Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет комп'ютерної інженерії та управління
(повна назва)

Кафедра електронних обчислювальних машин
(повна назва)

# КВАЛІФІКАЦІЙНА РОБОТА

## Пояснювальна записка

Рівень вищої освіти другий (магістерський)

Концептуальна модель двохступеневої класифікації тексту

(тема)

Виконав:
студент II курсу, групи КСМм-21-1
Настенко О.С.
(прізвище, ініціали)

Спеціальність
123 «Комп'ютерна інженерія»
(код і повна назва спеціальності)

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

Освітня програма
Комп'ютерні системи та мережі
(повна назва освітньої програми)

Керівник: доц. Барковська О.Ю.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри ЕОМ _____ Коваленко А.А.
(підпис) (прізвище, ініціали)

2022 р.

Харківський національний університет радіоелектроніки

Факультет комп’ютерної інженерії та управління

Кафедра електронних обчислювальних машин

Рівень вищої освіти другий (магістерський)

Спеціальність 123 «Комп’ютерна інженерія»
(код і повна назва)

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

Освітня програма Комп’ютерні системи та мережі
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
“____” _____ 20__ р.

# ЗАВДАННЯ

## НА КВАЛІФІКАЦІЙНУ РОБОТУ

студенту Настенко Олександру Сергійовичу
(прізвище, ім’я, по батькові)

1. Тема роботи Концептуальна модель двохступеневої класифікації тексту

затверджена наказом по університету від “ 07 ” листопада 2022 р. № 1453 Ст

2. Термін подання студентом роботи до екзаменаційної комісії 13 грудня 2022 р.

3. Вхідні дані до роботи

1) документація мови програмування Python;
2) методи класифікації;
3) набори текстових даних.

4. Перелік питань, що потрібно опрацювати у роботі

1) аналіз існуючих методів класифікації;
2) опис основних етапів роботи алгоритмів класифікації текстів;
3) сильні та слабкі сторони існуючих методів класифікації;
4) встановлення найбільш оптимального засобу, для класифікації тексту;
5) реалізація концептуальної моделі двохступеневої класифікації тексту;
6) проведення експериментальних досліджень;
7) висновки.

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (слайдів)

Слайд-презентація – 11 слайдів

6. Консультанти розділів роботи (заповнюється за наявності консультантів згідно з наказом, зазначеним у п.1 )

| Найменування розділу | Консультант (посада, прізвище, ім'я, по батькові) | Позначка консультанта про виконання розділу | |
|---|---|---|---|
| | | підпис | дата |
| | | | |
| | | | |

## КАЛЕНДАРНИЙ ПЛАН

| № | Назва етапів роботи | Термін виконання етапів роботи | Примітка |
|---|---|---|---|
| 1 | Аналіз існуючих методів класифікації | 08.11.22-11.11.22 | |
| 2 | Вибір та обгрунтування методики дослідження | 12.11.22-17.11.22 | |
| 3 | Вибір інструментальних засобів | 18.11.22-21.11.22 | |
| 4 | Розробка концептуальної моделі класифікації | 22.11.22-28.11.22 | |
| 5 | Проведення експериментів | 29.11.22-02.12.22 | |
| 6 | Оформлення матеріалів кваліфікаційної роботи | 03.12.22-06.12.22 | |
| 7 | Подання кваліфікаційної роботи керівникові та | 07.12.22-08.12.22 | |
| | її попередній захист | | |
| 8 | Подання кваліфікаційної роботи на рецензу- | 09.12.22-12.12.22 | |
| | вання | | |
| | | | |
| | | | |

Дата видачі завдання 07 листопада 2022 р.

Студент _____
(підпис)

Керівник роботи _____ доц. Барковська О.Ю.
(підпис) (посада, прізвище, ініціали)

# РЕФЕРАТ

Пояснювальна записка кваліфікаційної роботи: 58 с., 9 рис., 4 табл., 1 дод., 20 джерел.

КОНЦЕПТУАЛЬНА МОДЕЛЬ,ДВОХСТУПЕНЕВА КЛАСИФІКАЦІЯ, ОБРОБКА ТЕКСТУ, MACHINE LEARNING, PYTHON, BAG OF WORDS, TF-IDF.

Метою кваліфікаційної роботи є створення концептуальної моделі двохступеневоі класифікації тексту. Основною особливістю розробленої концептуальної моделі є використання двохступеневого підходу на основі TF-IDF та C-TF-IDF.

У ході виконання кваліфікаційної роботи були проаналізовані переваги та недоліки існуючих методів автоматичної класифікації тексту. Розроблена концептуальна модель та проведені експерименти.

Для проведення експериментів використовувалась мова програмування Python та бібліотека Scikit-learn. В якості наборів текстових даних використовувалися колекції Reuters-21578, NSF та MiniNg20.

# ABSTRACT

Master’s thesis: 58 pages, 9 figures, 4 tables, 1 appendice, 20 sources.

CONCEPTUAL MODEL, TWO-STAGE CLASSIFICATION, WORD PROCESSING, MACHINE LEARNING, PYTHON, BAG OF WORDS, TF-IDF.

The major goal of this thesis is to create a conceptual model of two-stage text classification. The main feature of the developed conceptual model is the use of a two-step approach based on TF-IDF and C-TF-IDF.

In the course of the qualification work, the advantages and disadvantages of existing methods of automatic text classification were analyzed. A conceptual model was developed and experiments were conducted.

The Python programming language and the Scikit-learn library were used for conducting experiments. The Reuters-21578, NSF, and MiniNg20 collections were used as text datasets.

ЗМІСТ

# ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

AW – метод розглядає всі слова (англ., All words)

AWK – метод розглядає всі слова з виділенням ключових слів (англ., All words with keyword selection)

AWP – метод розглядає всі слова з обрізкою (англ., All words with pruning)

AWPK – метод розглядає всі слова зі скороченням і виділенням ключових слів (англ., All words with pruning and keyword selection)

IDF – зворотна частота документа (англ., inverse document frequency)

MacroF – макросередня F-міра (англ., Macro-averaged F-measure)

MicroF – мікро-усереднена F-міра (англ., Micro-averaged F-measure)

ML – машинне навчання (англ., Machine Learning)

NLP – обробка природної мови (англ., Natural Language Processing)

NSF – Тези доповідей Національного наукового фонду за дослідження (англ., National Science Foundation Research Award Abstracts)

PL – рівень обрізки (англ., Pruning level)

SVM – метод опорних векторів (англ., Support Vector Machine)

TF – частота слова (англ., term frequency)

# ВСТУП

На неструктуровані дані припадає понад 80% усіх даних, причому текст є однією з найпоширеніших категорій. Оскільки аналіз, розуміння, організація та сортування текстових даних є складними та трудомісткими через їх безладний характер, більшість компаній не використовують їх у повній мірі, незважаючи на всі потенційні вигоди, які вони можуть принести.

Тут вступають в гру машинне навчання і класифікація тексту. Компанії можуть використовувати текстові класифікатори для швидкої та економічної організації всіх типів релевантних даних, включаючи електронні листи, юридичні документи, соціальні мережі, чат-боти, опитування тощо.

Класифікація тексту – це процес класифікації або категоризації необроблених текстів за заздалегідь визначеними групами. Іншими словами, це явище маркування неструктурованих текстів відповідними тегами, які передбачаються з набору заздалегідь визначених категорій. Наприклад, класифікація тексту використовується для фільтрації небажаних листів. Це використовується постачальниками послуг електронної пошти, такими як Alphabet та Microsoft, для своїх служб електронної пошти.

Сьогодні класифікація тексту використовується з широким спектром цифрових сервісів для визначення настрою клієнтів, аналізу виступів політичних лідерів і підприємців, моніторингу ненависті і знущань в соціальних мережах і багато чого іншого. Спочатку ці завдання виконувались вручну, але поширення інтернету та масштаб даних призвели до того, що організації використовували моделі класифікації тексту для безперебійного ведення своїх бізнес-операцій.

Метою даної кваліфікаційної роботи є аналіз і порівняння існуючих алгоритмів класифікації, та розробка концептуальної моделі двоступеневої класифікації тексту.

# 1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

## 1.1 Актуальність класифікації тексту

З ростом текстової інформації в електронному вигляді завдання автоматичної класифікації текстів набуває все більшої актуальності. Наприклад, таке завдання виникає при автоматичній обробці новинного потоку і розподілі текстів-новин по каталогах. Для зручності користувачів каталоги організовані в ієрархічну структуру: каталог складається з декількох підкаталогів і т. д.

Особливу важливість завдання класифікації відіграє в науковій сфері, де в кожній дисципліні щорічно додаються десятки тисяч монографій, статей, препринтів та інших видів публікацій. Ефективна обробка таких масивів, якість пошуку в них матеріалів, релевантних конкретному напрямку досліджень, вимагають точного співвіднесення кожної публікації з її тематичною категорією [1].

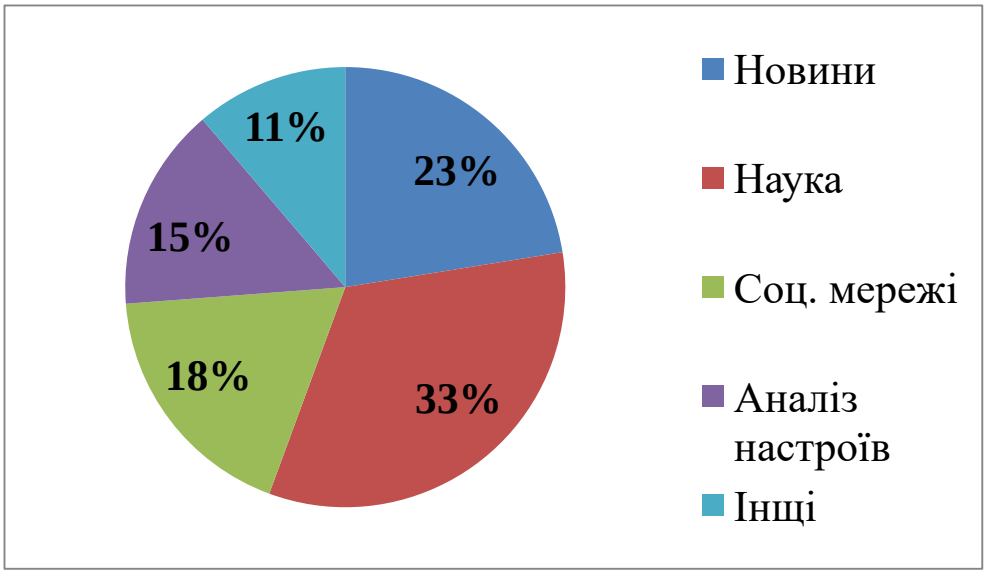


Рисунок 1.1 – Практичні сфери застосування класифікаторів текстових масивів

Після перетворення документів у векторний вигляд, дана задача може вирішуватися методами машинного навчання. Зараз TextMining активно розвивається: ведуться дослідження, запускаються проекти і конкурси на виявлення кращих за точністю алгоритмів. Проте, якість алгоритмів автоматичної класифікації залишають бажати кращого.

Для вирішення завдання класифікації текстових документів застосовується безліч різних методів [2]. Широко використовуваним є метод k-найближчих сусідів і його модифікації, де класифікований об'єкт відноситься до того класу, якому належать найближчі до нього об'єкти навчальної вибірки. Іншим алгоритмом є байєсівська класифікація, яка працює на обчислення апостеріорних ймовірностей класів. Представником лінійних класифікаторів є метод опорних векторів, який полягає в побудові гіперплощини, що розділяє об'єкти вибірки найбільш оптимальним способом. Останнім часом для вирішення завдання класифікації все частіше застосовуються нейронні мережі. В середньому точність різних алгоритмів класифікації текстової інформації варіюється від 70% до 90% і залежить не тільки від алгоритмів класифікації, але і від якості первинних даних.

## 1.2 Огляд існуючих рішень. Аналіз переваг та недоліків

Безліч існуючих зараз методів класифікації текстів базується на термінологічній близькості. Текст представляється у вигляді вектора в евклідовому просторі, де осі координат – це терми, *n*-грами або лексеми, що виділяються з тексту, а координатою по осі є статистична інформація про них. Таким чином, текст може бути представлений у вигляді частотних векторів виникнення слів на основі схем TF, TF*IDF, TF*CHI та інших.

Іншим важливим параметром у класифікації текстів є міра близькості, яка обчислюється між векторами. При цьому її вибір має вплив на якість класифікації. Відомими метриками є: відстань Евкліда, відстань Мінковського, коефіцієнт Отіаі, коефіцієнт Жаккара, проекційна відстань та ін.

Розглянемо докладніше основні методи, що застосовуються при класифікації текстів. Ці методи стосуються методів машинного навчання з учителем.

До метричних методів класифікації відноситься метод *k*-найближчих сусідів, де класифікований об'єкт відноситься до того класу, якому належать найближчі до нього об'єкти навчальної вибірки. У класичного алгоритму k-найближчих сусідів є безліч модифікацій. Це пов'язано з високою обчислювальною складністю алгоритму та низькою швидкістю класифікації. В одному з досліджень наведено порівняння результатів класифікації текстів університету Фудань п'ятьма методами: класичним методом *k*-найближчих сусідів, *k* зважених найближчих сусідів, нечітким методом *k*-найближчих сусідів, методом *k*-найближчих сусідів, заснованому на теорії Демпстера-Шафера, і *k*-найближчих сусідів, заснованому на нечіткому інтегралі [3]. Показано, що найкращу точність, 86%, показує алгоритм, заснований на нечіткому інтегралі, тоді як точність класичним алгоритмом *k*-найближчих сусідів становить лише 78%.

Іншою групою класифікаторів є імовірнісні. Широко використовуваним алгоритмом, що належить до цього класу, є наївна байєсівська класифікація. Вона представляє найбільш просту варіацію байєсівських класифікаторів – наївний байєсівський класифікатор, заснований на припущенні про незалежність ознак. Оскільки класичний підхід до наївної байєсівської класифікації часто не включає ваги вивчених ознак в оцінці умовної ймовірності, Liangxiao Jiang та співавтори у своєму дослідженні пропонують наївну байєсівську класифікацію з глибоким зважуванням ознак, яка обчислює зважені характеристики за частотами на основі навчальних даних, а потім ці ваги враховуються при розрахунку ймовірності [4]. У роботі наївна байєсівська класифікація застосовується при визначенні авторства текстів. Залежно від подання тексту, наприклад, у вигляді *n*-грам, точність методу в застосуванні до цього завдання показала результати від 40 % (при триграммах і тетраграммах) до 96,67% (при термах).

Робота виявила проблему в процесі оцінки параметрів, яка може впливати на точність наївної байєсівської класифікації текстової інформації. Для її усунення автори пропонують проводити для кожного документа нормалізацію тексту і використовувати метод зважування ознак. Для підвищення продуктивності методу наївною байєсівської класифікації також використовується метод допоміжних функцій, між словами розраховується відстань Кульбака-Лейблера, будують наївні байесовские дерева, проводять поліноміальну наївну байєсівську класифікацію, наївну байєсівську класифікацію Бернуллі, гаусову наївну байєсівську класифікацію та ін. У дослідженні показано, що поліноміальна наївна байєсівська класифікація дає кращий результат при класифікації текстів (хоча її точність становить лише 73,4%), ніж наївна байєсівська класифікація Бернуллі (її точність – 69,15 %). При порівнянні трьох методів, заснованих на наївній байєсівській класифікації, показано, що наївна байєсівська класифікації Бернуллі порівнянна за результатами з класичною, тоді як гауссовий наївний байєсівський класифікатор дає найкращу точність класифікації.

Одним з представників лінійних класифікаторів є метод опорних векторів, який полягає в побудові гіперплощини, що розділяє об'єкти вибірки найбільш оптимальним способом.

Також існує класифікація, що базується на методах теорії графів. До неї відноситься, наприклад, метод «випадковий ліс» (random forest). Він полягає в побудові ансамблю паралельно навчаючихся незалежних дерев рішень [5]. У ряді досліджень наводяться способи поліпшення роботи методу випадкового лісу. Так для вирішення багатокласових задач для обчислення ваг об'єктів пропонується використовувати метод XI-квадратів [6]. Завдяки новому методу зважування ознак для вибірки підпростору і методу вибору дерева ефективно зменшується розмір підпростору і підвищується продуктивність класифікації. Залежно від масиву даних метод може проявляти точність класифікації від 72% до 92 %. Алгоритм випадкового лісу

з урахуванням семантики на деревах різного розміру показує точність 73 – 78%, тоді як точність класичного алгоритму становить 57 – 60% [7].

Останнім часом для вирішення завдання класифікації все частіше застосовуються нейронні мережі. У своїй роботі Siwei Lai і співавтори для вирішення завдання класифікації текстів пропонують використовувати рекурентні згорткові нейронні мережі [8]. Автори приходять до висновку, що застосування нейронних мереж при класифікації текстових документів допоможе уникнути проблеми розрідженості даних, а також зібрати більше контекстуальної інформації про сутності в порівнянні з традиційними методами. Згорткові нейронні мережі показали високу точність (83,98%) і при класифікації патентних документів.

Таблиця 1.1 – Методи класифікації

| Методи | Точність | Повнота | Обчислювальна складність алгоритму | Швидкість класифікації |
|---|---|---|---|---|
| Метод *k*-найближчих сусідів | 78% – 86% | 86% – 91% | висока | низька |
| Метод опорних векторів | 63% – 90% | 83% – 87% | низька | низька |
| Наївна байєсівська класифікація | 40% – 83% | 80% – 90% | низька | низька |
| «Випадковий ліс» | 57% – 78% | 75% – 82% | висока | висока |
| Згорткові нейронні мережі | 83,98% | 70% – 85% | висока | висока |

Існує безліч робіт, спрямованих на порівняння точності класифікації текстових документів різними методами. Так, при порівнянні трьох методів: *k*-найближчих сусідів, на основі нечіткого інтеграла, методу опорних векторів і байєсівської класифікації – найкращу точність, 90%, показує метод опорних векторів. Під час класифікації твітів турецькою мовою методи показували різні результати класифікації залежно від розміру навчальної вибірки. Найкращі результати, від 63% до 83%, у всіх трьох випадках демонструвала байєсівська класифікація. При класифікації книг найкращу точність, 81 %, також показує байєсівський класифікатор. Але при класифікації індійських і англійських твітів, незважаючи на те що байєсівська класифікація була найефективнішою, її точність не перевищувала 63%. Дослідження використовує п'ять класифікаторів для класифікації даних з веб-сайтів новин: *k*-найближчі сусіди, випадковий ліс, поліноміальний наївний байєсівський класифікатор, логістична регресія та метод підтримки векторів. Найефективнішим алгоритмом виявився метод опорних векторів, який продемонстрував не тільки високу точність в 91%, але і найшвидший час роботи: мінімум в півтора рази нижче, ніж у інших досліджуваних алгоритмів [9].

## 1.3 Обґрунтування доцільності вдосконалення існуючих рішень

Для поліпшення точності класифікації використовують і комбінації різних алгоритмів класифікації. Наприклад, комбінація алгоритмів *k*-найближчих сусідів і методу опорних векторів робить точність класифікації вище на 1 – 2%, ніж при застосуванні цих класифікаторів окремо. Комбінація методів до-найближчих сусідів, алгоритму Роккіо і методу найменших квадратів зменшило число помилок класифікації на 15 %.

Таким чином, в середньому точність різних алгоритмів класифікації текстової інформації варіюється від 70% до 90 %. При цьому точність

класифікації залежить не тільки від обраного алгоритму класифікації, але і від початкових даних [10].

Доцільно проаналізувати та створити двохступеневий метод який б раціонально обробляв початкові данні та класифікував їх з меншою долею помилок.

## 1.4 Постановка задачі

Метою роботи є створення концептуальної моделі двохступеневої класифікації тексту на прикладі стандартизованих навчальних та тестових наборів текстових даних.

Для досягнення поставленої мети мають бути вирішені такі задачі:

- огляд існуючих методів класифікації текстових даних;
- аналіз методів попередньої обробки та підготовки вхідних текстових даних;
- розробка концептуальної модель двохступеневої класифікації тексту;
- виконання досліджень впливу скорочення вектору частот появи у тексті ключових слів на точність класифікації тексту. Дослідити вплив на текстових колекціях різного розміру та тематики;
- аналіз отриманих результатів.

# 2 АНАЛІЗ ТЕХНОЛОГІЙ ДЛЯ КОНЦЕПТУАЛЬНОЇ МОДЕЛІ ДВОХСТУПЕНЕВОІ КЛАСИФІКАЦІЇ ТЕКСТУ

## 2.1 Визначення апаратної бази для виконання експериментальної частини проекту

У якості апаратної бази для виконання експериментальної частини проекту використовуеться компьютер на базі процесора Intel з встановленою операційною системою Windows 10.

Відтворення експеримента можливо на будь-якому компьютері з встановленою операційною системою Windows 10. Також можлива адаптація під запуск на інших операційних системах. Можливе виконання на будь-яких процессорах підтримуючих набори команд 64-біт.

## 2.2 Аналіз технологій для вирішення поставленої задачі

### 2.2.1 Мова програмування Python

Python – це потужна мова програмування, створена Гвідо ван Россумом у 1991 році. Python був розроблений як інтерпретована мова програмування для загального використання. Інтерпретована мова означає, що програмний код перетворюється на байт-код, а потім виконується інтерпретатором, який у цьому випадку є віртуальною машиною Python.

З роками популярність і функціональність Python зростали, що призвело до його гнучкості у використанні. Можливість швидкого внесення і тестування змін в програмний код програмного забезпечення робить це простим завданням, з яким при необхідності можна впоратися "на льоту".

Python не тільки простий у використанні, але і легкий в освоєнні. Ці два фактори призвели до того, що ця мова стала основною мовою для

початківців, які вивчають розробку програмного забезпечення. Крім того, його універсальність як мови програмування загального призначення робить його придатним для потреб багатьох галузей промисловості.

Python чудово підходить практично для будь-яких потреб розробки, будь то програмування для серверів, операційних систем, програмного забезпечення, ігор тощо.

Він використовувався для сценаріїв та автоматизації в багатьох галузях промисловості, від програмування машин на великих заводах до використання в розважальних цілях, таких як відеоігри. Способи,якими написання сценаріїв та автоматизації приносять користь різним галузям промисловості, незліченні, як і кількість галузей промисловості.

Python використовується для розробки програмного забезпечення з моменту його першого випуску і продовжує залишатися дуже популярним для цієї мети. Більше того, він використовується для створення програмного забезпечення для декількох різних платформ, оскільки він сумісний з багатьма операційними середовищами, комп'ютерними операційними системами, мобільними пристроями та навіть інструментальними середовищами виконання.

Python добре підходить для аналізу даних, оскільки дозволяє створювати ефективні Візуальні уявлення складних наборів даних. Цей аспект його можливостей робить його основним продуктом у галузі науки про дані.

У галузі наук про дані доступ до широкого спектру методів візуалізації даних, таких як гістограми, лінійні графіки та кругові діаграми, є вигідним, оскільки це допомагає спростити процес. Крім того, можливість використовувати одну мову для управління сортуванням, керуванням та переглядом інформації робить цю мову безцінною.

Багато в чому це схоже на його використання в науках про дані, оскільки машинне навчання є формою науки про дані. Різниця тут полягає в тому, що потрібно менше візуального представлення, хоча воно часто все ще

використовується, натомість розробники зосереджуються на обробці даних значущим чином, щоб програмне забезпечення могло розумно використовувати їх. Машинне навчання і програмуванням на основі штучного інтелекту є простим завданням для Python.

### 2.2.2 Бібліотека Scikit-learn

Scikit-learn – один з найбільш широко використовуваних пакетів Python для Data Science та Machine Learning. Це дозволяє виконувати багато операцій і надає багато алгоритмів. Scikit-learn також пропонує чудову документацію щодо своїх класів, методів та функцій, а також опис використовуваних алгоритмів.

Scikit-Learn підтримує:

- попередню обробку даних;
- зменшення розмірності;
- вибір моделі;
- регресія;
- класифікація;
- кластерний аналіз.

Він також надає кілька наборів даних, які ви можете використовувати для тестування своїх моделей.

Scikit-learn не реалізує все, що стосується машинного навчання. Наприклад, він не має комплексної підтримки для:

- нейронних мереж;
- самоорганізованих карт (мереж Кохонена);
- навчання асоціативним правилам;
- навчання з підкріпленням (reinforcement learning).

Scikit-learn заснований на NumPy і SciPy, тому необхідно зрозуміти хоча б ази цих двох бібліотек, щоб ефективно застосовувати Scikit-learn.

Scikit-learn – це пакет з відкритим кодом. Як і більшість матеріалів з

екосистеми Python, він безкоштовний навіть для комерційного використання. Він ліцензований під ліцензією BSD.

Його можна використовувати для підготовки даних до алгоритмів машинного навчання: стандартизації або нормалізації даних, кодування категоріальних змінних тощо.

Scikit-learn підтримує різні методи класифікації, такі як логістична регресія і *k*-найближчих сусідів, метод опорних векторів, наївний байєсівський класифікатор, дерево прийняття рішень, а також ансамбль методів, такі як random forest, AdaBoost і градієнтний бустинг.

### 2.2.3 NLTK

NLTK – провідна платформа для створення програм на Python для роботи з даними на людській мові. Він забезпечує прості у використанні інтерфейси до понад 50 корпусів та лексичних ресурсів, таких як WordNet, а також набір бібліотек обробки текстів для класифікації, токенізації, виділення елементів, тегування, синтаксичного аналізу та семантичних міркувань.

NLTK підходить як для лінгвістів, інженерів, студентів, викладачів, дослідників, так і для промислових користувачів. Він доступний для Windows, Mac OS X та Linux. Найкраще, що NLTK – це безкоштовний проект з відкритим кодом, керований спільнотою.

### 2.2.4 Matplotlib

Matplotlib – одна з найбільш успішних і часто використовуваних бібліотек, що надає різні інструменти для візуалізації даних на Python.

Це одна з найпотужніших бібліотек побудови графіків у Python, яка надає різні інструменти для створення 2D-графіків із даних у списках або

масивах, для публікації графіків і фігур в різних форматах експорту і різних середовищах (pycharm, Jupyter notebook) на різних платформах.

Бібліотека забезпечує процедурний інтерфейс під назвою Pylab, який використовується для того, щоб він працював подібно до MATLAB, мови програмування, яку використовують вчені, дослідники.

Matplotlib разом з NumPy можна розглядати як еквівалент MATLAB з відкритим кодом.

Це чудовий спосіб створити якісні статичні візуалізації, які можна використовувати для публікацій та професійних презентацій.

Він також забезпечує сумісність з різними іншими сторонніми бібліотеками та пакетами, які розширюють його функціональність. Наприклад, seaborn, ggplot, які надають більше можливостей для побудови графіків, а також базова карта і картографія, які використовуються для побудови геопросторових даних.

Очевидно, що matplotlib з різними сумісними сторонніми бібліотеками надає користувачеві потужні інструменти для візуалізації різних даних.

### 2.2.5 TensorFlow

TensorFlow – це наскрізна платформа з відкритим кодом для побудови машинного навчання. Це набір інструментів символьної математики, який виконує багато завдань, включаючи глибоке навчання нейронної мережі та виведення з використанням потоку даних та диференційованого програмування.

TensorFlow від Google в даний час є найвідомішим пакетом глибокого навчання на планеті. Машинне навчання використовується Google у всіх своїх продуктах для поліпшення пошуку, перекладу, підписів до картинок і рекомендацій.

Попередня обробка даних, побудова моделі та, нарешті, навчання та оцінка моделі – це три елементи структури Tensorflow.

Tensorflow отримав свою назву завдяки тому, що він приймає вхідні дані у вигляді багатовимірного масиву, широко відомого як тензори. Ви можете створити блок-схему операцій, які ви хотіли б виконати на цьому вході. Вхідні дані надходять з одного кінця, проходять через цю систему різних процесів і з'являються як вихідні дані на іншому.

Тензор входить, виконує набір операцій, а потім виходить з іншого боку, саме тому він називається TensorFlow.

Google прагне використовувати машинне навчання, щоб максимально використати свої величезні набори даних, щоб забезпечити своїм споживачам найкращий досвід.

Tensor Flow був розроблений для масштабування, оскільки Google має не лише дані; вони також мають найпотужніший комп'ютер у світі. TensorFlow – це дослідницька бібліотека машинного навчання та глибоких нейронних мереж, створена командою Google Brain.

Він був розроблений для роботи на багатьох процесорах або графічних процесорах, а в деяких випадках і на мобільних операційних системах, і включає обгортки на Python, C++ та Java.

## 2.3 Аналіз методологічного підґрунтя для рішення поставленої задачі

### 2.3.1 Векторне представлення документа

Документ, як послідовність символів, є поганим об'єктом для класифікації. Зазвичай документи перетворюються в векторне представлення в просторі (зване також простором ознак), так як більшість алгоритмів машинного навчання працює саме в ньому.

Відображення документа в простір ознак – завдання складне і неоднозначне, його детальний розгляд виходить за рамки даної роботи. Проте, для кращого розуміння даних, над якими буде проводитися подальша робота, розглянемо класичний підхід до перетворення документів.

Метод заснований на припущенні, що потрапляння документа в той чи інший клас залежить від слів, що входять в даний текст. Перетворення полягає в тому, що кожному слову, яке зустрічається в якомусь документі, відповідає певна координата в просторі ознак. Значення цієї координати конкретного документа – це кількість слів, що зустрічаються в документі. Для слова, яке не зустрічається в документі, значення відповідної координати дорівнює нулю.


Рисунок 2.1 – Базові методи текст-процесінгу

Кількість всіх слів, що зустрічаються в документах величезне, при цьому багато з них не несуть смислового навантаження, інші мають синоніми.

Для зменшення розмірності і збільшення інформативності використовуються різні техніки ще на етапі приведення документа до векторного виду.

По-перше, складається список, так званих «стоп-слів» (прийменники, союзи і т.п.), ці слова не включаються в ознаковий простір. Якщо вирішується завдання рубрикації web-сторінок, то текст також потрібно очистити від елементів управління, оформлення та іншого «сміття» [11].

По-друге, використовується нормалізація слів: Всі слова в документі

наводяться до початкової форми, а іноді всі синоніми замінюються на одне слово. Для англійської мови завдання нормалізації вирішується простіше, ніж для української. Звичайною процедурою є відсікання закінчення слова (stemming). В українській мові виникають проблеми багатозначності слів або неоднозначності початкової форми.

### 2.3.2 Методи класифікації, засновані на правилах

Методи класифікації текстів, засновані на правилах, працюють за допомогою чітко розроблених лінгвістичних правил. Система використовує правила, створені інженером, щоб визначити, до якого класу повинен належати даний фрагмент тексту, шукаючи підказки у вигляді семантично релевантних текстових елементів. Кожне правило має шаблон, якому повинен відповідати текст, щоб бути поміщеним у відповідну категорію.

Перевага систем, заснованих на правилах, полягає в тому, що їх вхідні та вихідні дані передбачувані та інтерпретовані людьми, і їх можна покращити за допомогою ручного втручання інженера. Однак методи класифікації, засновані на правилах, також є дещо крихкими, і їх часто важко узагальнити, оскільки вони можуть дотримуватися лише заздалегідь визначених шаблонів, які були запрограмовані. Наприклад, слово "хмара" може означати вологість у небі або цифрову хмару, в якій зберігаються дані. Системам, заснованим на правилах, важко впоратися з цими нюансами без того, щоб інженери не витрачали неабияку кількість часу, намагаючись вручну передбачити і підлаштовуватися під ці тонкощі.

### 2.3.3 Системи машинного навчання

Як згадувалося вище, системи, засновані на правилах, мають обмеження, оскільки їх функції та правила повинні бути попередньо запрограмовані. Навпаки, системи класифікації, засновані на машинному

навчанні, працюють, застосовуючи алгоритми, які аналізують набори даних на предмет шаблонів, пов'язаних з певним класом.

Алгоритмам машинного навчання передаються попередньо позначені / попередньо класифіковані екземпляри, які аналізуються на предмет відповідних функцій. Ці попередньо позначені екземпляри є навчальними даними.

Класифікатор машинного навчання аналізує навчальні дані та вивчає закономірності, пов'язані з різними класами. Після цього невидимі екземпляри позбавляються своїх міток і передаються алгоритму класифікації, який присвоює екземплярам мітку. Потім призначені мітки порівнюються з оригінальними мітками, щоб побачити, наскільки точним був класифікатор машинного навчання, оцінюючи, наскільки добре модель засвоїла, які моделі передбачають, які класи.

Алгоритми машинного навчання працюють на основі аналізу числових даних. Це означає, що для того, щоб використовувати алгоритм машинного навчання на текстових даних, текст повинен бути перетворений в числовий формат. Існують різні методи кодування текстових даних у вигляді числових даних та створення методів машинного навчання на основі цих даних.

### 2.3.4 Мішок слів

Набір слів є одним із найбільш часто використовуваних підходів для кодування та представлення текстових даних. Термін "мішок слів" походить від того факту, що ви по суті берете всі слова в документах і поміщаєте їх усі в один "пакет", не звертаючи уваги на порядок слів або граматику, звертаючи увагу лише на частоту слів у пакеті. В результаті виходить довгий масив, або вектор, що містить єдине представлення всіх слів у вхідних документах [12].

Після визначення розміру вектора об'єктів кожному документу в списку спільних документів присвоюється власний вектор, заповнений числами, які вказують, скільки разів слово, про яке йде мова, зустрічається в

поточному документі. Це означає, що якщо слово "їжа" зустрічається вісім разів в одному текстовому документі, то відповідний вектор об'єктів / масив об'єктів матиме вісімку у відповідній позиції.

Іншими словами, всі унікальні слова, які з'являються у вхідних документах, складаються в один пакет, а потім кожен документ отримує вектор слів однакового розміру, який потім заповнюється кількістю разів, коли різні слова з'являються в документі.

Текстові набори даних часто містять велику кількість унікальних слів, але більшість з них використовуються не дуже часто. З цієї причини кількість слів, що використовуються для створення вектора слів, зазвичай обмежується.

### 2.3.5 TF-IDF

Інший спосіб подання документа на основі слів, які він містить, називається частота терміна – зворотна частота документа (TF-IDF). Підхід TF-IDF також створює вектор, що представляє документ на основі слів, які він містить, але на відміну від набору слів, ці слова зважуються не лише за їх частотою [13]. TF-IDF розглядає важливість слів у документах, намагаючись кількісно визначити, наскільки це слово стосується теми документа. Іншими словами, TF-IDF аналізує релевантність замість частоти, а кількість слів у векторі ознак замінюється оцінкою TF-IDF, яка обчислюється для всього набору даних.

Підхід TF-IDF заснований на першому обчисленні частоти термінів, тобто кількості разів, коли унікальні терміни з'являються в конкретному документі. Однак TF-IDF також дбає про обмеження впливу надзвичайно поширених слів, таких як "the", "або" і "і", оскільки ці "стоп-слова" дуже поширені, але передають дуже мало інформації про зміст документа. Ці слова потрібно відкинути, на що посилається частина TF-IDF "частота зворотного документа". Це робиться тому, що чим в більшій кількості

документів зустрічається певне слово, тим менш корисно це слово для відмінності його від інших документів в списку всіх документів. Формула, яку TF-IDF використовує для обчислення важливості слова, призначена для збереження слів, які є найчастішими та найбільш семантично багатими.

Вектори ознак, створені за допомогою підходу TF-IDF, містять нормалізовані значення, які в сумі дорівнюють одиниці, присвоюючи кожному слову зважене значення, розраховане за формулою TF-IDF.

### 2.3.6 Вкладення слів

Вкладення слів – це методи представлення тексту, які гарантують, що слова зі схожими значеннями мають схожі числові подання.

Вкладання слів працює шляхом "векторизації" слів, що означає, що воно представляє слова як вектори з реальними значеннями у векторному просторі. Вектори існують у сітці або матриці, і вони мають напрямок і довжину (або величину). При поданні слів у вигляді векторів слова перетворюються в вектори, що складаються з дійсних значень [14]. Кожне слово зіставляється з одним вектором, і слова, схожі за значенням, мають однаковий напрямок і величину. Цей тип кодування дозволяє алгоритму машинного навчання вивчати складні взаємозв'язки між словами.

Вкладення, що представляють різні слова, створюються з урахуванням того, як використовуються розглянуті слова. Оскільки слова, які використовуються подібними способами, матимуть подібні вектори, процес створення вкладень слів автоматично перекладає частину значення, яке мають ці слова. Підхід "мішок слів", навпаки, створює тендітні уявлення, в яких різні слова матимуть несхожі уявлення, навіть якщо вони використовуються в дуже схожих контекстах.

Існують різні алгоритми та підходи, які використовуються для створення вкладень слів. Деякі з найбільш поширених і надійних методів вбудовування слів включають: шари вбудовування, word2vec і GloVe.

### 2.3.7 Вбудовування шарів

Одним із потенційних способів використання вбудовування слів поряд із системою машинного навчання / глибокого навчання є використання шару вбудовування. Шари вбудовування – це шари глибокого навчання, які перетворюють слова у вбудовування, які потім передаються решті системи глибокого навчання. Вкладення слів вивчаються в міру того, як мережа тренується для виконання конкретного текстового завдання [15].

При підході до вбудовування слів схожі слова матимуть схожі уявлення і будуть ближче один до одного, ніж до несхожих слів.

Щоб використовувати шари вбудовування, текст спочатку потрібно попередньо обробити. Текст у документі повинен бути бінарно закодованим, а розмір вектора повинен бути вказаний заздалегідь. Потім закодований текст перетворюється на словесні вектори, а вектори передаються в модель машинного навчання.

### 2.3.8 Word2Vec

Слова-вектори (word vectors) – це чисельні уявлення слів, що зберігають семантичний зв'язок між ними. Наприклад, для вектора cat (кішка) одним із найближчих буде слово dog (собака). Однак векторне уявлення слова pencil (олівець) досить сильно відрізнятиметься від вектора cat. Ця схожість обумовлена частотою народження двох слів (тобто [cat, dog] або [cat, pencil]) в одному контексті.

Word2Vec – це поширений метод вбудовування слів, який використовує статистичні методи для перетворення слів у вкладення та оптимізований для використання з моделями на основі нейронних мереж. Word2Vec був розроблений дослідниками Google, і це один з найбільш часто використовуваних методів вбудовування, оскільки надійно забезпечує корисні, насичені вбудовування.

### 2.3.9 GloVe

GloVE, або глобальний вектор для представлення слів, базується на алгоритмах вбудовування, що використовуються Word2Vec. Методи вбудовування GloVE поєднують аспекти як Word2Vec, так і методів факторизації матриць, таких як прихований семантичний аналіз. Перевага Word2Vec полягає в тому, що він може фіксувати контекст, але як компроміс він погано фіксує глобальну текстову статистику. І навпаки, традиційні векторні подання корисні для визначення глобальної текстової статистики, але вони не корисні для визначення контексту слів і фраз. GloVE використовує найкраще з обох підходів, створюючи контекст слів на основі глобальної текстової статистики.

# 3 ОПИС ЗАПРОПОНОВАНОЇ МОДЕЛІ ДВОХСТУПЕНЕВОІ КЛАСИФІКАЦІЇ ТЕКСТУ

## 3.1 Опис концептуальної моделі

При виборі конкретного алгоритму класифікації тексту слід враховувати особливості кожного з них. Як і раніше, залишається невирішеним питання визначення набору класифікуючих ознак, їх кількості та способів обчислення ваг. У алгоритмах глибокого навчання точність класифікації залежить від наявності навчальної вибірки відповідного розміру. Підготовка такої вибірки – дуже трудомісткий процес. Досі залишається відкритою проблема підбору параметрів деяких алгоритмів на етапі навчання.

На рисунку 3.1 представлено загальну схему процесу класифікації з урахуванням основних етапів та варіантів їх реалізації.

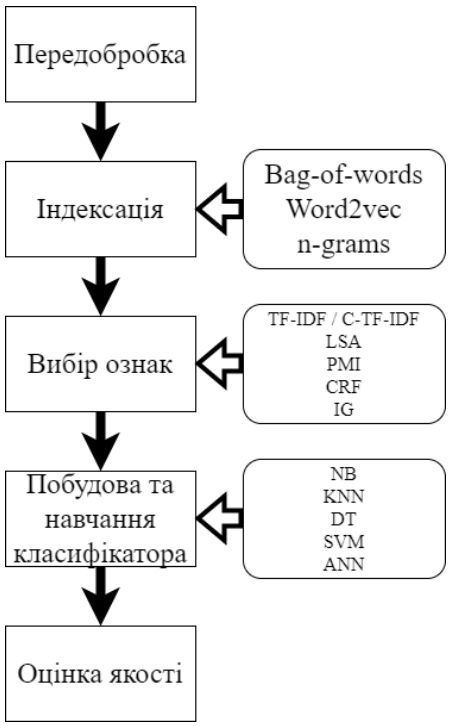


Рисунок 3.1 – Етапи процесу автоматичної класифікації тексту

Після аналізу існуючих способів автоматичної класифікації текстів була розроблена нова модель двоступеневої класифікації. Вона представлена на рисунку 3.2 у вигляді IDEF0 нотації.

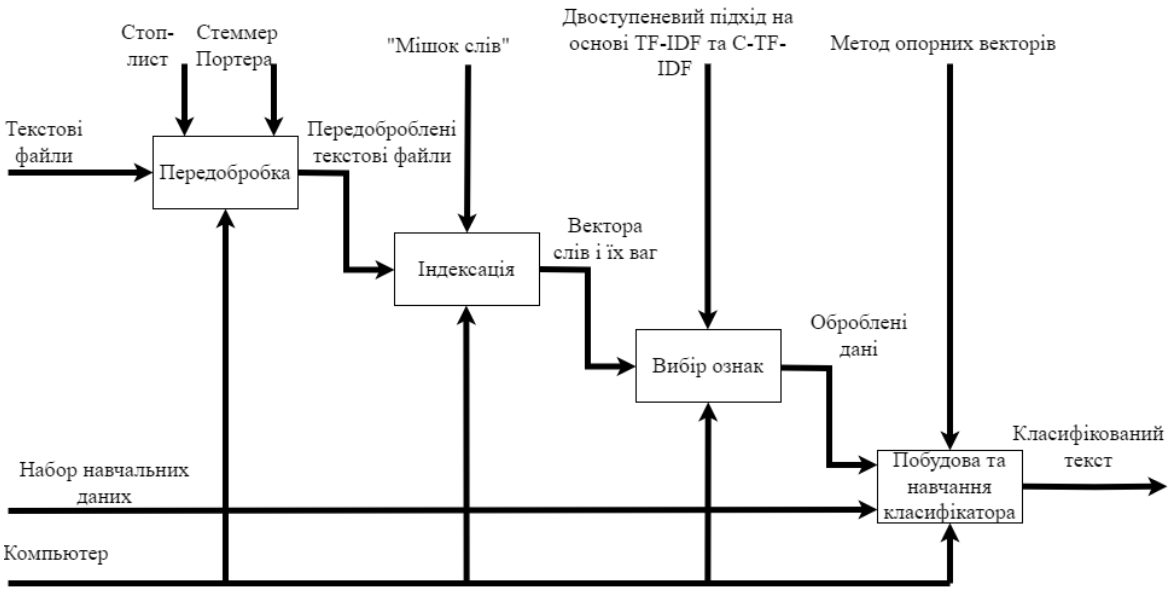


Рисунок 3.2 – Концептуальная модель двоступеневої класифікації тексту у вигляді IDEF0 нотації

У запропонованій моделі вибірка ознак виконується завдяки двоступеневому підходу на основі TF-IDF та C-TF-IDF.

C-TF-IDF – це процедура TF-IDF на основі класів, яка може використовуватися для створення об'єктів із текстових документів на основі класу, в якому вони знаходяться.

Ціль TF-IDF на основі класів – надати всім документам в межах одного класу один і той самий векторний клас. Для цього ми повинні почати розглядати TF-IDF з точки зору класів, а не окремих документів.

C-TF-IDF краще всього можна пояснити як формулу TF-IDF, прийняту для кількох класів, шляхом об'єднання всіх документів для кожного класу. Таким чином, кожен клас преобразуется в один документ, а не в набір документів.

Далі розглянемо кожен з етапів моделі більш докладно.

## 3.2 Попередня обробка, представлення документа та зважування термів

Для попередньої обробки документів ми виконуємо всі стандартні операції попередньої обробки, такі як видалення неалфавітних символів і тегів розмітки, вирівнювання регістру, виключення стоп-слів і виділення ключових слів. Ми використовуємо інтелектуальний системний стоп-лист для видалення стоп-слів і широко використовуваний стеммер Портера для вилучення кореневих слів.

Форма "мішок слів" прийнята як найпростіший і успішний підхід до представлення документа в задачах класифікації тексту. У цьому стандартному підході лише слова в документах розглядаються як ознаки в алгоритмі машинного навчання, що використовується для класифікації. Використання алгоритму машинного навчання з цими основними функціями з навчальними та тестовими даними є прямою, фундаментальною та загальноприйнятою архітектурою для задач класифікації тексту [16].

Як підхід до зважування термінів ми використовуємо метрику TF-IDF, яка є простою мірою, що враховує частоти термів і зменшує важливість термів, спільних для всього набору даних, використовуючи частоти документів [16]. Для оптимізованого обчислення TF-IDF кожен вектор документа нормалізується таким чином, щоб він мав одиничну довжину для обліку документів різної довжини.

## 3.3 Алгоритм машинного навчання

Кілька досліджень порівнювали характеристики різних підходів до машинного навчання, і загалом було показано, що метод опорних векторів з лінійним ядром дає успішні результати [17,18]. Для фундаментальних проблем в області класифікації текстів (висока розмірність, рідкісні екземпляри, розділюваність класів) метод опорних векторів забезпечує ефективні рішення, будучи більш несприйнятливим до проблеми

перенавчання, використовуючи адитивний алгоритм з індуктивним зміщенням, який підходить для задач з щільними поняттями і розрідженими екземплярами, і використовуючи базову модель лінійного поділу, яка відповідає дискримінації більшості класів [19]. Грунтуючись на цих позитивних аспектах і його успіхи в попередніх дослідженнях, ми вирішили використовувати метод опорних векторів з лінійним ядром в якості модуля машинного навчання в цій роботі.

Метод опорних векторів створює гіперплощину або набір гіперплощин у великому або нескінченно вимірному просторі, який можна використовувати для класифікації, регресії чи інших завдань. Інтуїтивно зрозуміло, що хороше розділення досягається гіперплощиною, яка має найбільшу відстань до найближчих точок навчальних даних будь-якого класу (так званий функціональний запас), оскільки, як правило, чим більший запас, тим менша помилка узагальнення класифікатора. На рисунку 3.3 показана функція рішення для задачі з лінійним відокремленням із трьома зразками на границях полів, які називаються «опорними векторами»:

Загалом, коли задача не є лінійно роздільною, опорними векторами є вибірки в межах полів.

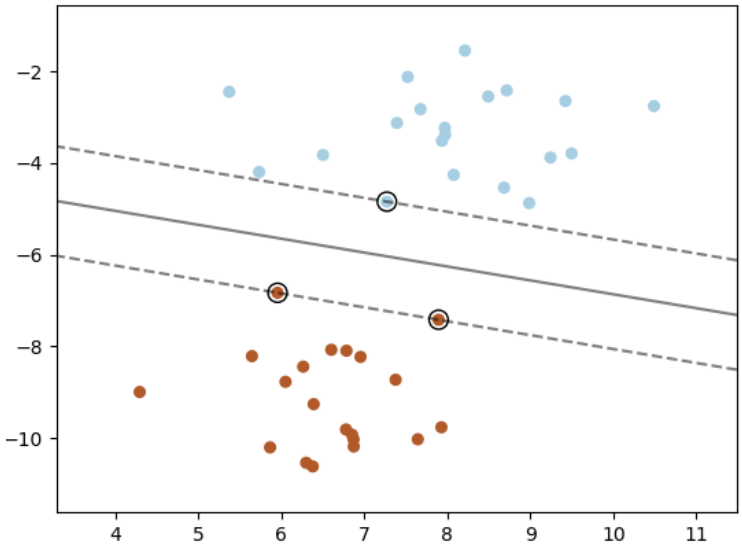


Рисунок 3.3 – Розділяюча гіперплощина в методі опорних векторів

Ціль алгоритму опорних векторів машини – знайти гіперплоскость в $N$-мірному просторі ($N$ – кількість об'єктів), яка чітко класифікує точки даних.

Щоб розділити два класи точок даних, можна вибрати безліч можливих гіперплоскостей. Ціль – знайти площину з максимальним запасом, то є максимальна відстань між точками даних обох класів. Максимізація відстані між полями забезпечує деяке усилення, щоб майбутні точки даних можна було класифікувати з більшою впевненістю.

Гіперплоскості – це межі прийняття рішень, які допомагають класифікувати точки даних. Точки даних, що потрапляють по обидві сторони від гіперплоскості, можуть бути віднесені до різних класів. Крім того, розмір гіперплоскості залежить від кількості об'єктів. Якщо кількість вхідних об'єктів рівна 2, то гіперплоскость – це просто лінія. Якщо кількість вхідних об'єктів дорівнює 3, то гіперплоскость стає двомірною площиною.

Опорні вектори – це точки даних, які знаходяться ближче до гіперплоскості і впливають на положення і орієнтацію гіперплоскості. Використовуючи ці опорні вектори, ми максимізуємо запас класифікатора. Удалення опорних векторів змінило положення гіперплоскості. Це моменти, які допомагають нам створити наш SVM.

Ми використовуемо реалізацію метода опорних векторів у бібліотеці Scikit-learn для програмної мови Python.

## 3.4 Вибір функції

Основною мотивацією в цій роботі є зосередження уваги як на підходах до вибору ознак на основі корпусу, так і на основі класу і їх об'єднання таким чином, щоб підвищити продуктивність класифікатора. З одного боку, для вибору ознак на основі корпусу ми застосовуємо обрізку (фільтрацію низькочастотних термінів) до всього набору даних і виконуємо аналіз для оптимального рівня обрізки, використовуючи сім різних рівнів від

2 до 30. У літературі для цієї мети зазвичай використовується довільно вибране і мале значення (наприклад, 2 або 3). З іншого боку, ми досліджуємо вибір функцій за допомогою TF-IDF на основі класу (який перевершує TF-IDF на основі корпусу, як згадувалося раніше), витягуючи ряд найбільш інформативних ключових слів у кожному класі. Ми експериментуємо з п'ятьма різними ключовими словами в діапазоні від 250 до 4000, щоб визначити оптимальну кількість ключових слів. Надалі будемо називати ці два кроки скороченням (на основі корпусу) та вибором ключових слів (на основі класу) відповідно. Нарешті, аналізуючи результати цих процесів фільтрації та відбору та витягуючи параметри, що відповідають оптимальним характеристикам у цих експериментах, ми отримуємо додатковий етап, який поєднує скорочення на основі корпусу з вибором ключових слів на основі класу.

## 3.5 Методи

У цій роботі ми в основному реалізуємо чотири основні підходи: усі слова (AW), усі слова зі скороченням на основі корпусу (AWP), усі слова з вибором ключових слів на основі класу (AWK) та двоступеневий вибір ознак як зі скороченням, так і з вибором ключових слів (AWPK).

Метод AW – це основний метод, який використовує стандартний підхід bow з усіма словами у векторі ознак.

AWP враховує всі слова в колекції документів, але фільтрує їх за допомогою процесу обрізання. У цьому методі фільтруються терміни, які зустрічаються менше певного порогового значення у всьому навчальному наборі. Ми називаємо це порогове значення рівнем обрізки (PL). PL = n ($n \geq 1$) вказує на те, що терміни, що зустрічаються принаймні n разів у навчальному наборі, використовуються у векторі рішення, тоді як інші ігноруються. Зверніть увагу, що PL=1 відповідає методу AW (тобто без обрізання). Ми виконуємо налаштування параметрів, аналізуючи різні значення для кожного

набору даних, щоб досягти оптимальних значень PL для методу AWP. Ми проводимо експерименти з різними рівнями обрізки від 2 до 30: 2, 3, 5, 8, 13, 20, і 30.

У методі AWK для кожного класу вибираються окремі ключові слова. Такий підхід надає рівну вагу кожному класу на етапі вибору ключового слова. Ми експериментуємо з п’ятьма різними кількостями ключових слів (250, 500, 1000, 2000 та 4000) і порівнюємо результати з AW, який включає всі слова як об'єкти у вектор рішення.

Метод AWPK розроблений як оптимальне поєднання AWP і AWK шляхом зміни рівня обрізки і кількості параметрів ключових слів. Значення параметрів, які дають найкращі результати в базових методах, використовуються для експериментів AWPK.

# 4 АНАЛІЗ ОТРИМАНИХ РЕЗУЛЬТАТІВ

## 4.1 Опис методології проведення дослідження

На основі підходів, розглянутих у попередньому розділі, у цьому розділі ми визначаємо оптимальні значення параметрів (рівень обрізання та кількість ключових слів) для методів у всіх наборах даних. Експерименти оцінювали, а методи порівнювали щодо мікро-усередненої F-міри (MicroF), яка є середнім показником успішності документів, та макросередньої F-міри (MacroF), яка є середнім показником успішності категорій [16].

Для трьох наборів даних аналізувалися залежності між:

- вектором частот ключових слів та точністю класифікації;
- розміром текстових колекцій та якістю класифікації;
- методами класифікації та наборами текстових даних.

В результаті експериментів має бути отримана оцінка впливу скорочення вектора частот появи у тексті ключових слів на точність класифікації тексту, а також проаналізовано вплив вибору діапазону ключових слів за метрикою TF-IDF на якість класифікації.

Дослідження впливу вибору рангу ключових слів за метрикою TF-IDF на якість класифікації є другим екпериментом, що також має проведено на трьох текстових колекціях.

## 4.2 Набори даних

У цій роботі ми використовуємо три добре відомі набори даних із сховища машинного навчання UCI: Reuters-21578 (Reuters), реферати премії Національного наукового фонду за дослідження (NSF) та групи новин Mini 20 (MiniNg20). [20]. Ці набори даних мають різні характеристики, які можуть мати вирішальне значення для ефективності класифікації. Асиметрія – одна з

ключових властивостей набору даних, яка визначається як розподіл кількості документів за класами. Набір даних, що має низький коефіцієнт асиметрії, вказує на те, що це збалансований набір даних з приблизно однаковою кількістю зразків документів для кожного класу. Допустимість декількох класів для документів (що вказують на те, що документ може належати до більш ніж однієї теми), довжина документа (наприклад, короткі тези або довгі новинні статті), пропорції поділу (навчальні та тестові набори), рівень формальності (наприклад, офіційні документи журналу або неофіційні повідомлення інтернет-форуму) є іншими властивостями наборів даних.

В експериментах ми використовуємо стандартні розбиття наборів даних Reuters і MiniNg20. Для NSF дані, пов'язані з 2001 роком, були обрані випадковим чином, і з цього року було вибрано п'ять розділів (чотири розділи для навчання та один розділ для тестування). Формуємо п'ять різних розбивок, повторюємо всі тести з ними та беремо середнє як кінцевий результат.

### 4.2.1 Набір даних Reuters-21578

Reuters-21578 є, мабуть, найбільш часто використовуваною колекцією для класифікації текстів за останні два десятиліття, і вона використовується в найвпливовіших статтях у цій галузі. Цей набір даних містить структуровану інформацію про статті стрічки новин, які можуть бути віднесені до декількох класів, що створює проблему з кількома мітками. У ньому сильно спотворено розподіл документів за категоріями, де велика частина документів відноситься до декількох тем.

Колекція складається з 21 578 документів, включаючи документи без теми та друкарських помилок. З цієї причини традиційно використовується підмножина колекції під назвою "ModApte". Цей набір даних включає 9 603 документів для навчання та 3299 для тестування. Цей поділ присвоює документи від 7 квітня 1987 року і раніше навчальному набору, а документи від 8 квітня 1987 року і після – тестовому набору. Крім того, додатковим

кроком є зосередження уваги лише на категоріях, які мають принаймні один документ у навчальному наборі та наборі тестів. Після цього набір даних містить 90 категорій із навчального набору обсягом 7769 документів та тестовим набором із 3019 документів.

Основна причина, по якій ми зосередимося на цій колекції, полягає в тому, що це одна з найбільш класичних колекцій текстової класифікації, і вона дозволить нам порівняти наші результати з великим набором раніше опублікованих результатів для декількох алгоритмів, маючи можливість запустити її на наших компьютерах.

### 4.2.2 Набір даних NSF

Набір даних NSF складається з 129 000 рефератів, що описують нагороди NSF за фундаментальні дослідження в період з 1990 по 2003. Рівень формальності набору даних високий. Це дозволяє використовувати кілька тем. NSF не є ідеально збалансованим набором даних, але його коефіцієнт асиметрії також не такий високий, як у Reuters. Довжина документа коротка через його абстрактний зміст.

Тези по одному на файл були надані Національним науковим фондом (NSF). Пакет даних було отримано шляхом автоматичної обробки тез текстовим аналізатором NSFAbst, створеним з використанням VisualText. Хоча більшість полів вихідних даних дуже точні, автори не витягли дані з поля Investigator зі 100% точністю через велику мінливість у цьому полі. Список слів отримано в результаті окремого процесу і може не включати всі цікаві слова в анотаціях.

### 4.2.3 Набір даних MiniNg20

Набір даних MiniNg20 складається з 2000 повідомлень (розділені на 1600 для навчання та 400 для тестування), що є набором із 100 повідомлень

для кожної з 20 різних груп новин Usenet. На відміну від двох інших наборів даних, MiniNg20 є неформальним, із багатьма граматичними помилками, дозволяє лише одну тему на текст і є збалансованим набором даних, що містить однакову кількість повідомлень для кожної теми.

## 4.3 Аналіз рівня обрізки – AWP

У цьому експерименті метод AWP було реалізовано з кількома значеннями PL (PL=1 відповідає AW) для трьох наборів даних. У таблиці 4.1 наведено номер функції та показники успіху «мікро» та «макро» для кожного рівня скорочення. У першому стовпці таблиці вказується метод і значення параметра PL, розділені комою. Як можна побачити, процес скорочення покращує рівень успіху класифікатора, і найкращі результати (висока точність із низькими номерами ознак) отримують приблизно PL=13 послідовно у всіх трьох наборах даних із двома різними параметрами продуктивності.

Таблиця 4.1 – Показники успішності AWP (оптимальні результати жирним шрифтом)

| Метод, Параметр | Reuters | | | NSF | | | MiniNg20 | | |
|---|---|---|---|---|---|---|---|---|---|
| | Featu-re# | MicroF | MacroF | Featu-re# | MicroF | MacroF | Featu-re# | MicroF | MacroF |
| AW | 20292 | 85.58 | 43.83 | 13424 | 64.46 | 46.11 | 30970 | 46.42 | 43.44 |
| AWP,2 | 12959 | 85.55 | 43.84 | 8492 | 64.41 | 46.21 | 13102 | 49.73 | 47.13 |
| AWP,3 | 9971 | 85.52 | 43.93 | 6328 | 64.62 | 46.42 | 9092 | 49.64 | 47.19 |
| AWP,5 | 7168 | 85.51 | 44.56 | 4528 | 64.86 | 46.49 | 6000 | 51.26 | 48.52 |
| AWP,8 | 5268 | 85.73 | 44.91 | 3376 | 64.66 | 46.38 | 4169 | 52.48 | 49.90 |
| **AWP,13** | **3976** | **85.84** | **44.85** | **2478** | **64.58** | **46.49** | **2863** | **53.62** | **51.02** |
| AWP,20 | 3046 | 86.02 | 44.55 | 1875 | 64.23 | 46.67 | 2025 | 53.78 | 51.02 |
| AWP,30 | 2237 | 81.29 | 43.59 | 1419 | 63.84 | 46.21 | 1384 | 52.89 | 50.46 |

Дотримуючись узагальнення, що слова, які зустрічаються менше 10 – 15 разів у наборі даних, швидше за все, не є хорошим показником для класифікації текстів, ми встановили PL=13 в експериментах на основі скорочення. Цей результат вказує на те, що поширена в літературі думка про те, що для усунення неінформативних термів досить рівня скорочення в 2 – 3, не відповідає дійсності.

## 4.4 Аналіз вибору ключових слів на основі класу – AWK

У цьому експерименті продуктивність методу AWK аналізувалася за допомогою різних параметрів номера ключового слова (функції). Результати наведено в таблиці 4.2. Показники успіху для AW також включені в таблицю для порівняння.

Таблиця 4.2 – Показники успішності AWK (оптимальні результати виділено жирним шрифтом)

| Метод, Параметр | Reuters | | NSF | | MiniNg20 | |
|---|---|---|---|---|---|---|
| | MicroF | MacroF | MicroF | MacroF | MicroF | MacroF |
| AWK,250 | 83.69 | 51.15 | 62.04 | 49.51 | 56.65 | 55.72 |
| AWK,500 | 84.71 | 50.92 | 62.92 | 49.31 | 56.16 | 55.01 |
| AWK,1000 | 85.16 | 51.72 | 64.69 | 49.33 | 53.68 | 52.17 |
| **AWK,2000** | **85.58** | **52.03** | **65.19** | **49.31** | **54.04** | **52.10** |
| **AWK,4000** | **85.84** | **52.10** | **65.71** | **49.35** | **55.25** | **53.73** |
| AW | 85.58 | 43.83 | 64.46 | 46.11 | 46.42 | 43.44 |

Загалом, метод AWK із кількістю ключових слів від 2000 до 4000 підвищує показники успіху в усіх наборах даних порівняно з методом AW. Тому можна зробити висновок, що використання певного набору ключових слів для кожного класу дає більш успішні результати, ніж використання всіх слів у векторі ознак.

Коли ми аналізуємо результати AWP і AWK разом, ми бачимо, що покращення AWP порівняно з AW є явним у збалансованому наборі даних (MiniNg20), тоді як покращення у спотворених наборах даних (Reuters і NSF) є меншим. З іншого боку, покращення AWK порівняно з AW є більш значним, ніж покращення AWP у всіх наборах даних. Цей приріст продуктивності більш явний у показнику MacroF. У підходах, що базуються на корпусі, документи рідкісних класів мають тенденцію бути неправильно класифікованими, оскільки слова переважаючих класів домінують у векторі ознак.

На рисунку 4.1 і 4.2 показано результати microF і macroF відповідно для підходів на основі класів і корпусів із tf-idf представленням документа за допомогою усіх слів і ключових слів у діапазоні від 10 до 2000.

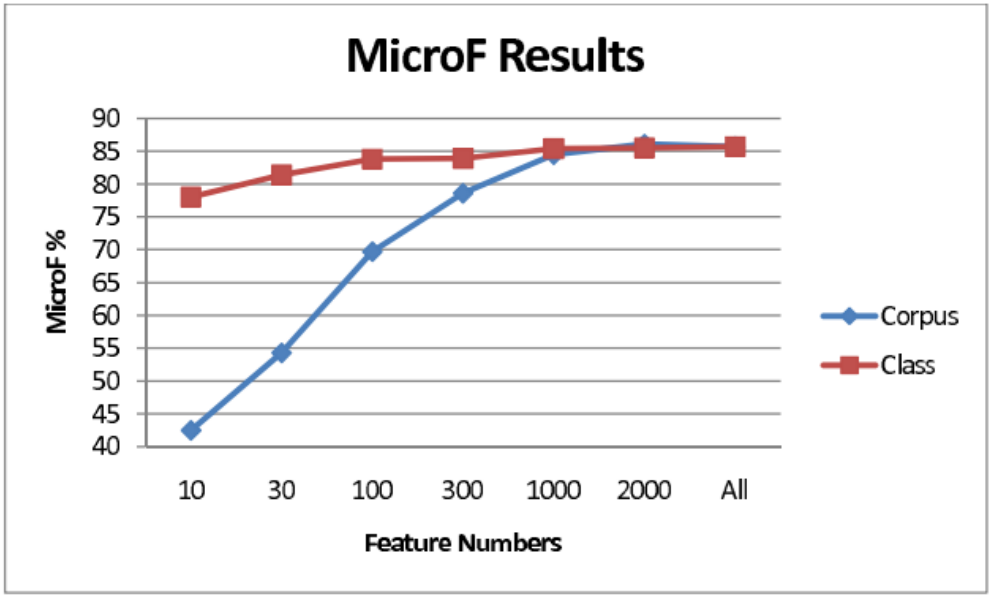


Рисунок 4.1 – MicroF для корпусних і класових підходів

Що стосується результатів microF (рисунок 4.1), ми можемо зробити висновок, що вибір ознак на основі класу досягає вищого рівня microF, ніж підхід на основі корпусу для невеликої кількості ключових слів. У класифікації тексту більша частина навчання відбувається з невеликою, але важливою частиною ключових слів для класу. Вибір ознак на основі класу, за

визначенням, зосереджується на цій невеликій частині; з іншого боку, підхід на основі корпусу знаходить загальні ключові слова, що стосуються всіх класів. Отже, маючи невелику кількість ключових слів, підхід на основі класу досягає набагато більшого успіху, знаходячи більш важливі ключові слова класу. Підхід на основі корпусу не є успішним із такою невеликою частиною, але має крутішу криву навчання, яка досягає пікового значення експериментів (86%) із 2000 ключовими словами на основі корпусу.

Для результатів macroF (рисунок 4.2) ми проаналізували, що вибір ознак на основі класу забезпечує незмінно вищу продуктивність macroF, ніж підхід на основі корпусу. Висока асиметрія при розподілі класів у наборі даних негативно впливає на значення macroF, оскільки macroF надає рівну вагу кожному класу, а не кожному документу, а документи рідкісних класів частіше класифікуються неправильно. Відповідно, середнє значення правильних класифікацій класів різко падає для наборів даних, що мають багато рідкісних класів. Вибір ознак на основі класу є дуже корисним для цієї асиметрії. Як зазначено вище, навіть із невеликою частиною слів (наприклад, 100) метод tf-idf на основі класу досягає 50% успіху, що набагато краще, ніж 43,9% успіху tf-idf з усіма словами.

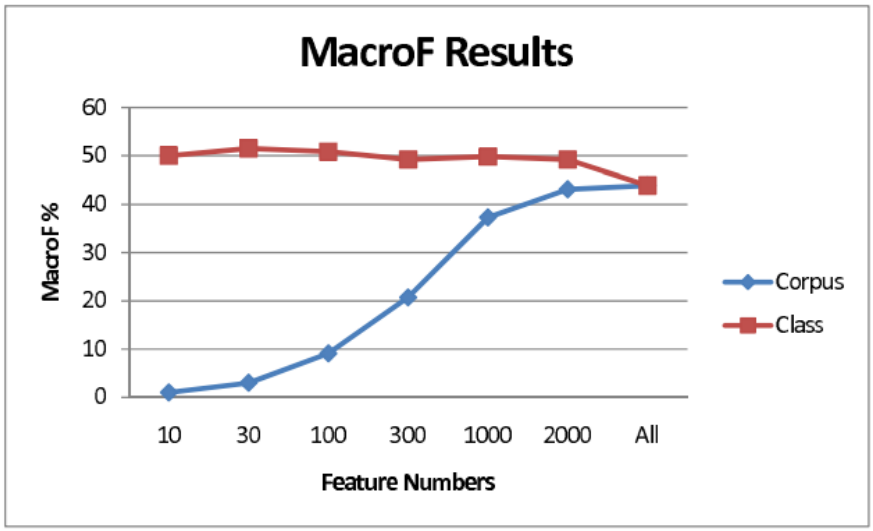


Рисунок 4.2 – MacroF для корпусних і класових підходів

Рідкісні класи успішно характеризуються вибором ознак на основі класу, оскільки кожен клас має власні ключові слова для проблеми категоризації. Корпусний підхід показує гірші результати, оскільки більшість ключових слів вибрано з переважаючих класів, що не дозволяє рідкісним класам бути справедливо представленими своїми ключовими словами.

Показник MacroF надає рівну вагу кожному класу при визначенні успішності класифікатора. Таким чином, особливо для сильно спотворених наборів даних, коли рідкісні класи погано представлені вибраними ознаками, середнє значення правильних класифікацій для рідкісних класів різко падає. Це стосується як AW, так і AWP у викривлених наборах даних, які використовують загальний набір функцій для всіх класів. Однак із вибором ключових слів на основі класу, оскільки кожен клас має власні ключові слова під час класифікації, рідкісні класи характеризуються більш успішним чином. Таким чином, ми спостерігаємо значне збільшення показника успіху (MacroF) за допомогою методу AWK у викривлених наборах даних.

## 4.5 Аналіз двоступеневого вибору ознак (AWPK)

Метод AWPK поєднує в собі оптимальні шаблони використання підходів AWP і AWK. Тому параметрами методу є рівень скорочення та кількість ключових слів. У цьому експерименті ми використовуємо оптимальні значення цих параметрів, визначені під час попередніх аналізів для кожного набору даних: рівень скорочення 13 і кількість ключових слів 2000 і 4000. Результати наведено в таблиці 4.3. У таблиці також показано найкращі показники AW, AWP і AWK для порівняння.

Як видно з таблиці, двоступеневий підхід до вибору ознак перевершує попередні підходи. Вибір найкращих 2000–4000 ключових слів для кожного класу з початковим кроком скорочення значно покращує найкращі показники AWP (з PL=13) і AWK (з 2000–4000 ключовими словами) у всіх трьох наборах даних.

Таблиця 4.3 – Коефіцієнт успішності AWPK (оптимальні результати виділено жирним шрифтом)

| Метод, Параметр | Reuters | | NSF | | MiniNg20 | |
|---|---|---|---|---|---|---|
| | MicroF | MacroF | MicroF | MacroF | MicroF | MacroF |
| **AWPK,13,2000** | **86.40** | **53.95** | **66.06** | **50.11** | **57.43** | **55.66** |
| **AWPK,13,4000** | **86.70** | **53.98** | **66.10** | **50.12** | **57.43** | **55.66** |
| AW | 85.58 | 43.83 | 64.46 | 46.11 | 46.42 | 43.44 |
| AWP,13 | 85.84 | 44.85 | 64.58 | 46.49 | 53.62 | 51.02 |
| AWK,2000 | 85.58 | 52.03 | 65.19 | 49.31 | 54.04 | 52.10 |
| AWK,4000 | 85.84 | 52.10 | 65.71 | 49.35 | 55.25 | 53.73 |

На діаграмі (рисунок 4.3) можна побачити порівняння трьох методів по мікро-усередненій F-мірі, метод AWPK показує себе краще за інших, особливо в наборі Reuters.

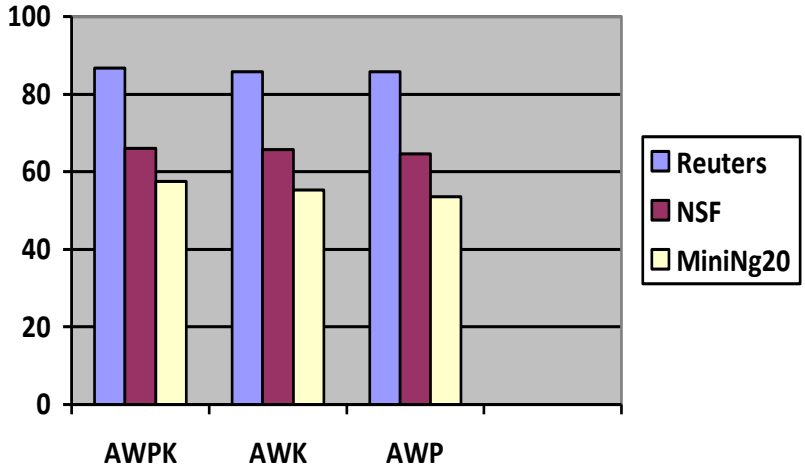


Рисунок 4.3 – Діаграмма порівняння microF трьох методів

При порівнянні трьох методів за макросередньою F-мірою на рисунку 4.4, видно що метод AWPK кращий за інші, але найкращі показники вже з набором даних MiniNg20. Також через неоднорідність даних можна побачити різні показники мікро-усередненої та макросередньої F-міри для різних наборів даних.

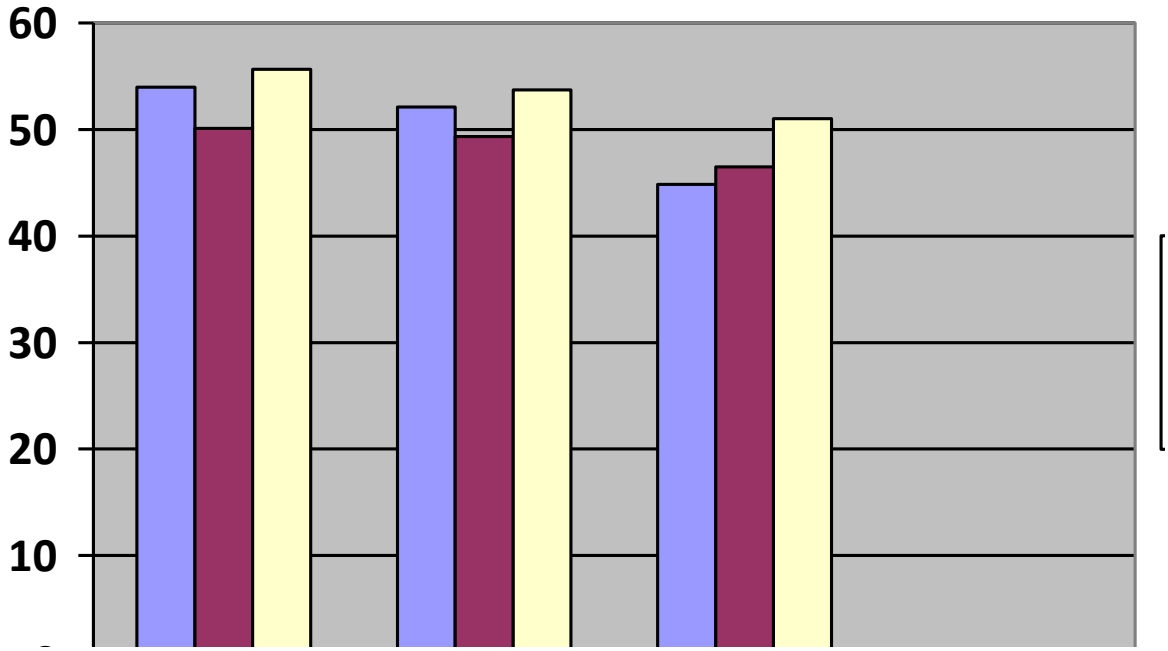


Рисунок 4.4 – Діаграмма порівняння macroF трьох методів

Отже, ми можемо зробити висновок, що додатковий ефект скорочення на основі корпусу продовжується, коли воно поєднується з метрикою вибору ключового слова TF-IDF на основі класу. Як наслідок, метод, запропонований у цій роботі, AWPK, дає найкращу продуктивність. Значущість результатів для трьох методів була виміряна за допомогою тесту статистичних ознак. Ми помітили, що загалом кожен метод значно перевершує метод-попередник. У цьому сенсі AWP і AWK значно кращі, ніж стандартний метод AW, а AWPK значно кращий, ніж AWP і AWK. Таким чином, найдосконалішим методом у цьому дослідженні (AWPK) є оптимальний метод із двоступеневим аналізом вибору ознак.

## 4.6 Підсумок експериментів

У цих експериментах ми зосередилися на політиках охоплення об'єктів (вибір об'єктів на основі корпусу або класу), що використовуються в області класифікації тексту. Спочатку ми проаналізували ефективність обрізання на

основі корпусу (AWP) та вибору ключових слів на основі класу за допомогою TF-IDF (AWK) окремо. Потім, визначивши оптимальні значення параметрів для кожного методу, ми сформували метод AWPK, який є комбінацією цих двох підходів. Наскільки нам відомо, це перша робота, яка поєднує в собі вибір об'єктів на основі класів та корпусів у галузі класифікації текстів.

Можлива майбутня робота полягає у застосуванні двоетапного підходу до відбору ознак до більш семантично орієнтованих методів класифікації текстів, таких як методи, що використовують мовні моделі, лінгвістичні особливості або лексичні залежності. Інтеграція концепцій скорочення та вибору ключових слів у ці методи як двох послідовних кроків може призвести до вищої продуктивності класифікації.

# ВИСНОВКИ

У будь-якій екосистемі накопичується безліч текстових даних, неструктурованих і в різних форматах. Щоб витягти з цього тенденції або значущу інформацію, нам потрібно відсортувати дані за різними категоріями. Класифікація тексту – це простий, потужний метод аналізу для сортування текстового сховища за різними тегами, кожен з яких представляє певне значення. Типові приклади класифікації включають віднесення відгуків клієнтів до категорії позитивних чи негативних, а новин – до категорії спортивних чи політичних.

Машинне навчання використовується для вилучення ключових слів із тексту та класифікації їх за категоріями. Класифікація тексту може бути реалізована за допомогою контрольованих алгоритмів, найпоширенішими варіантами яких є наївний Байєс, SVM та глибоке навчання.

Класифікація текстів знаходить широке застосування в NLP для виявлення спаму, аналізу настроїв, позначення теми або аналізу намірів. Автоматизація рутинних завдань робить пошук, аналіз і прийняття рішень швидше і простіше. Класифікація тексту дуже ефективна при роботі з історичними даними. Вона також може бути використана для аналізу текстового введення в режимі реального часу.

В останні роки спостерігається експоненціальне зростання числа складних документів і текстів, які вимагають більш глибокого розуміння методів машинного навчання, щоб мати можливість точно класифікувати тексти в багатьох випадках. Багато підходів до машинного навчання досягли чудових результатів в обробці природної мови. Успіх цих алгоритмів навчання залежить від їх здатності зрозуміти складні моделі та нелінійні взаємозв'язки в даних. Однак пошук відповідних структур, архітектур і методів для класифікації текстів є складним завданням для дослідників. У цій роботі обговорюється короткий огляд алгоритмів класифікації тексту. У

цьому огляді розглядаються різні методи вилучення текстових об'єктів, методи зменшення розмірності, існуючі алгоритми і методики, а також методи оцінки. Нарешті, обговорюються обмеження кожного методу та їх застосування в реальних завдання.

У цій кваліфікаційній роботі були досліджені методи класифікації текстів, а також можливі модифікації стандартних алгоритмів, проаналізовано зміни якості роботи при зміні конфігурацій алгоритмів, описано концептуальну модель двоступеневої класифікації тексту та виконано програмну реалізацію даної моделі.

# ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Барковська О. Ю., Хомич В. М., Настенко О. С. Дослідження методів обробки та аналізу тексту при організації електронних сховищ інформаційних об’єктів. Сучасний стан наукових досліджень та технологій в промисловості. 2022. № 1 (19). С. 5–12. DOI: https://doi.org/10.30837/ITSSI.2022.19.005

2. Настенко О. С., Барковська О. Ю. Обґрунтування вибору ознак для класифікації тексту. Проблеми інформатизації. 2022. № 2. С. 68. DOI: https://doi.org/10.32620/PI.22.t2

3. Батура Т.В. Методи автоматичної класифікації текстів. Програмні продукти та системи. 2017, № 1. С. 85–99. DOI: https://doi.org/10.15827/0236-235X.117.085-099

4. Jiang M., Liang Y., Feng X., Fan X., Pei Z., Xue Y., Guan R.. Text classification based on deep belief network and softmax regression. Neural Comput. 2018, С. 61–70.

5. Chen W., Xie X., Wang J., Pradhan B., Hong H., Bui D.T., Duan Z., Ma J. A comparative study of logistic model tree, random forest, and classification and regression tree models for spatial prediction of landslide susceptibility. 2017, С. 147–160.

6. Manevitz L.M., Yousef M. One-class SVMs for document classification. 2001, С. 139–154.

7. Marchionini Gary. Exploratory search: from finding to understanding. Communications of the ACM. 2006. Т. 49, № 4, С. 41–46. DOI: https://doi.org/10.1145/1121949.1121979

8. Choudhary Bhoopesh, Bhattacharyya Pushpak. Text clustering using semantics. Proceedings of the 11th International World Wide Web Conference. 2002, С. 1–4.

9. 9. Albitar S., Espinasse B., Fournier S. Towards a Supervised Rocchio-

based Semantic Classification of Web Pages. In Proceedings of the KES, 2012; C. 460–469

10. LeCun Y., Bengio Y., Hinton G. Deep learning. 2015, C. 436–444.

11. Zhang C. Automatic keyword extraction from documents using conditional random fields. J. Comput. Inf. Syst. 2008, C. 1169–1180.

12. Caropreso M.F., Matwin S. Beyond the bag of words: A text representation for sentence selection. In Conference of the Canadian Society for Computational Studies of Intelligence. Berlin, 2006, C. 324–335.

13. Qu Z., Song X., Zheng S., Wang X., Song X., Li Z. Improved Bayes Method Based on TF-IDF Feature and Grade Factor Feature for Chinese Information Classification. In Proceedings of the 2018 IEEE International Conference on Big Data and Smart Computing (BigComp), Shanghai, 2018; C. 677–680.

14. Rezaeinia S.M., Ghodsi A., Rahmani R. Improving the Accuracy of Pre-trained Word Embeddings for Sentiment Analysis., 2017, C. 11–15.

15. Sebastiani F., Machine learning in automated text categorization. ACM Computing Surveys, T. 34, 2002, C. 1–47.

16. Manning C., P. Raghavan, H. Schütze. Introduction to information retrieval. Cambridge University Press, 2008, C. 12–15.

17. Forman G., An extensive empirical study of feature selection metrics for text classification. Journal of Machine Learning Research, 2003, C. 34.

18. Gao Y., S. Sun., An empirical evaluation of linear and nonlinear kernels for text classification using support vector machines. In Proceedings of the 7th international conference on fuzzy systems and knowledge discovery (FSKD), 2010, C. 1502–1505.

19. Joachims T., Advances in kernel methods: support vector learning. MIT Press, 1999, C. 53–55.

20. Frank A., A. Asuncion, UCI machine learning repository, University of California, 2010, C. 78.