count =  count_vect.transform(train_x)
xvalid_count =  count_vect.transform(valid_x)
```

В подальшому, одержана матриця буде використана при обгрунтуванні та експериментальному дослідженні різних моделей прогнозування для забезпечення тематичної рубрикації документів.

Оцінка TF-IDF представляє собою відносну важливість терміна у документі та в цілому корпусі.

Оцінка TF-IDF складається з двох доданків:

перший доданок обчислює нормалізовану частоту (TF);

другий доданок – обернену частоту документів (IDF), що обчислюється як логарифм кількості документів у корпусі, поділений на кількість документів, де фігурує конкретний термін.

В загальному випадку, формулу для знаходження TF та IDF можна записати наступним чином:

$$\mathrm{TF}\,(\mathrm{t}) = \frac{\text{Частота появи терміну t у документі}}{\text{Загальна кількість термінів у документі}} \qquad (3.1)$$

$$\mathrm{IDF}\,(\mathrm{t}) = \log\frac{\text{Загальна кількість документів}}{\text{Кількість документів із терміном t}} \qquad (3.2)$$

Вектори TF-IDF можуть генеруватися на різних рівнях вхідних лексем (слова, символи, n-грами):

– рівень слова TF-IDF – матриця, що представляє оцінки tf-idf кожного терміна в різних документах;

– N-грамний рівень TF-IDF – N-грами – це поєднання N термінів разом і відповідно матриця представляється у вигляді tf-idf N-грамів;

– рівень символів TF-IDF – матриця, що представляє оцінки tf-idf рівня n-грамів символу в корпусі документів.

Програмна реалізація векторів TF-IDF на різних рівнях наведена у лістингах 3.5 – 3.7. У лістингу 3.5 показано реалізацію TF-IDF на рівні слів.

Лістинг 3.5 – Реалізація TF-IDF на рівні слів

```
# рівень слів tf-idf
tfidf_vect = TfidfVectorizer(analyzer='word',
token_pattern=r'\w{1,}', max_features=5000)
tfidf_vect.fit(trainDF['text'])
```

```
xtrain_tfidf =  tfidf_vect.transform(train_x)
xvalid_tfidf =  tfidf_vect.transform(valid_x)
```

У лістингу 3.6 показано реалізацію векторизації документу на рівні N-грам.

Лістинг 3.6 – Реалізація TF-IDF на рівні N-грам

```
# рівент n gram tf-idf
tfidf_vect_ngram = TfidfVectorizer(analyzer='word',
token_pattern=r'\w{1,}', ngram_range=(2,3),
max_features=5000)
tfidf_vect_ngram.fit(trainDF['text'])
xtrain_tfidf_ngram =  tfidf_vect_ngram.transform(train_x)
xvalid_tfidf_ngram =  tfidf_vect_ngram.transform(valid_x)
```

При реалізації алгоритму TF-IDF на рівні N-грам використано біграми та триграми, що задано у функції параметром «`ngram_range=(2,3)`». Подібно реалізується алгоритм на рівні символів, що показано у лістингу 3.7.

Лістинг 3.7 – Реалізація TF-IDF на рівні символів

```
# tf-idf на рівні символів
tfidf_vect_ngram_chars = TfidfVectorizer(analyzer='char',
token_pattern=r'\w{1,}', ngram_range=(2,3),
max_features=5000)
tfidf_vect_ngram_chars.fit(trainDF['text'])
xtrain_tfidf_ngram_chars =
tfidf_vect_ngram_chars.transform(train_x)
xvalid_tfidf_ngram_chars =
tfidf_vect_ngram_chars.transform(valid_x)
```

Word embedding – це форма представлення слів і документів за допомогою щільності векторного представлення. Позиція слова у векторному просторі

добувається з тексту і базується на словах, які знаходяться біля цього слова. Вбудовування слів можна навчити, використовуючи сам корпус докуентів, або створити за допомогою попередньо навчених вбудовувань слів, таких як Glove, FastText та Word2Vec.

Будь-який з них можна завантажити та використовувати як навчальний дата сет. Для цього потрібно виконати такі кроки:

– завантаження попередньо навчених вкладених слів;

– створення токенізатора;

– перетворення текстових документів у послідовність маркерів та їх заповнення;

– створення відображення токена та відповідних вбудовувань.

Виконання першого кроку щодо завантаження попередньо навчених вкладених слів на ведено у лістингу 3.8.

Лістинг 3.8 – Завантаження попередньо навчених вкладених слів

```
# завантаження pre-trained word-embedding векторів
embeddings_index = {}
for i, line in enumerate(open('data/wiki-news-300d-1M.vec')):
    values = line.split()
    embeddings_index[values[0]] = numpy.asarray(values[1:],
dtype='float32')
```

Програмна реалізація другого кроку представлена у лістингу 3.9.

Лістинг 3.9 – Створення маркерів об’єктів

```
# створення токенізатора
token = text.Tokenizer()
token.fit_on_texts(trainDF['text'])
word_index = token.word_index
```

Наступні два кроки алгоритму наведено у лістингу 3.10.

Лістинг 3.10 – Програмний код перетворення токенів у послідовності

```
# перетворення тексту у послідовність токенів
train_seq_x =
sequence.pad_sequences(token.texts_to_sequences(train_x),
maxlen=70)
valid_seq_x =
sequence.pad_sequences(token.texts_to_sequences(valid_x),
maxlen=70)

# створення token-embedding відображення
embedding_matrix = numpy.zeros((len(word_index) + 1, 300))
for word, i in word_index.items():
    embedding_vector = embeddings_index.get(word)
    if embedding_vector is not None:
        embedding_matrix[i] = embedding_vector
```

Також можна створити ряд додаткових ознак тексту, які іноді корисні для вдосконалення моделей класифікації тексту. Наприклад, такми ознакми можуть бути:

– загальна кількість слів у документах;
– загальна кількість символів у документах;
– середня довжина слів, що використовуються у документах;
– загальна кількість розділових знаків у документах;
– загальна кількість слів із верхнього регістру в документах;
– кількість заголовних слів у документах4
– частотний розподіл тегів за частинами мови: кількість іменників, кількість дієслів, кількість прикметників, кількість прислівників, кількість займенників.

Ці особливості є надзвичайно експериментальними та повинні використовуватися лише відповідно до постановки проблеми.

Моделювання тем – це техніка виявлення груп слів (що називається темою) із корпусу документів. У кваліфікаційній роботі використовується прихований розподіл Діріхле для створення ознак тем. LDA – це ітераційна модель, яка базується на основі фіксованої кількості тем.

Кожна тема представлена у вигляді розподілу за словами, а кожен документ – у вигляді розподілу за темами. Незважаючи на те, що самі лексеми використовувати не доцільно, розподіл ймовірностей за словами, передбачений темами, дає уявлення про різні ідеї, що містяться в документах. Реалізація алгоритму LDA наведена у лістингу 3.12.

Лістинг 3.12 – Визначення тем на основі LDA

```
# навчання LDA моделі
lda_model =
decomposition.LatentDirichletAllocation(n_components=20,
learning_method='online', max_iter=20)
X_topics = lda_model.fit_transform(xtrain_count)
topic_word = lda_model.components_
vocab = count_vect.get_feature_names()

# перегляд тем моделей
n_top_words = 10
topic_summaries = []
for i, topic_dist in enumerate(topic_word):
    topic_words =
numpy.array(vocab)[numpy.argsort(topic_dist)][:-
(n_top_words+1):-1]
    topic_summaries.append(' '.join(topic_words))
```

3.3 Реалізація рубрикатора документів на основі моделей класифікації

Оскільки, у попередньому розділі першою моделлю для класифікації текстових документів використовувалась модель Баєса, то програмна її реалізація з врахуванням висвітлених аспектів показана у лістингу 3.13.

Лістинг 3.13 – Баєсівський класифікатор

```
# Класифікатор Count Vectors
accuracy = train_model(naive_bayes.MultinomialNB(),
xtrain_count, train_y, xvalid_count)
print "NB, Count Vectors: ", accuracy

# Класифікатор Word Level TF IDF
accuracy = train_model(naive_bayes.MultinomialNB(),
xtrain_tfidf, train_y, xvalid_tfidf)
print "NB, WordLevel TF-IDF: ", accuracy

# Класифікатор on Ngram Level TF IDF Vectors
accuracy = train_model(naive_bayes.MultinomialNB(),
xtrain_tfidf_ngram, train_y, xvalid_tfidf_ngram)
print "NB, N-Gram Vectors: ", accuracy

# Класифікатор on Character Level TF IDF Vectors
accuracy = train_model(naive_bayes.MultinomialNB(),
xtrain_tfidf_ngram_chars, train_y, xvalid_tfidf_ngram_chars)
print "NB, CharLevel Vectors: ", accuracy
```

Результат щодо точності класифікації документів на основі моделі Баєса з врахуванням різновидів TF-IDF показано на рис. 3.2.

```
NB, Count Vectors:  0.7004
NB, WordLevel TF-IDF:  0.7024
NB, N-Gram Vectors:  0.5344
NB, CharLevel Vectors:  0.6872
```

Рисунок 3.2 – Результат рубрикації документів на основі Баєсівського класифікатора

Логістична регресія вимірює взаємозв'язок між категоріально залежною змінною та однією або кількома незалежними змінними шляхом оцінки ймовірностей за допомогою логістичної / сигмоїдної функції. Програмна реалізація моделі логістичної регресії показана у лістингу 3.14.

Лістинг 3.14 – Класифікатор на основі логістичної регресії

```
# Лінійний класифікатор на основі Count Vectors
accuracy = train_model(linear_model.LogisticRegression(),
xtrain_count, train_y, xvalid_count)
print "LR, Count Vectors: ", accuracy

# Лінійний класифікатор на основі Word Level TF IDF Vectors
accuracy = train_model(linear_model.LogisticRegression(),
xtrain_tfidf, train_y, xvalid_tfidf)
print "LR, WordLevel TF-IDF: ", accuracy

# Лінійний класифікатор на основі Level TF IDF Vectors
accuracy = train_model(linear_model.LogisticRegression(),
xtrain_tfidf_ngram, train_y, xvalid_tfidf_ngram)
print "LR, N-Gram Vectors: ", accuracy

# Лінійний класифікатор на основі Character Level TF IDF
Vectors
accuracy = train_model(linear_model.LogisticRegression(),
xtrain_tfidf_ngram_chars, train_y, xvalid_tfidf_ngram_chars)
print "LR, CharLevel Vectors: ", accuracy
```

Результати щодо точності класифікації документів на основі логістичної регресії наведено на рис. 3.3.

```
LR, Count Vectors:  0.7048
LR, WordLevel TF-IDF:  0.7056
LR, N-Gram Vectors:  0.4896
LR, CharLevel Vectors:  0.7012
```

Рисунок 3.3 – Результати рубрикації на основі логістичної регресії

Метод опорних векторів є алгоритм машинного навчання із вчителем, який можна використовувати як для рубрикації документів. У лістингу 3.15 наведено реалізацію цього алгоритму, а на рис. 3.5 – точність класифікації.

Лістинг 3.15 – Реалізація класифікатора на основі методу опорних векторів

```
# SVM on Ngram Level TF IDF Vectors
accuracy = train_model(svm.SVC(), xtrain_tfidf_ngram,
train_y, xvalid_tfidf_ngram)
print "SVM, N-Gram Vectors: ", accuracy
```

```
SVM, N-Gram Vectors:  0.5296
```

Рисунок 3.5 – Точність класифікації на основі методу опорних векторів

Моделі випадкових лісів є різновидом ансамблевих моделей. Вони є частиною сімейства дерев прийняття рішень. Ці моделі показують доволі високу точність при розв’язані задач класифікації і регресії, тому її також було реалізовано для рубрикації текстових документів (лістинг 3.16).

Лістинг 3.16 – Рубрикація документів на основі Random Forest

```
# RF on Count Vectors
accuracy = train_model(ensemble.RandomForestClassifier(),
xtrain_count, train_y, xvalid_count)
print "RF, Count Vectors: ", accuracy

# RF on Word Level TF IDF Vectors
accuracy = train_model(ensemble.RandomForestClassifier(),
xtrain_tfidf, train_y, xvalid_tfidf)
print "RF, WordLevel TF-IDF: ", accuracy
```

Результат щодо точності рубрикації документів наведено на рис. 3.6.

```
RF, Count Vectors:  0.6972

RF, WordLevel TF-IDF:  0.6988
```

Рисунок 3.6 – Точність рубрикації на базі Random Forest

Нейронна мережа – це математична модель, яка призначена для прогнозування поведінки, яка емулює функціонування біологічних нейронів та нервової системи. Ці моделі використовуються для розпізнавання складних образів та взаємозв'язків, що існують у межах маркованих даних. Неглибока нейронна мережа містить в основному три типи шарів – вхідний, прихований і вихідний. Структура простої нейронної мережі показана на рис. 3.7.

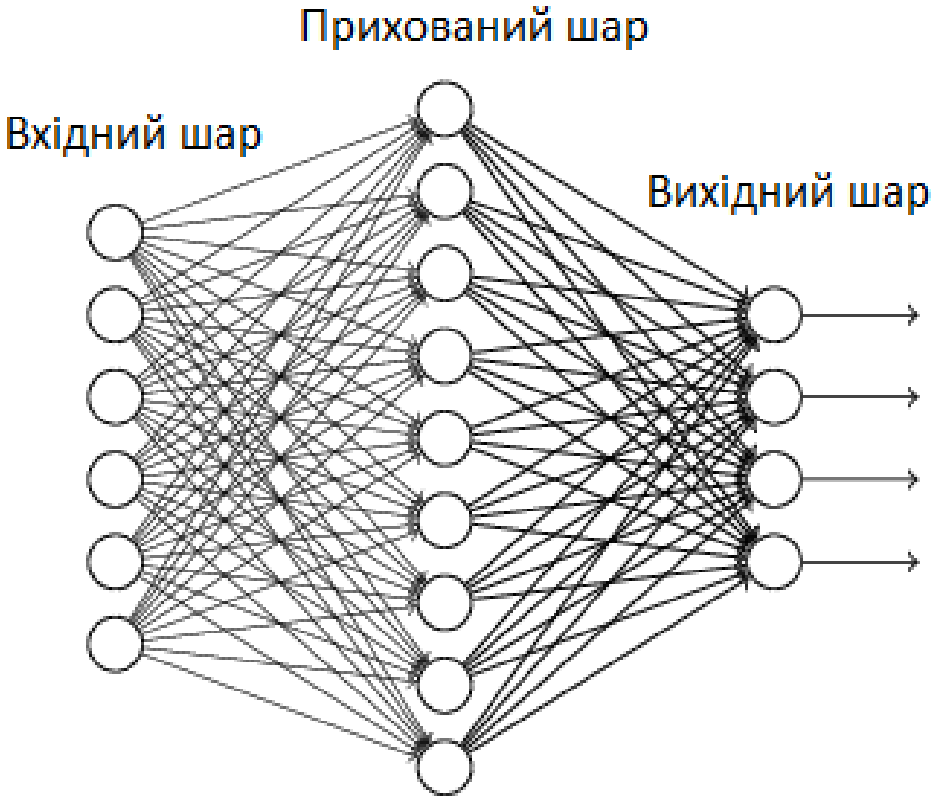


Рисунок 3.7 – Структура простої нейронної мережі

Програмна функція щодо реалізації простої нейронної мережі (НМ), до складу якої входить лише три шари, наведено у лістингу 3.17.

Лістинг 3.17 – Функція побудови простої нейронної мережі

```
def create_model_architecture(input_size):
    # create input layer
    input_layer = layers.Input((input_size, ), sparse=True)

    # create hidden layer
    hidden_layer = layers.Dense(100,
activation="relu")(input_layer)

    # create output layer
    output_layer = layers.Dense(1,
activation="sigmoid")(hidden_layer)

    classifier = models.Model(inputs = input_layer, outputs =
output_layer)
    classifier.compile(optimizer=optimizers.Adam(),
loss='binary_crossentropy')
    return classifier
```

Безпосередня реалізація класифікатора документів на основі простої нейронної мережі наведено у лістингу 3.18.

Лістинг 3.18 – Створення класифікатора на основі простої НМ

```
classifier =
create_model_architecture(xtrain_tfidf_ngram.shape[1])
accuracy = train_model(classifier, xtrain_tfidf_ngram,
train_y, xvalid_tfidf_ngram, is_neural_net=True)
print "NN, Ngram Level TF IDF Vectors",  accuracy
```

Точність результату виконання рубрикації текстових документів на основі класифікатора, наведеного у лістингу 3.18 показано на рис. 3.8

```
Epoch 1/1
7500/7500 [==============================] - 1s 67us/step - loss: 0.6909
NN, Ngram Level TF IDF Vectors 0.5296
```

Рисунок 3.8 – Точність класифікації документів на основі простої НМ

Глибокі нейронні мережі – це більш складні нейронні мережі, в яких приховані шари виконують набагато складніші операції, ніж прості активації сигмоподібних функцій або relu. Різні типи моделей глибокого навчання можуть застосовуватися в задачах класифікації тексту. На рис. 3.9 показано типову загальну структуру глибокої нейронної мережі.

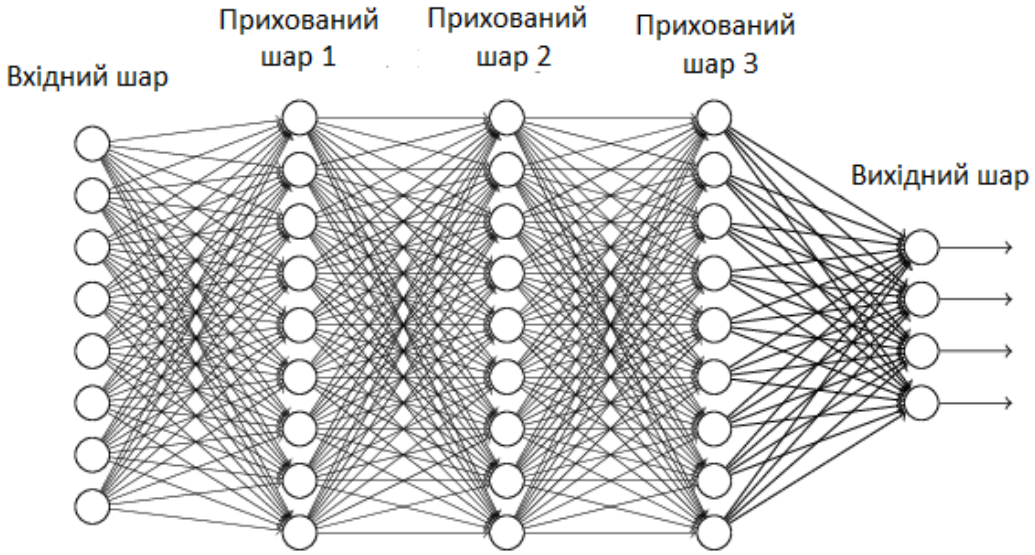


Рисунок 3.9 – Приклад структури глибокої нейронної мережі

Одним з екземлярів глибокої нейронної мережі є клас згорткових нейронних мереж. У таких мережах для обчислення вихідних даних використовуються згортки над вхідним шаром. Це призводить до локальних зв’язків, де кожна область входу пов’язана з нейроном на виході. Кожен шар застосовує різні фільтри та поєднує їх результати. На рис. 3.10 показано приклад згорткової нейронної мережі.

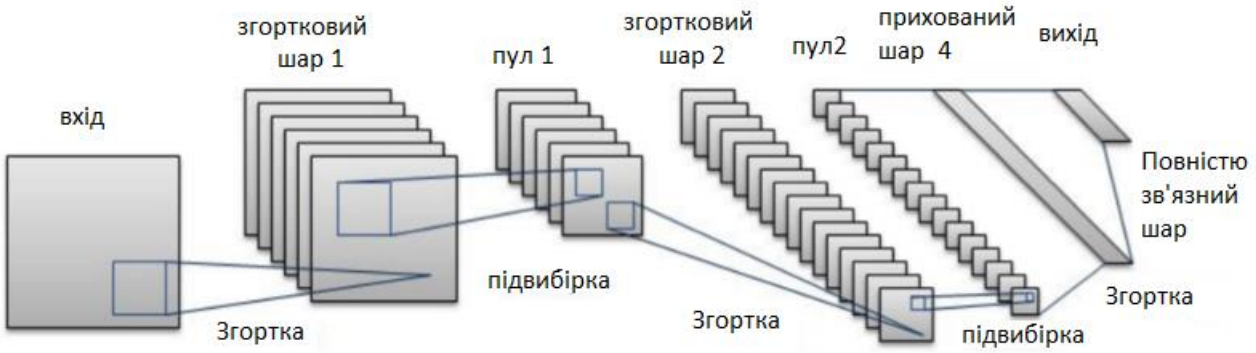


Рисунок 3.10 – Приклад згорткової нейронної мережі

Реалізація класифікатора на основі згорткової нейронної мережі наведена у лістингу 3.19, а точність класифікації представлена на рис. 3.11.

Лістинг 3.19 – Реалізація рубрикатора на основі CNN

```
def create_cnn():
    # Add an Input Layer
    input_layer = layers.Input((70, ))

    # Add the word embedding Layer
    embedding_layer = layers.Embedding(len(word_index) + 1, 300,
weights=[embedding_matrix], trainable=False)(input_layer)
    embedding_layer = layers.SpatialDropout1D(0.3)(embedding_layer)

    # Add the convolutional Layer
    conv_layer = layers.Convolution1D(100, 3,
activation="relu")(embedding_layer)

    # Add the pooling Layer
    pooling_layer = layers.GlobalMaxPool1D()(conv_layer)

    # Add the output Layers
    output_layer1 = layers.Dense(50,
activation="relu")(pooling_layer)
```

```
    output_layer1 = layers.Dropout(0.25)(output_layer1)

    output_layer2 = layers.Dense(1,
activation="sigmoid")(output_layer1)

    # Compile the model
    model = models.Model(inputs=input_layer,
outputs=output_layer2)
    model.compile(optimizer=optimizers.Adam(),
loss='binary_crossentropy')

    return model

classifier = create_cnn()
accuracy = train_model(classifier, train_seq_x, train_y,
valid_seq_x, is_neural_net=True)
print "CNN, Word Embeddings",  accuracy
```

```
Epoch 1/1
7500/7500 [==============================] - 12s 2ms/step - loss: 0.5847
CNN, Word Embeddings 0.5296
```

Рисунок 3.11 – Результат класифікації на основі CNN

У роботі також використано ряд інших моделей, програмний код реалізації яких наведений у додатку Б, зокрема серед них є рекурентна нейронна мережа і її різновиди:

– двонаправлена нейронна мережа;
– рекурентна нейронна мережа на основі LSTM;
– рекурентна нейронна мережа на основі GRU;
– рекурентна згорткова нейронна мережа.

Результат щодо точності рубрикації текстових документів без попереднього препроцесингу наведено у табл. 3.1.

Таблиця 3.1 – Результати точності рубрикації текстових документів

№ з/п	Модель	Тип ознак	Точність
1.	Наївний Баєсівський класифікатор	Count Vectors	0.7004
		TF-IDF на рівні слів	0.7024
		TF-IDF на рівні N-грам	0.5344
		TF-IDF на рівні символів	0.6872
2.	Логістична регресія	Count Vectors	0.7048
		TF-IDF на рівні слів	0.7056
		TF-IDF на рівні N-грам	0.4896
		TF-IDF на рівні символів	0.7012
3.	SVM	TF-IDF на рівні N-грам	0.5296
4.	Випадкові ліси (Random Forest)	Count Vectors	0.6972
		TF-IDF на рівні слів	0.6988
5.	XGBoost	Count Vectors	0.6324
		TF-IDF на рівні слів	0.6364
		TF-IDF на рівні символів	0.6548
6.	Прецептрон (проста нейронна мережа)	TF-IDF на рівні N-грам	0.5296
7.	CNN	Семантична векторизація	0.5296
8.	RNN-LSTM	Семантична векторизація	0.5124
9.	RNN-GRU	Семантична векторизація	0.5124
10.	RNN-Bidirectional	Семантична векторизація	0.5124
11.	RCNN	Семантична векторизація	0.5124

Як видно з табл. 3.1, найкращою моделлю для рубрикації текстових неструктурованих документів на заданому вхідному наборі даних є логістична регресія, що використовує TF-IDF на рівні слів.

РОЗДІЛ 4 БЕЗПЕКА ЖИТТЄДІЯЛЬНОСТІ, ОСНОВИ ОХОРОНИ ПРАЦІ

4.1 Суть та зміст управління охороною праці

Основними завданнями управління охороною праці є:

1) опрацювання заходів щодо здійснення державної політики з охорони праці на регіональному та галузевому рівнях;

2) підготовка, прийняття та реалізація заходів, спрямованих на забезпечення:

– належних, безпечних і здорових умов праці;

– утримання в належному стані виробничого устаткування, будівель і споруд, інженерних мереж, безпечного ведення технологічних процесів;

– необхідних засобів індивідуального захисту для працівників;

– організації і проведення навчання працівників з питань охорони праці;

– пропаганди охорони праці;

– обліку, аналізу та оцінки стану умов і безпеки праці; -професійного добору працівників окремих спеціальностей;

– страхування працівників від нещасного випадку на виробництві та профзахворювань;

3) організаційно-методичне керівництво на регіональному та галузевому рівнях;

4) стимулювання інтеграції управління охороною праці в єдину систему загального управління організацією виробництва;

5) широке впровадження позитивного досвіду у сфері охорони праці.

Основні функції управління охороною праці:

– організація та координація робіт у галузі охорони праці;

– облік, аналіз та оцінка показників стану умов та безпеки праці;

– планування та фінансування робіт;

– контроль за дотриманням вимог нормативно-правових актів з питань охорони праці.

Нормативно-правове забезпечення управління охороною праці має вдосконалюватися у таких напрямках:

– необхідно продовжити перебудову чинної нормативно-правової бази з охорони праці з урахуванням сучасних умов, вимог законодавства України, міжнародних або європейських норм;

– після прийняття нової редакції Закону України "Про охорону праці" слід переглянути відповідні нормативно-правові акти;

– проаналізувати стан нормативно-правової бази, визначити пріоритети щодо черговості перегляду нормативно-правових актів з охорони праці;

– необхідно забезпечити розробку та реалізацію в кожній галузі перспективних і поточних планів нормотворчої діяльності та опрацювання проектів ДНАОП на рівні сучасних вимог;

– на допомогу суб'єктам малого й середнього бізнесу ННДІОП опрацьовує довідково-методичні матеріали з питань охорони праці.

Першочерговим у системі управління охороною праці є забезпечення органів державного управління охорони праці та служб охорони праці підприємств, установ, організацій кваліфікованими фахівцями з охорони праці.

Належна кваліфікація й обізнаність усіх працівників із питань охорони праці є запобіжником ризику отримати виробничу травму чи професійне захворювання. Тому у процесі реформування управління охороною праці одним із найбільш пріоритетних напрямів є підвищення рівня знань працівників із цих питань, що має забезпечуватися у закладах освіти і безперервно шляхом навчання працівників у процесі їх трудової діяльності.

Для підвищення рівня знань фахівців із питань охорони праці необхідно:

– опрацювати проект положення "Про підготовку, перепідготовку та підвищення кваліфікації працівників системи Держнаглядохоронпраці";

– розробити й реалізувати комплекс заходів щодо визначення пріоритетних питань при підготовці та підвищенні кваліфікації державних інспекторів з охорони праці з урахуванням наукових досліджень, досягнень та технічних рішень щодо створення безпечних умов праці в галузях виробництва.

Інформаційне забезпечення в галузі охорони праці має здійснюватися органами управління охороною праці на всіх рівнях і потребує вдосконалення шляхом визначення та поширення міжнародного й вітчизняного досвіду щодо пропаганди безпечних методів і засобів праці, вирішення інших актуальних питань у цій сфері із залученням сучасних інформаційних технологій, ЗМІ, оперативного розповсюдження посібників, пам'яток, методик, листівок відповідного спрямування.

ННДІОП має забезпечити збирання, обробку й доведення до кожного підприємства незалежно від сфери управління (галузевого чи регіонального рівня) інформації з питань управління та нагляду за охороною праці.

Для зниження ризиків, пов'язаних із виробничим устаткуванням, технологічними процесами, будівлями й спорудами, необхідно:

– переглянути нормативну базу, що регламентує безпечність виробничого устаткування, технологічних процесів, будівель і споруд, привести її у відповідність до вимог директив Європейського Союзу;

– удосконалити порядок проведення експертизи устаткування, технологічних процесів, будівель і споруд на їх відповідність вимогам безпеки з урахуванням міжнародних та європейських норм;

– ужити заходів щодо виведення з експлуатації (поетапно) морально застарілого і фізично зношеного виробничого устаткування, будівель, споруд тощо.

Враховуючи те, що протягом останніх років організація виробництва засобів індивідуального захисту (ЗІЗ) в Україні не дає очікуваних результатів, необхідно докорінно переглянути підхід до вирішення цієї проблеми, використовуючи досвід Білорусі, Литви, Латвії, Росії. Для цього слід упровадити на території України ЗІЗ, які вже отримали відповідний міжнародний сертифікат,

виробників 313 у державі зорієнтувати виключно на ті, впровадження у виробництво яких є економічно вигідним. Доцільно вивчити питання щодо заснування в Україні спільних з іноземними представництвами підприємств із виробництва таких ЗІЗ, які б відповідали вимогам європейських норм і мали відповідні міжнародні сертифікати.

Для вирішення питань, пов'язаних із обліком, аналізом та оцінкою стану умов та безпеки праці, слід:

– опрацювати (удосконалити) і забезпечити впровадження єдиної державної статистичної звітності щодо обліку, аналізу та оцінки стану безпеки й умов праці;

– законодавчо врегулювати звітність щодо обліку, аналізу та оцінки стану безпеки й умов праці підприємств недержавної форми власності;

– надати матеріальну підтримку ННДІОП шляхом включення до державного бюджету витрат, пов'язаних із проведенням обґрунтованого аналізу стану охорони праці, наглядової діяльності та їх взаємозв'язку, опрацюванням періодичних аналітичних матеріалів щодо стану охорони праці в Україні.

Планування робіт з охорони праці має здійснюватися з урахуванням результатів аналізу й оцінки стану охорони праці, визначення пріоритетних напрямів діяльності.

Фінансування робіт з охорони праці. Необхідно створити належне правове підґрунтя і забезпечити фінансування заходів з охорони праці на державному, галузевому і регіональному рівнях за рахунок коштів:

– Фонду соціального страхування від нещасних випадків, виділених на профілактику виробничого травматизму й профзахворювань;

– державного бюджету і місцевих бюджетів - для часткового фінансування (разом із коштами Фонду соціального страхування від нещасних випадків) Національної, галузевих і регіональних програм поліпшення стану безпеки, гігієни праці та виробничого середовища або інших цільових програм з охорони праці, а також заходів з охорони праці, передбачених програмами соціально-економічного і культурного розвитку України та її адміністративно-

територіальних одиниць; при цьому кошти на охорону праці в державному й місцевих бюджетах виділяються окремими рядками;

– інших джерел фінансування, не заборонених законодавством.

Система контролю за витрачанням коштів, виділених на охорону праці на рівні підприємства, має бути вдосконалена таким чином, щоб забезпечити їх спрямування за цільовим призначенням відповідно до Переліку заходів та засобів з охорони праці, що затверджується Кабінетом Міністрів України.

4.2 Аналіз умов праці за показниками шкідливості та небезпечності чинників виробничого середовища

У результаті активної діяльності людини в середовищі існування, воно поволі змінювало свій вигляд, що призвело до порушення біосфери і появи штучного середовища, яке називають техногенним (техносферою). На сьогоднішній день майже все середовище, в якому перебуває людина, є техногенним.

Техногенне середовище (техносфера) як складова навколишнього середовища є похідною діяльності людини, яка виникла як наслідок впливу антропогенних чинників.

Діючи у техногенному середовищі, людина безперервно виконує, як мінімум, два основних завдання:

– забезпечує своє комфортне перебування у середовищі перебування як на робочому місці, так і в побуті;

– створює та використовує системи захисту від його негативних чинників впливу.

До середини XX століття людина ще була неспроможною ініціювати великомасштабні аварії та катастрофи, які б викликали зміни у біосфері. Поява об’єктів ядерної енергетики, потужних хімічних підприємств та висока концентрація їх у певних регіонах зумовили руйнування екосистеми.

Створена руками і розумом людини техніка ніби й була покликана максимально задовольнити її потреби у комфорті та безпеці, але загалом не виправдала сподівань. Біосфера у багатьох регіонах планети активно змінювалася техносферою. Це, у свою чергу, призвело до зниження якості компонентів системи "Людина – Навколишнє середовище" і, перш за все, природного середовища. За прогнозами вчених, цей вплив буде і в подальшому збільшуватися із поглибленням глобалізації світової економіки.

Розрізняють прямий і непрямий вплив на навколишнє середовище та організм людини негативних чинників техносфери. Прямий вплив – це виробничий і побутовий травматизм, професійні захворювання. Непрямий вплив – це погіршення складу повітря, якості води, їжі тощо.

При певних умовах цей негативний вплив може призвести до зростання концентрації домішок у біосфері і погіршення екологічної рівноваги, збільшення кількості захворювань населення та тварин, посилення епідеміологічного неблагополуччя.

Середовище техносфери сучасного існування людини поділяють на побутове та виробниче.

Виробниче середовище – це простір, де людина провадить свою трудову діяльність. До нього належать підприємства, організації, установи, заклади освіти, транспорт, комунікації тощо. Виробниче середовище характеризується певними параметрами його життєздатності і життєдіяльності, специфічними для кожного виробництва. В умовах виробничого середовища на здоров'я людини можуть впливати небезпечні та шкідливі виробничі фактори (НіШВФ).

Деякими з таких факторів є:

- електричний струм;
- рівень шуму;
- рівень вібрації;
- рівень теплового, електромагнітного випромінювань;
- ступінь загазованості, запиленості.

Електричний струм – поширений уражаючий фактор на виробництві та у побуті у зв'язку з широким застосуванням електричних приладів та агрегатів. Працюючи з ними, необхідно дотримуватися правил електробезпеки (організаційні і технічні заходи та засоби, які забезпечують захист людей від шкідливого і небезпечного впливу електричного струму).

Шум – виробничий, побутовий – безпосередньо впливає на якість праці. Довготривала робота у шумному середовищі може призвести до порушення центральної нервової системи і спричинити аварії на виробництві.

Зростання інтенсивності шуму понад природний рівень у людини викликає швидку втомлюваність, зниження розумової активності, а при досягненні 90 – 100 дБ – поступову втрату слуху.

Зокрема, наприклад, шум, що утворюється під час тихої розмови між студентами в умовах навчальної аудиторії, вимірюється в 10 – 12 дБ, що уже шкодить навчальному процесу.

Електромагнітне випромінювання (ЕМВ) – процес утворення вільного електромагнітного поля, яке випромінює прискорено рухомі заряджені частинки, що впливають на середовище і людину в ньому. Джерелами ЕМВ є лінії електропередач, радіо і телебачення, робота деяких промислових і побутових приладів.

Теплове випромінювання – це випромінювання, яке утворюється за рахунок внутрішньої енергії речовини і підвищує температуру середовища. Характеризується наявністю теплового потоку (кількість тепла, яке проходить в одиницю часу через одиницю поверхні); може опекти, спричинити вибух.

Перелічені небезпечні і шкідливі виробничі фактори повинні відповідати певним параметрам, які людина визначає сама, проектуючи і будуючи ті чи інші об'єкти. Межа зміни параметрів повинна гарантувати безпеку, а у деяких випадках — і комфорт трудової діяльності. При цьому функціонування об'єкта загалом повинно бути безпечним. Дія небезпечних і шкідливих виробничих факторів може призвести до травматизму і професійного захворювання людини. Кожні 3 хвилини у світі внаслідок виробничого травматизму чи професійного захворювання помирає людина.

ВИСНОВКИ

У кваліфікаційній роботі спроектовано та реалізовано програмний прототип комп'ютеризованої системи тематичної рубрикації текстових документів. Під документом, у даному випадку, може виступати звичний для пересічного користувача текстовий файл, або, наприклад, повідомлення із соціальних мереж чи лист з електронної пошти. При рубрикації документів повинна бути наявна мітка щодо його приналежності до певної категорії чи теми, оскільки дана задача відноситься до алгоритмів машинного навчання з вчителем.

Перед початком безпосередньої реалізації системи спроектовано архітектуру системи, що визначає основні її компоненти та зв'язки між ними. До її складу входять сховище документів та компонент, що відповідає з рубрикацію документів на основі аналізу їх вмісту.

Компонент рубрикації документів складається з наступних модулів:

– модуль попереднього опрацювання тексту;
– модуль виявлення ознак тексту;
– модуль класифікації документів;

У роботі описано принцип застосування алгоритмів препроцесингу текстових даних до яких відноситься токенізація, лематизація стемінг, стандартизація.

В якості методів для виявлення ознак тексту у документі запропоновано використати різновиди статистичних ознак алгоритму TF-IDF, а також семантична векторизації.

Експериментальні дослідження щодо рубрикації документів виконано на основі моделей Баєсівського класифікатора, методу опорних векторів, логістичної регресії і методів, що базуються на використанні простих і глибоких нейронних мереж. Результати, одержані у процесі експерименту на наборі неструктурованих даних без виконання попереднього опрацювання тексту показали, що найбільшу точність можна досягти при використанні логістичної регресії, яка становить трохи більше 70%.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Барсегян А., Куприянов М. , Степаненко В. , Холод И. Технологии анализа данных. СПб. : Изд-во " БХВ-Петербург". 2008. 384 с.

2. . Барсегян А. А, Куприянов М. С., Холод И. И., Тесс М. Д., Елизаров С. И. Анализ данных и процессов: учеб. пособие . 3-е изд., перераб. и доп. Санкт-Петербург : БХВ-Петербург, 2009. 512 с.

3. Yang Y. A re-examination of text categorization methods. Proc. of Int. ACM Conference on Research and Development in Information Retrieval (SIGIR-99), 1999. P. 42-49.

4. Вагин В. Н., Головина Е. Ю., Загорянская А. А.,. Фомина М. В., Вагин В. Н. Достоверный и правдоподобный вывод в интеллектуальных системах. Москва : Физматлит, 2004. 704 с.

5. Барсегян А. А. Технология анализа данных: Data Mining, Visual Mining, Text Mining, OLAP. Санкт-Петербург : БХВ-Петербург, 2007. 384 с.

6. CJC Burges. A Tutorial on Support Vector Machines for Pattern Recognition URL : http://www.music.mcgill.ca/rfergu/adamTex/references/ Burges98.pdf (дата звернення 17.04.2021 р.).

7. Yang Y. A re-examination of text categorization methods. Proc. SIGIR’2012, 22nd ACM International Conference on Research and Development in Information Retrieval, 2012. P. 42-49.

8. Sebastiani F. Machine learning in automated text categorization / F. Sebastiani. ACM Comput. Surv. March 2010. Vol. 34, No. 1. P. 1-47.

9. Yang Y., Liu X. A re-examination of text categorization methods . Proc. of Int. ACM Conference on Research and Development in Information Retrieval (SIGIR-99), 2007. P. 42-49.

10. Bing L. Sentiment Analysis and Opinion Mining. New Jersey - Morgan & Claypool Publishers, 2012. 167 p.

11. Khurshid A. Affective Computing and Sentiment Analysis: Metaphor, Ontology, Affect and Terminology. Berlin. Springer Science & Business Media, 2011.164 p.

12. .Narayanan, V. Fast and accurate sentiment classification using an enhanced naive bayes model. [Text] / V. Narayanan, I. Arora, A. Bhatia // Intelligent Data Engineering and Automated Learning IDEAL / V. Narayanan, I. Arora, A. Bhatia. – Berlin: Springer, 2013. – (volume 8206 of Lecture Notes in Computer Science). – pp. 194–201.

13. Patel D., Saxena S., Verma T. Sentiment Analysis using Maximum Entropy Algorithm in Big Data. International Journal of Innovative Research in Science, Engineering and Technology. 2016. pp. 8355–8361.

14. Руководство по работе с HTTP в Python. Библиотека requests. URL: https://khashtamov.com/ru/ (дата звернення 29.04.2021 р.).

15. Named-entity Recognition. URL: https://en.wikipedia.org/wiki/Named-entity_recognition (дата звернення 21.04.2021 р.).

16. Фанифатьева А. Д. Автоматический анализ тональности рецензий с использованием библиотеки tensorflow. URL: http://library.eltech.ru/ files/vkr/2017/bakalavri/ (дата звернення 15.05.2021 р.).

17. Convolutional Neural Networks (CNNs / ConvNets). URL: http://cs231n.github.io/neural-networks-1/ (дата звернення 18.05.2021 р.).

18. ДСанПіН 3.3-2.007-98 «Державні санітарні правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин».

19. НПАОП 0.00-1.28-10 «Правила охорони праці під час експлуатації електронно-обчислювальних машин».

20. НАПБ А.01.001-2004 «Правила пожежної безпеки в Україні».

21. Стеблюк М.І. Цивільна оборона та цивільний захист: Підручник. — 2-ге вид., переробл. — К.: Знання, 2010. — 487 с.

22. Тарасова, В.В. Екологічна статистика. [Текст] / В.В.Тарасова. – Київ: «Центр учбової літератури», 2008. – 391с.

Додаток А.

Технічне завдання

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем і програмної інженерії

Кафедра комп'ютерних систем та мереж

"Затверджую"

Завідувач кафедри КС

_____ Осухівська Г.М.

"___" _____2021 р

КОМП'ЮТЕРИЗОВАНА СИСТЕМА ТЕМАТИЧНОЇ РУБРИКАЦІЇ ДОКУМЕНТІВ

ТЕХНІЧНЕ ЗАВДАННЯ

на _*11*_ листках

Вид робіт: Кваліфікаційна робота

На здобуття освітнього ступеня «Бакалавр»

Спеціальність 123 «Комп'ютерна інженерія»

«УЗГОДЖЕНО»	«ВИКОНАВЕЦЬ»
Керівник кваліфікаційної роботи	Студент групи СІс-44
_____ к.т.н., доц. Луцків А.М.	_____ Григораш В.С.
«____» _____ 2021 р.	«____» _____ 2021 р.

Тернопіль 2021

1 Загальні відомості

1.1 Повна назва та її умовне позначення

Повна назва теми кваліфікаційної роботи: «Комп’ютеризована система тематичної рубрикації документів».

Умовне позначення кваліфікаційної роботи: КС КРБ 123.166.00.00

1.2 Виконавець

Студент групи СІс-44, факультету комп’ютерно-інформаційних систем і програмної інженерії, кафедри комп’ютерних систем та мереж, Тернопільського національного технічного університету імені Івана Пулюя, Григораш Вадим Святославович.

1.3 Підстава для виконання роботи

Підставою для виконання кваліфікаційної роботи є наказ по університету (№ 4.7-97 від 10.02.2021 р.)

1.4 Планові терміни початку та завершення роботи

Плановий термін початку виконання кваліфікаційної роботи – 10.02.2021 р.

Плановий термін завершення виконання кваліфікаційної роботи – 20.06.2021 р.

1.5 Порядок оформлення та пред’явлення результатів роботи

Порядок оформлення пояснювальної записки та графічного матеріалу здійснюється у відповідності до чинних норм та правил ІСО, ГОСТ, ЕСКД, ЕСПД та ДСТУ.

Пред’явлення проміжних результатів роботи з виконання кваліфікаційної роботи здійснюється у відповідності до графіку, затвердженого керівником роботи.

Попередній захист кваліфікаційної роботи відбувається при готовності роботи на 90% , наявності пояснювальної записки та графічного матеріалу.

Пред’явлення результатів кваліфікаційної роботи відбувається шляхом захисту на відповідному засіданні ЕК, ілюстрацією основних досягнень за допомогою графічного матеріалу.

2 Призначення і цілі створення системи

2.1 Призначення системи

Основне призначення комп’ютеризованої системи тематичної рубрикації документів полягає в автоматизації процесу визначення вмісту документу і класифікації або кластеризації його у відповідності до теми. Дана система може бути різновидом або частиною більш складних комп’ютерно-інформаційних комплексів з опрацювання текстової інформації.

Застосування комп’ютеризованих систем такого класу стосуються побудови тематичних моделей для сфери добування тексту та пошуку інформації. Проектовану комп’ютеризовану систему можна використовувати при вирішенні задач класифікації, категоризації, узагальнення та сегментації документів.

Комп’ютеризована система тематичної рубрикації документів може також бути ефективним інструментом у галузі комп'ютерного зору, популяційної генетики та соціальних мереж. При пошуку інформації, тематичне моделювання допомагає розширювати критерії запитів, а також персоналізувати результати пошуку або побудови рекомендацій шляхом відображення тих тем і тих документів, які відповідають уподобанням користувачів.

У соціальних науках застосування комп’ютеризованої системи рубрикації документів дає змогу якісно аналізувати настрій користувачів, його емоції та

будувати психологічний портрет людини на основі приналежності документів до певної рубрики.

Окрім цього, комп’ютеризована система рубрикації документів може використовуватись у сфері програмної і комп’ютерної інженерії, що дозволить на основі моделювання тем проводити аналіз вихідного коду, змін журналів, стану баз даних та ін.

2.2 Мета створення системи

Метою побудови комп’ютеризованої системи тематичної рубрикації документів є створення автоматизованого інтелектуального засобу для класифікації текстових документів, які дозволять формувати релевантні та ранжовані списки у відповідності до запиту користувача. Проектована комп’ютеризована система повинна забезпечити ефективність процесу пошуку і рубрикації документів за рахунок зменшення часових затрат, у випадку виконання таких дій людиною.

Мета роботи щодо створення комп’ютеризованої системи рубрикації документів передбачає виконання і розв’язок ряду задач, основними з яких є:

– дослідження сучасних методів та інструментів опрацювання природної мови автоматизованими засобами;

– обґрунтування і побудова моделі розпізнавання контексту у текстових документах;

– реалізація моделі аналізу текстових даних;

– реалізація інтелектуалізованих алгоритмів класифікації і рубрикації документів;

– обґрунтоване застосування метрик для вимірювання подібності документів між собою та приналежності до певної рубрики;

– проведення експериментальних досліджень щодо рубрикації документів;

– скорочення часових ресурсів формування рубрик і категорій документів.

2.3 Характеристика об'єкту

2.3.1 Основні задачі та функції об’єкту

Основними задачами комп’ютеризованої системи тематичної рубрикації документів є:

– опрацювання корпусу текстових документів засобами автоматизованого аналізу;

– препроцесинг текстової інформації;

– виявлення інформативних ознак документів для встановлення їх приналежності до певної рубрики або категорії;

– можливість навчання і проведення процедур тестування тематичної рубрикації документів;

– класифікація та рубрикація документів;

– ранжування документів у межах заданих рубрик;

– формування адекватних відповідей у вигляді ранжованого списку документів на запит користувача, що стосується певної тематики корпусу документів;

– формування анотацій документів;

– можливість формування ключових слів за документами;

– формування додаткових рекомендацій документів відносно запиту користувача;

– забезпечення точності рубрикації документів з точністю не нижче за 70%;

– формування кількісних критеріїв щодо якості та ефективності тематичної рубрикації корпусу документів;

– можливість формування розподілу за рубриками у наявному корпусі документів.

До найбільш важливих функцій комп’ютеризованої системи тематичної рубрикації документів належить автоматизація процесу класифікації документів на основі аналізу їх вмісту із застосуванням алгоритмів машинного навчання.

3 Вимоги до системи

3.1 Вимоги до системи в цілому

Загальні вимоги до комп’ютеризованої системи тематичної рубрикації документів пов’язані із поставленими у кваліфікаційній роботі задачами і передбачають виконання усіх етапів, характерних для опрацювання природної мови.

Комп’ютеризована система в цілому повинна розв’язувати задачі класифікації документів за наперед визначеними категоріями, або як альтернативну виконувати їхню кластеризацію. Окрім цього, важливою вимогою до системи є формування анотацій документів у зрозумілій для людини формі на основі виявлених ключових слів або фраз.

Рубрикація за темами документів повинна відбуватись на відкритому наборі даних, який може бути попередньо розмічений. Розбиття на навчальну і тестові вибірки повинні бути або рівномірними, або складати максимальне співвідношення 80:20.

Архітектурна композиція комп’ютеризованої системи автоматичної тематичної рубрикації текстових документів повинна передбачати використання технології клієнт-сервер, як на рівні апаратного забезпечення, так і на рівні програмного. Це забезпечить комунікацію між програмним забезпечення користувача, яка в даному випадку виступає в якості терміналу, та сервером, що зберігає корпус документів.

При програмній реалізації системи автоматичної тематичної рубрикації доцільно використовувати середовища з підтримкою мов програмування у галузі інтелектуального аналізу даних, зокрема Python.

В цілому, набір основних вимог до комп’ютеризованої системи можна сформулювати наступним чином:

– можливість зчитування тексту з корпусу документів;
– забезпечення можливості векторного представлення документу;

– можливість застосування метрик для виявлення подібності між документами;

– здатність аналізувати контекст документу та визначати ключові слова;

– можливість визначення теми документу на основі навчальної вибірки;

– здатність одночасного віднесення документу до кілько категорій;

– здатність комп’ютеризованої системи взаємодіяти з іншими системами;

– забезпечення точності рубрикації документів на рівні не менше, ніж 70%.

3.1.1 Вимоги до структури та функціонування системи

Вимогами щодо функціонування комп’ютеризованої системи тематичної рубрикації документів, з врахуванням особливостей структурних компонентів, є забезпечення зв’язку прикладного додатку з джерелом даних, що містить корпус документів, які необхідно прокласифікувати, або вказати приналежність до певної наперед визначеної категорії.

Кожен документ повинен бути доступним для зчитування його вмісту, а також мати унікальний ідентифікатор та володіти інформативним метаописом. В метаописі можлива наявність інформації про кількість слів, розмір документу, автора, дати створення та інших, що є важливими з позиції його комплексного опису.

Працездатність комп’ютеризованої системи повинна забезпечуватись двома важливими функціональними блоками:

– сховище корпусу документів даних, яка може бути сформована у вигляді база даних текстової інформації;

– інтелектуальна компонента, що безпосередньо виконує тематичну рубрикацію документів.

Інтелектуальну складову комп’ютеризованої системи рекомендовано реалізувати на основі таких відкритих бібліотек:

– Scikit-learn – бібліотека з відкритим кодом, що володіє простими та ефективними інструментами для прогнозування і аналізу даних різної природи D;

– TextBlob – це бібліотека для обробки текстових даних, що забезпечує простий API для опрацювання природної мови, зокрема визначення частини мови, виділення іменникових фраз, аналіз настроїв, класифікації, перекладу тощо.

– Pandas – швидкий, потужний, гнучкий та простий у використанні інструмент аналізу та маніпулювання даними з відкритим кодом, який побудований поверх мови програмування Python;

– Keras – бібліотека, що пропонує послідовні та прості API, мінімізує кількість дій користувача, необхідних для загальних випадків використання, а також забезпечує чіткі повідомлення про помилки;

– XGBoost – це оптимізована, розподілена градієнтно-підсилювальна бібліотека, розроблена для забезпечення високої ефективності, гнучкості та портативності та реалізації алгоритмів машинного навчання в рамках Gradient Boosting.

Система тематичної рубрикації документів має показувати стійкі результати щодо визначення приналежності документу до певної категорії, а також формувати анотацію з використанням ключових слів.

3.1.2 Вимоги до способів та засобів зв’язку між компонентами системи

Способи і засоби зв’язку між компонентами комп’ютеризованої системи тематичної рубрикації документів можна поділити на два типи: локальне навчання моделі тематичного моделювання та віддалене тестування працездатності системи.

При локальному тематичному моделюванні використовується частина корпусу текстових документів для проведення експериментальних досліджень щодо вибору оптимальної моделі рубрикації документів.

У випадку віддаленого тестування та експлуатації комп’ютеризованої системи корпус документів може бути розміщений на серверах мережі Інтернет або у хмарних сховищах. Протокол передачі та обміну даними, який при цьому використовується – HTTP/HTTPS.

3.1.3 Вимоги по діагностуванню системи

Особливих вимог щодо проведення діагностики комп’ютеризованої системи тематичної рубрикації документів не висувається, однак вона повинна бути виконана у відповідності до затвердженого розкладу. Крім того, має проводитись також регулярна перевірка наповнення сховища даних текстовими документами, фіксації зміни їх локації, а також тестування зв’язку між клієнтом і сервером.

При виникненні збоїв, у випадку віддаленого зв’язку між інтелектуальним сервісом і сховищем документів, необхідно провести діагностику каналів передачі даних та усунути проблему.

3.1.4 Перспективи розвитку, модернізація системи

До перспектив розвитку комп’ютеризованої системи тематичної рубрикації документів належить можливість її адаптації та інтеграції із пошуковими інтелектуальними системами, системами управління архівами документів у різних сферах, зокрема електронних бібліотек та ряду інших. Окрім цього, до системи автоматизованої рубрикації можна додавати інтерфейси користувача в залежності від його потреб, тобто сама система повинна реалізовувати так звану back end логіку.

Шляхами модернізації проектованої системи є можливість внесення змін для підвищення продуктивності виконання операцій щодо класифікації і кластеризації текстових документів, формування контейнерів із відповідними залежностями між бібліотеками у вигляді контейнерів, що забезпечить її кросплатформність та стійкість функціонування у середовищі кінцевого користувача.

3.1.5 Вимоги до надійності системи

Комп’ютеризована система тематичної рубрикації документів повинна відповідати вимогам надійності, які висуваються до такого класу систем, володіти інструментами авторизації користувачів на рівні програмного забезпечення клієнта і сервера, на якому розміщено сховище документів, а також характеризуватися здатністю до відновлюваності при виникнення аварійних ситуацій.

Надійність комп’ютеризованої системи тематичної рубрикації повинна відповідати наступним критеріям:

– сталість функціонування та формування результатів класифікації документів протягом визначеного періоду часу, зазвичай 8-12 год./день (період робочого дня підприємства);

– робастність алгоритмів машинного навчання при рубрикації документів;

– відповідність вимогам часу напрацювання на відмови, тобто час безперервного функціонування не менше, ніж 100000 год.;

– можливість швидкого усунення збоїв у роботи комп’ютеризованої системи рубрикації документів;

– наявність механізмів забезпечення захисту даних на програмному та апаратного рівні;

– наявність інструментів управління та контролю за виконанням запитів користувачів;

– здатність підтримувати стабільність зв’язку із зовнішніми системами.

3.1.6 Вимоги до функцій та задач, які виконує система

Вимоги і задачі відносно проектування функцій комп’ютеризованої системи тематичної рубрикації документів стосуються :

– можливості аналізу вмісту документа та проведення токенізації тексту;

– здатності виявлення і видалення шумів з документу;

– можливість проведення лематизації і стеммінгу;

– здатності проведення процедур стандартизації текстових даних;

– можливість виявлення сутностей у документів;

– здатності формувати контекст документу;

– можливості формування статистичних показників документу;

– здатності визначення інформативних ознак тексту у корпусі документу;

– можливості класифікації документів із визначеною точністю і достовірністю приналежності до визначених категорій.

3.1.7 Вимоги до апаратного забезпечення

При моделюванні та експлуатації комп’ютеризованої системи тематичної рубрикації документів до апаратного забезпечення висуваються такі рекомендовані вимоги:

– процесор – Intel Core i5 4300M з частотою 2,2 ГГц або 2,3 ГГц (1 сокет, 8 ядер, 2 потоки на ядро);

– об’єм оперативної пам’яті – 16 ГБ.

Альтернативними конфігураціями при реалізації рекомендаційної системи є:

– Xeon E5-2498 v3 з частотою 1,8 ГГц (1 сокет, 10 ядер кожен, 2 потік на ядро, або Xeon Phi 7210 з частотою 1,3 ГГц (1 сокет, 64 ядра, 4 потоки на ядро);

– 32 ГБ або 64 ГБ об’єм оперативної пам’яті;

– накопичувач (жорсткий диск) з наявністю не менше, ніж 2-3 ГБ простору.

3.1.8 Вимоги до програмного забезпечення

Операційні системи для ефективного роботи системи тематичної рубрикації можуть бути будь-якого типу (Windows, Linux, MacOS) однак повинні підтримувати мову програмування Python відповідної версії, а також перелік бібліотек, які описані у пункті структури та функціонування комп’ютеризованої системи.

4 Вимоги до документації

Документація повинна відповідати вимогам ЄСКД та ДСТУ

Комплект документації повинен складатись з:

– пояснювальної записки;

– графічного матеріалу:

1. Кластеризація, тематичне моделювання і класифікація текстової інформації.
2. Архітектура комп’ютеризованої системи тематичної рубрикації документів.
3. Схема препроцесингу текстових документів.
4. Архітектура комп’ютеризованої системи тематичної рубрикації документів.
5. Алгоритм тематичної рубрикації документів

6. Результати тематичної рубрикації текстових документів.

*Примітка: У комплект документації можуть вноситися міни та доповнення в процесі розробки.

5 Стадії та етапи проектування

Таблиця 1 – Стадії та етапи виконання кваліфікаційної роботи бакалавра

№ етапу	Назва етапу виконання кваліфікаційної роботи	Термін виконання
1.	Розробка та аналіз технічного завдання	10.02-19.02.2021
2.	Аналіз моделей і сфер застосування тематичного моделювання	19.02-05.03.2021
3.	Проектування структури і компонентів комп'ютеризованої системи тематичної рубрикації документів	05.03-26.03.2021
4.	Методи та інструменти реалізації систем тематичної рубрикації документів	26.03-01.04.2021
5.	Програмна реалізація комп'ютеризованої системи тематичної рубрикації документів	01.04-22.04.2021
6.	Розробка інструкцій із встановлення та налаштування параметрів комп'ютеризованої системи	22.04-10.05.2021
7.	Безпека життєдіяльності, основи охорони праці	10.05-18.05.2021
8.	Оформлення кваліфікаційної роботи	18.05-06.06.2021
9.	Попередній захист кваліфікаційної роботи	06.06-18.06.2021
10.	Захист кваліфікаційної роботи	22.06-27.06.2021

6 Додаткові умови виконання кваліфікаційної роботи

Під час виконання кваліфікаційної роботи у дане технічне завдання можуть вноситися зміни та доповнення.

Додаток Б.

Реалізація додаткових моделей тематичної рубрикації документів

Лістинг Б.1 – Модель XGBoost

```
    # Extereme Gradient Boosting on Count Vectors
    accuracy = train_model(xgboost.XGBClassifier(),
xtrain_count.tocsc(), train_y, xvalid_count.tocsc())
    print "Xgb, Count Vectors: ", accuracy

    # Extereme Gradient Boosting on Word Level TF IDF Vectors
    accuracy = train_model(xgboost.XGBClassifier(),
xtrain_tfidf.tocsc(), train_y, xvalid_tfidf.tocsc())
    print "Xgb, WordLevel TF-IDF: ", accuracy

    # Extereme Gradient Boosting on Character Level TF IDF Vectors
    accuracy = train_model(xgboost.XGBClassifier(),
xtrain_tfidf_ngram_chars.tocsc(), train_y,
xvalid_tfidf_ngram_chars.tocsc())
    print "Xgb, CharLevel Vectors: ", accuracy
```

Результат виконання лістингу Б.1

```
Xgb, Count Vectors:  0.6324
Xgb, WordLevel TF-IDF:  0.6364
Xgb, CharLevel Vectors:  0.6548
```

Лістинг Б.2 – Модель на основі рекурентної НМ з LSTM

```
# Add an Input Layer
    input_layer = layers.Input((70, ))

    # Add the word embedding Layer
    embedding_layer = layers.Embedding(len(word_index) + 1, 300,
weights=[embedding_matrix], trainable=False)(input_layer)
```

```
        embedding_layer =
layers.SpatialDropout1D(0.3)(embedding_layer)

        # Add the LSTM Layer
        lstm_layer = layers.LSTM(100)(embedding_layer)

        # Add the output Layers
        output_layer1 = layers.Dense(50,
activation="relu")(lstm_layer)
        output_layer1 = layers.Dropout(0.25)(output_layer1)
        output_layer2 = layers.Dense(1,
activation="sigmoid")(output_layer1)

        # Compile the model
        model = models.Model(inputs=input_layer,
outputs=output_layer2)
        model.compile(optimizer=optimizers.Adam(),
loss='binary_crossentropy')

        return model

    classifier = create_rnn_lstm()
    accuracy = train_model(classifier, train_seq_x, train_y,
valid_seq_x, is_neural_net=True)
    print "RNN-LSTM, Word Embeddings",  accuracy
```

Результат виконання лістингу Б2

```
Epoch 1/1
7500/7500 [==============================]
- 22s 3ms/step - loss: 0.6899
RNN-LSTM, Word Embeddings 0.5124
```

Лістинг Б.3 – Модель RNN з GRU

```
    # Add an Input Layer
        input_layer = layers.Input((70, ))

        # Add the word embedding Layer
        embedding_layer = layers.Embedding(len(word_index) + 1, 300,
weights=[embedding_matrix], trainable=False)(input_layer)
        embedding_layer =
layers.SpatialDropout1D(0.3)(embedding_layer)

        # Add the GRU Layer
        lstm_layer = layers.GRU(100)(embedding_layer)

        # Add the output Layers
        output_layer1 = layers.Dense(50,
activation="relu")(lstm_layer)
        output_layer1 = layers.Dropout(0.25)(output_layer1)
        output_layer2 = layers.Dense(1,
activation="sigmoid")(output_layer1)

        # Compile the model
        model = models.Model(inputs=input_layer,
outputs=output_layer2)
        model.compile(optimizer=optimizers.Adam(),
loss='binary_crossentropy')

        return model

    classifier = create_rnn_gru()
    accuracy = train_model(classifier, train_seq_x, train_y,
valid_seq_x, is_neural_net=True)
    print "RNN-GRU, Word Embeddings",  accuracy
```

Результат виконання лістингу Б.3

```
Epoch 1/1
7500/7500 [==============================]
 - 19s 3ms/step - loss: 0.6898
RNN-GRU, Word Embeddings 0.5124
```

Лістинг Б.4 – Двлнаправлена RNN

```
# Add an Input Layer
    input_layer = layers.Input((70, ))

    # Add the word embedding Layer
    embedding_layer = layers.Embedding(len(word_index) + 1, 300,
weights=[embedding_matrix], trainable=False)(input_layer)
    embedding_layer =
layers.SpatialDropout1D(0.3)(embedding_layer)

    # Add the LSTM Layer
    lstm_layer =
layers.Bidirectional(layers.GRU(100))(embedding_layer)

    # Add the output Layers
    output_layer1 = layers.Dense(50,
activation="relu")(lstm_layer)
    output_layer1 = layers.Dropout(0.25)(output_layer1)
    output_layer2 = layers.Dense(1,
activation="sigmoid")(output_layer1)

    # Compile the model
    model = models.Model(inputs=input_layer,
outputs=output_layer2)
    model.compile(optimizer=optimizers.Adam(),
loss='binary_crossentropy')

    return model
```

```
    classifier = create_bidirectional_rnn()
    accuracy = train_model(classifier, train_seq_x, train_y,
valid_seq_x, is_neural_net=True)
    print "RNN-Bidirectional, Word Embeddings",  accuracy
```

Результат виконання лістингу Б.4

```
    Epoch 1/1
    7500/7500 [==============================]
     - 32s 4ms/step - loss: 0.6889
    RNN-Bidirectional, Word Embeddings 0.5124
```

Лістинг Б.5 – Модель RCNN

```
    # Add an Input Layer
        input_layer = layers.Input((70, ))

        # Add the word embedding Layer
        embedding_layer = layers.Embedding(len(word_index) + 1, 300,
weights=[embedding_matrix], trainable=False)(input_layer)
        embedding_layer =
layers.SpatialDropout1D(0.3)(embedding_layer)

        # Add the recurrent layer
        rnn_layer = layers.Bidirectional(layers.GRU(50,
return_sequences=True))(embedding_layer)

        # Add the convolutional Layer
        conv_layer = layers.Convolution1D(100, 3,
activation="relu")(embedding_layer)

        # Add the pooling Layer
        pooling_layer = layers.GlobalMaxPool1D()(conv_layer)

        # Add the output Layers
```

```
        output_layer1 = layers.Dense(50,
activation="relu")(pooling_layer)
        output_layer1 = layers.Dropout(0.25)(output_layer1)
        output_layer2 = layers.Dense(1,
activation="sigmoid")(output_layer1)

        # Compile the model
        model = models.Model(inputs=input_layer,
outputs=output_layer2)
        model.compile(optimizer=optimizers.Adam(),
loss='binary_crossentropy')

        return model

    classifier = create_rcnn()
    accuracy = train_model(classifier, train_seq_x, train_y,
valid_seq_x, is_neural_net=True)
    print "CNN, Word Embeddings",  accuracy
```

Результат виконання лістингу Б.5

```
Epoch 1/1
7500/7500 [==============================]
 - 11s 1ms/step - loss: 0.6902
CNN, Word Embeddings 0.5124
```