Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

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

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

Методи обробки природної мови доповнені знаннями
(тема)

Виконала:
студентка 2 курсу, групи СШІздм-22-1
Десятніченко О. П.
(прізвище, ініціали)

Спеціальність 122 Комп'ютерні науки
(код і повна назва спеціальності)
Тип програми освітньо-наукова
(освітньо-професійна або освітньо-наукова)
Освітня програма Системи штучного інтелекту
(повна назва спеціалізації)
Керівник доц. Головянко М.В.
(посада, прізвище, ініціали)

Допускається до захисту

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

2024 р.

Харківський національний університет радіоелектроніки

Центр Післядипломної освіти
(повна назва)

Кафедра Штучного інтелекту
(повна назва)

Рівень вищої освіти другий (магістерський)

Спеціальність 122 Комп'ютерні науки
(код і повна назва)

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

Освітня програма Системи штучного інтелекту
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«_____» _____ 20 ___ р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Десятніченко Олександрі Павлівні
(прізвище, ім'я, по батькові)

1. Тема роботи Методи обробки природної мови доповнені знаннями

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

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

3. Вихідні дані до роботи Науково-технічні публікації, дані Інтернет-джерел та відомих наукових проєктів, документація до бібліотек TensorFlow, PyTorch, Natural Language Toolkit, SpaCy

4. Перелік питань, що потрібно опрацювати в роботі
1) Аналіз предметної галузі
2) Опис проведених теоретичних досліджень
3) Опис системи, що пропонується
4) Опис проведених експериментальних досліджень

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Отримання завдання на кваліфікаційну роботу	22.04.2024	виконано
2	Аналіз предметної галузі	01.05.2024	виконано
3	Вибір методів обробки тексту	07.05.2024	виконано
4	Навчання та тестування моделей	14.05.2024	виконано
5	Аналіз результатів	21.05.2024	виконано
6	Написання пояснювальної записки	10.06.2024	виконано
7	Перевірка на академічний плагіат	11.06.2024	виконано
8	Нормоконтроль	11.06.2024	виконано
9	Підготовка презентації та доповіді	11.06.2024	виконано
10	Попередній захист	11.06.2024	виконано
11	Рецензування	12.06.2024	виконано
12	Захист перед ЕК	13.06.2024	

Дата видачі завдання 22 квітня 2024 р.

Студент _____
(підпис)

Керівник роботи _____ доц. Головянко М.В.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка: 78 с., 2 рис., 4 дод., 61 джерело.

АВТОМАТИЧНЕ АНОТУВАННЯ, ВІДПОВІДІ НА ЗАПИТАННЯ, ГЛИБОКЕ НАВЧАННЯ, ЗНАННЄВІ БАЗИ, ЗНАННЄВІ ГРАФИ, ІНТЕГРАЦІЯ ЗНАНЬ, КОНТЕКСТУАЛЬНИЙ АНАЛІЗ, МАШИННЕ НАВЧАННЯ, МАШИННИЙ ПЕРЕКЛАД, ОБРОБКА ПРИРОДНОЇ МОВИ, ОНТОЛОГІЇ, РОЗШИРЕННЯ МОВНИХ МОДЕЛЕЙ, СЕМАНТИЧНЕ РОЗУМІННЯ, ТРАНСФОРМЕРИ, ШТУЧНИЙ ІНТЕЛЕКТ.

Об’єкт дослідження – методи обробки природної мови (NLP), які доповнені знаннями, включаючи класичні методи машинного навчання, глибокі нейронні мережі та сучасні трансформери.

Предмет дослідження – вплив інтеграції знань на ефективність методів обробки природної мови, зокрема на точність, надійність і здатність до логічних висновків у різних задачах NLP..

Мета роботи – дослідження і розробка методів обробки природної мови, доповнених знаннями, для підвищення їх ефективності та точності.

Методи дослідження – аналіз літератури, експериментальні дослідження, моделювання та симуляція, аналіз результатів.

ABSTRACT

Master’s thesis contains: 78 pp., 2 fig., 4 ann., 61 references.

ARTIFICIAL INTELLIGENCE, AUTOMATIC ANNOTATION, CONTEXTUAL ANALYSIS, DEEP LEARNING, KNOWLEDGE BASES, KNOWLEDGE GRAPHS, KNOWLEDGE INTEGRATION, LANGUAGE MODEL ENHANCEMENT, MACHINE LEARNING, MACHINE TRANSLATION, NATURAL LANGUAGE PROCESSING, ONTOLOGIES, QUESTION ANSWERING, SEMANTIC UNDERSTANDING, TRANSFORMERS.

Object of the study: natural language processing (NLP) methods that are augmented with knowledge, including classical machine learning methods, deep neural networks, and modern transformers.

Subject of the study: the impact of knowledge integration on the effectiveness of natural language processing methods, particularly on accuracy, reliability, and logical inference capabilities in various NLP tasks.

Purpose of the work: investigation and developing knowledge-augmented natural language processing methods to improve their efficiency and accuracy.

Research methods: literature analysis, experimental research, modeling and simulation, results analysis.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

ШІ – штучний інтелект;

BERT – Bidirectional Encoder Representations from Transformers – двонаправлені представлення енкодера з трансформерів;

CSV – Comma-Separated Values – формат розділених комами значень;

Georgetown-IBM experiment – експеримент Джорджтаун-IBM;

GloVe – Global Vectors for Word Representation – вектори глобального контексту слів;

GPT – Generative Pre-trained Transformer – генеративний попередньо натренований трансформер;

HMM – Hidden Markov Models – приховані марковські моделі;

LLM – Large Language Models – великі мовні моделі;

LSTM – Long Short-Term Memory – довга короткочасна пам'ять;

Naive Bayes – наївний байєсів класифікатор;

NLP – Natural Language Processing – обробка природної мови;

NLTK – Natural Language Toolkit – набір бібліотек і програм для символьної та статистичної обробки природної мови для англійської мови, написаних мовою програмування Python;

RNN – Recurrent Neural Networks – рекурентні нейронні мережі;

SVM – Support Vector Machines – метод опорних векторів;

SpaCy – сучасна бібліотека для обробки природної мови на Python;

WordNet – лексична база даних.

ВСТУП

В сучасному світі штучний інтелект (ШІ) і, зокрема, обробка природної мови (Natural language processing або NLP) відіграють вирішальну роль у способі нашої взаємодії з технологіями. Від простих чат-агентів до складних аналітичних систем, здатних аналізувати величезні обсяги тексту. NLP змінює принцип за яким ми обмінюємося інформацією та приймаємо рішення. Однак, незважаючи на значні досягнення в галузі машинного навчання та штучного інтелекту, сучасні системи NLP все ще стикаються з обмеженнями, особливо коли справа доходить до глибокого розуміння людської мови. Для подолання обмежень потрибен пошук нових підходів та методологій, які можуть покращити ці процеси.

Ця магістерська робота зосереджується на дослідженні «Методів обробки природної мови, доповнених знаннями». Метою цього дослідження є аналіз і використання знань – структурованих даних, які можуть бути інтегровані у NLP системи для підвищення їх ефективності та точності. Використання онтологій, баз знань і графів знань може забезпечити більш глибокий контекст та семантичне розуміння, що є критично важливим для багатьох застосувань, від машинного перекладу до автоматичного резюмування текстів.

В рамках цього дослідження буде проведено огляд сучасних методів NLP, аналіз сценаріїв їх застосування, виявлено виклики та обмеження, з якими можуть зіткнутися дослідники та практики. Також будуть визначені перспективні напрямки майбутніх досліджень у цій області що динамічно розвивається.

Магістерська робота має на меті не тільки теоретично оцінити внесок знаннєвих систем у NLP, але й продемонструвати практичне застосування розроблених методів на конкретних даних та задачах, що забезпечить міцну основу для подальших досліджень та розвитку галузі.

1 ОГЛЯД СУЧАСНИХ МЕТОДІВ ОБРОБКИ ПРИРОДНОЇ МОВИ ДОПОВНЕНИХ ЗНАННЯМИ

1.1 Історичний огляд розвитку методів NLP

1.1.1 Початкові спроби обробки природної мови

Обробка природної мови (Natural Language Processing, NLP) має багату історію, що розпочинається з ранніх спроб автоматизації лінгвістичних задач ще в середині XX століття. Перші значні кроки в цьому напрямку були зроблені в 1950-х роках, коли науковці та інженери почали розробляти системи для автоматичного перекладу текстів. Однією з найвідоміших таких систем була система для автоматичного перекладу між англійською та російською мовами, розроблена в США під час холодної війни. Ця система стала відомою завдяки експерименту Джорджтаун-IBM (Georgetown-IBM experiment) у 1954 році, який демонстрував автоматичний переклад 60 речень з російської на англійську [1].

На початку свого розвитку, методи обробки природної мови були здебільшого основані на жорстко закодованих правилах (rule-based systems). Ці системи використовували лінгвістичні правила, що були вручну створені експертами, для аналізу та генерації мови. Прикладом може служити система ELIZA, розроблена в 1966 році Джозефом Вейценбаумом. ELIZA була однією з перших програм, що демонструвала можливості обробки природної мови через симуляцію психотерапевта, що відповідала на питання користувачів за допомогою простих шаблонів та правил [2].

У 1970-х роках розвиток обробки природної мови отримав новий імпульс завдяки використанню статистичних методів. Одним із піонерів цього підходу став IBM, який розробив модель на основі Марковських процесів для задачі автоматичного розпізнавання мови. Ці моделі, відомі як

приховані марковські моделі (Hidden Markov Models, HMM), стали основою для багатьох подальших досліджень та розробок у сфері NLP [3].

У 1980-х та 1990-х роках розвиток обчислювальної потужності та доступність великих корпусів текстів призвели до значного прогресу в статистичному підході до обробки природної мови. Зокрема, методи на основі максимальної правдоподібності та байєсового висновку стали широко застосовуватися для задач класифікації текстів, розпізнавання іменованих сутностей та інших задач NLP. Одним із значних досягнень цього періоду було створення WordNet, лексичної бази даних, що організовує англійські слова за їх значеннями та семантичними зв'язками, розробленої під керівництвом Джорджа Міллера в Прінстонському університеті [4].

1.1.2 Розвиток машинного навчання та його вплив на NLP

У 1980-х та 1990-х роках розвиток машинного навчання суттєво змінив підхід до обробки природної мови (NLP). Цей період ознаменувався переходом від жорстко закодованих правил до статистичних методів, які використовували великі обсяги даних для навчання моделей.

Одним із ключових проривів стало застосування прихованих марковських моделей (Hidden Markov Models, HMM) для різних задач NLP. Ці моделі стали основою для систем автоматичного розпізнавання мовлення та тегування частин мови, оскільки вони дозволяли ефективно моделювати послідовні дані, такі як текст або мовлення [3]. Використання HMM зробило можливим більш точне розуміння та генерацію мови.

На початку 1990-х років статистичні методи стали основним підходом у NLP. Наївний байєсів класифікатор (Naive Bayes) та метод опорних векторів (Support Vector Machines, SVM) стали популярними для задач класифікації тексту та розпізнавання іменованих сутностей. Наївний байєсів класифікатор, завдяки своїй простоті та ефективності, широко

застосовувався для фільтрації спаму та аналізу настроїв [5]. SVM забезпечували високу точність у задачах класифікації тексту, особливо у випадках, коли доступно обмежене число навчальних прикладів [6].

Розвиток нейронних мереж та глибокого навчання у 2000-х роках мав ще більший вплив на NLP. Рекурентні нейронні мережі (RNN), особливо їх вдосконалені варіанти, такі як довга короткочасна пам'ять (Long Short-Term Memory, LSTM), стали основним інструментом для моделювання послідовних даних. LSTM дозволили значно покращити результати у багатьох задачах NLP, таких як машинний переклад та розпізнавання мовлення, завдяки своїй здатності моделювати довгострокові залежності в текстах [7].

Прорив у використанні трансформерів став ще одним значним кроком уперед у розвитку NLP. Введені у 2017 році, моделі на основі трансформерів, такі як BERT (Bidirectional Encoder Representations from Transformers) і GPT (Generative Pre-trained Transformer), забезпечили значні покращення у задачах розуміння тексту та генерації мови [8]. Трансформери використовують механізм уваги, який дозволяє моделі фокусуватися на різних частинах вхідного тексту при обробці кожного слова, що значно покращує контекстне розуміння. Ці моделі стали основою для багатьох сучасних NLP систем, забезпечуючи високу точність та гнучкість у розв'язанні різних лінгвістичних задач.

1.1.3 Розвиток машинного навчання та його вплив на NLP

Розвиток обробки природної мови (NLP) пройшов довгий шлях від використання класичних методів до впровадження глибокого навчання та моделей великих мовних моделей (LLM). Ця еволюція відбулася завдяки зростанню обчислювальної потужності, наявності великих обсягів даних та розробці нових алгоритмів.

На початку 2000-х років, із зростанням доступності великих корпусів текстів та обчислювальних ресурсів, почали широко застосовуватися методи глибокого навчання. Рекурентні нейронні мережі (RNN) і їх вдосконалені варіанти, такі як довга короткочасна пам'ять (LSTM), стали основним інструментом для роботи з послідовними даними [7]. Ці моделі дозволили обробляти текст у контексті, зберігаючи інформацію про попередні слова у реченні, що було важливо для задач, таких як машинний переклад та автоматичне резюмування текстів.

Перехід до глибокого навчання відкрив нові можливості для створення складніших моделей, здатних розуміти та генерувати природну мову на значно вищому рівні. Одним із найважливіших досягнень цього періоду стало створення трансформерів – архітектури нейронних мереж, яка використовує механізм уваги для ефективної обробки великих текстових корпусів. Моделі на основі трансформерів, такі як BERT (Bidirectional Encoder Representations from Transformers) та GPT (Generative Pre-trained Transformer), змогли досягти значних покращень у задачах розуміння та генерації мови [9], [10].

BERT, розроблений у 2018 році, використовує двонаправлене кодування контексту, що дозволяє моделі враховувати як попередні, так і наступні слова при обробці тексту. Це значно покращило точність у задачах розпізнавання іменованих сутностей, відповіді на питання та інших [9]. GPT, з іншого боку, зосереджується на генерації мови та здатен створювати зв'язний і контекстно-логічний текст, що є надзвичайно корисним для задач автоматичного написання текстів та діалогових систем [10].

Сучасні великі мовні моделі (LLM), такі як GPT-3, що складається з 175 мільярдів параметрів, демонструють вражаючі результати у багатьох задачах NLP, включаючи переклад, створення контенту, та взаємодію з користувачами у реальному часі [11]. Ці моделі використовують величезні обсяги даних для попереднього тренування і можуть бути адаптовані до конкретних задач через тонке налаштування (fine-tuning).

1.2 Класичні методи машинного навчання

1.2.1 Naive Bayes

Naive Bayes – це один із найпростіших та найефективніших алгоритмів для класифікації тексту, який базується на теоремі Байєса. Цей метод належить до сімейства байєсівських класифікаторів і використовує припущення про незалежність між ознаками, що виявляється особливо корисним у багатьох задачах обробки природної мови.

Принцип роботи Naive Bayes базується на застосуванні теореми Байєса для обчислення ймовірності належності певного зразка до конкретного класу. Формула теореми Байєса виглядає наступним чином:

$$P(C|X) = \frac{P(X|C)\cdot P(C)}{P(X)}, \tag{1.1}$$

де $P(C|X)$ – апостеріорна ймовірність класу C при наявності ознак X;

$P(X|C)$ – ймовірність ознак X при заданому класі C;

$P(C)$ – апріорна ймовірність класу C;

$P(X)$ – ймовірність ознак X.

У випадку якщо Naive Bayes припускається, що всі ознаки X_i є незалежними між собою, що дозволяє спростити обчислення:

$$P(C|X) = \frac{P(C)\cdot \prod_{i-1}^{n} P(X_i|C)}{P(X)}. \tag{1.2}$$

Це припущення про незалежність ознак є досить спрощеним, але на практиці воно часто дає гарні результати, особливо при класифікації тексту [5].

Naive Bayes є одним із найшвидших алгоритмів класифікації, оскільки його навчання та прогнозування мають низьку обчислювальну

складність [12]. Алгоритм добре працює навіть з невеликими наборами даних. Завдяки своїй простоті та швидкості, Naive Bayes добре підходить для задач, що вимагають обробки великих обсягів текстової інформації, таких як фільтрація спаму або аналіз настроїв [13].

Головний недолік Naive Bayes полягає в припущенні про незалежність ознак, яке рідко відповідає реальним даним. Це може призводити до помилкових висновків у задачах, де ознаки мають сильні залежності між собою [14]. У порівнянні з більш складними моделями, такими як нейронні мережі або SVM, Naive Bayes має обмежену здатність до захоплення складних патернів у даних. Алгоритм може погано працювати, якщо навчальні дані містять багато шуму або нерелевантних ознак.

1.2.2 Support Vector Machines

Support Vector Machines (SVM) – це потужний метод машинного навчання, який використовується для класифікації та регресії. Він став особливо популярним у задачах класифікації тексту завдяки своїй високій точності та здатності ефективно працювати навіть з високорозмірними даними.

Основна ідея SVM полягає у знаходженні гіперплощини, яка максимально розділяє дані різних класів у багатовимірному просторі. Ця гіперплощина вибирається так, щоб максимізувати відстань (маржу) між найближчими точками (опорними векторами) різних класів. Такий підхід дозволяє забезпечити хорошу узагальнюючу здатність моделі.

Формально, для лінійно роздільних даних SVM намагається знайти гіперплощину, що задовольняє наступне рівняння:

$$w \cdot x - b = 0, \tag{1.3}$$

де w – нормальний вектор до гіперплощини;

b – зсув.

Максимізація маржі зводиться до розв'язання задачі оптимізації:

$$min\ _{w,b}\frac{1}{2}||w||^2 \tag{1.4}$$

під обмеженням:

$$y_i(w \cdot x_i - b) \geq 1 \tag{1.5}$$

для всіх навчальних прикладів (x_i, y_i), де $y_i \in \{-1,1\}$ [15]

Для нелінійно роздільних даних використовуються ядра (kernel), які дозволяють проектувати дані у високорозмірний простір, де вони стають лінійно роздільними. Популярні ядра включають поліноміальне ядро, радіальне базисне функціональне ядро (RBF) та сигмоїдне ядро [16].

SVM демонструє високу точність у задачах класифікації, особливо коли дані мають чіткі роздільні межі між класами [6]. SVM ефективно працює з даними, що мають велику кількість ознак, що робить його придатним для задач класифікації тексту [17]. Використання різних ядер дозволяє SVM вирішувати як лінійні, так і нелінійні задачі класифікації, що забезпечує високу гнучкість методу [16].

Але SVM може бути чутливим до вибору параметрів, таких як параметр регуляризації C та параметри ядра, що може вимагати значних зусиль на налаштування [18]. Навчання SVM з великими наборами даних може бути обчислювально інтенсивним, особливо при використанні складних ядр [19]. Результати SVM можуть бути важко інтерпретувати, особливо у випадках з нелінійними ядрами, де гіперплощина розділення може бути складною [20].

1.3 Методи на основі глибокого навчання

1.3.1 Long Short-Term Memory

Long Short-Term Memory (LSTM) – це тип рекурентної нейронної мережі (RNN), спеціально розроблений для подолання проблеми зникання градієнта, що є типовою для стандартних RNN. Основна ідея LSTM полягає у використанні спеціальних структур, відомих як «комірки пам'яті», які здатні зберігати інформацію протягом довгих часових відрізків.

Кожна комірка LSTM містить три основні компоненти: вхідний, вихідний та забутій гейти. Ці гейти контролюють потік інформації в і з комірки, дозволяючи моделі вибірково зберігати чи ігнорувати інформацію.

Вхідний гейт вирішує, яку частину нової інформації додати до стану комірки. Забутій гейт визначає, яку частину поточної інформації з комірки потрібно видалити. Вихідний гейт вирішує, яку частину інформації з комірки використовувати як вихідний сигнал.

Формули для обчислення цих гейтів є наступними:

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i), \tag{1.6}$$

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f), \tag{1.7}$$

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o), \tag{1.8}$$

$$C_t = f_t * C_{t-1} + i_t * \tanh\,(W_C \cdot [h_{t-1}, x_t] + b_C), \tag{1.9}$$

$$h_t = o_t * \tanh\,(C_t), \tag{1.10}$$

де σ – це сигмоїдна функція активації;

$\tanh$ – гіперболічна тангенс-функція;

W – вагові матриці;

b – зміщення [7].

LSTM знайшли широке застосування в задачах обробки природної мови завдяки своїй здатності ефективно обробляти послідовні дані. LSTM використовуються в архітектурах нейронних мереж для перекладу тексту з однієї мови на іншу [21]. Завдяки своїй здатності зберігати довготривалі залежності, LSTM добре підходять для задач розпізнавання мовлення [22]. Також LSTM застосовуються для класифікації текстів за їх настроєм, що корисно для аналізу відгуків, соціальних медіа та інших джерел [23]. LSTM використовуються для створення коротких резюме великих текстів, зберігаючи основні ідеї та зміст [24].

Основна перевага LSTM полягає у їх здатності ефективно зберігати та використовувати інформацію протягом довгих часових відрізків [7]. LSTM можуть працювати з різними типами послідовних даних, що робить їх універсальним інструментом для багатьох задач NLP [21]. LSTM менш чутливі до шуму у вхідних даних порівняно зі стандартними RNN, завдяки механізмам гейтів [22].

Нажаль навчання моделей LSTM може бути обчислювально інтенсивним, що вимагає значних ресурсів, особливо для великих наборів даних [23]. Також LSTM мають багато гіперпараметрів, які потрібно налаштовувати, що може ускладнити процес навчання [24]. Як і багато інших глибоких моделей, LSTM є "чорними ящиками", і інтерпретація їхніх рішень може бути складною [15].

1.3.2 Gated Recurrent Unit

Gated Recurrent Unit (GRU) – це тип рекурентної нейронної мережі (RNN), який був розроблений для зменшення складності LSTM, зберігаючи при цьому здатність моделювати довготривалі залежності в

послідовних даних. GRU було запропоновано Кюнгхюном Чо та його колегами у 2014 році [26].

Основна структура GRU включає два гейти: гейт оновлення та гейт скидання. Ці гейти спрощують архітектуру та обчислення, зменшуючи кількість параметрів, необхідних для навчання.

Гейт оновлення визначає, яку частину попереднього стану потрібно зберегти. Гейт скидання визначає, яку частину попереднього стану потрібно забути або скинути.

Формули для обчислення цих гейтів є наступними:

$$z_t = \sigma(W_z \cdot [h_{t-1}, x_t]), \tag{1.11}$$

$$r_t = \sigma(W_r \cdot [h_{t-1}, x_t]), \tag{1.12}$$

$$\widetilde{h_t} = tanh(W \cdot [r_t * h_{t-1}, x_t]), \tag{1.13}$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * h_t, \tag{1.14}$$

де z_t – вектор оновлення;

r_t – вектор скидання;

h_t – новий стан;

σ – сигмоїдна функція активації;

а $tanh$ – гіперболічна тангенс-функція [27].

GRU широко використовуються в задачах обробки природної мови завдяки своїй здатності ефективно моделювати послідовні дані при меншій обчислювальній складності порівняно з LSTM. GRU використовуються в системах машинного перекладу для моделювання послідовностей слів, що дозволяє досягати високої точності перекладу [28]. GRU застосовуються для задач розпізнавання мовлення завдяки своїй здатності обробляти довгі послідовності звукових сигналів [29]. GRU використовуються для аналізу

настроїв у текстах, таких як відгуки користувачів та повідомлення в соціальних мережах [23]. GRU застосовуються для автоматичної генерації текстів, включаючи створення діалогів та написання статей [10].

GRU мають менш складну архітектуру порівняно з LSTM, що знижує кількість параметрів та обчислювальні витрати [26]. Завдяки меншій кількості параметрів, GRU швидше навчаються та потребують менше ресурсів для тренування [27]. Також GRU добре підходять для задач, де важливо зберігати довготривалі залежності в послідовностях [28].

Але у деяких випадках GRU можуть бути менш ефективними, ніж LSTM, для моделювання дуже складних залежностей у даних [29]. Хоча GRU використовуються в багатьох дослідженнях, вони не завжди стандартизовані так само, як LSTM, що може створювати труднощі в їх застосуванні [23]. Як і інші глибокі моделі, GRU є «чорними ящиками», і їх внутрішні процеси важко інтерпретувати [10].

1.4 Моделі на основі трансформерів

1.4.1 Bidirectional Encoder Representations from Transformers

Bidirectional Encoder Representations from Transformers (BERT) – це революційна модель, розроблена Google, яка використовує двонаправлене кодування для обробки тексту. Відмінною рисою BERT є її здатність враховувати контекст слова як зліва, так і справа одночасно, що забезпечує глибше розуміння тексту порівняно з однонаправленими моделями.

Основою архітектури BERT є трансформер, який складається з енкодерів, що використовують механізм самоуваги. Цей механізм дозволяє моделі оцінювати важливість кожного слова у контексті всього речення, що дозволяє більш точно розуміти значення слів.

BERT навчається за допомогою двох основних задач: Masked Language Model (MLM) та Next Sentence Prediction (NSP).

В MLM деякі слова у реченні маскуються, і модель повинна передбачити ці слова на основі контексту. Це дозволяє моделі навчитися розуміти контекст з обох боків слова. В NSP модель навчається визначати, чи є одне речення продовженням іншого, що покращує її здатність розуміти зв'язки між реченнями [9].

BERT знайшов широке застосування у багатьох задачах обробки природної мови завдяки своїй здатності забезпечувати високу точність та розуміння контексту. BERT використовується для виявлення і класифікації іменованих сутностей у тексті, таких як імена, дати та місця [30]. Модель застосовується для класифікації текстів за настроєм, що є корисним для аналізу відгуків та соціальних медіа [31]. BERT використовується в системах питань-відповідей, забезпечуючи точні відповіді на запитання на основі контексту [32]. Модель застосовується для задач машинного перекладу, забезпечуючи високоякісний переклад між різними мовами [33].

BERT враховує контекст з обох боків слова, що забезпечує глибше розуміння тексту [9]. Модель може бути налаштована для різних задач NLP без необхідності зміни її основної архітектури [30]. Висока точність: BERT демонструє високі результати у багатьох стандартних задачах NLP, таких як розпізнавання іменованих сутностей та системи питань-відповідей [32].

Навчання та використання BERT вимагає значних обчислювальних ресурсів, що може бути недоступним для багатьох організацій [33]. Модель потребує великого обсягу даних для ефективного навчання, що може бути складним завданням для специфічних доменів. Навчання BERT займає значний час, навіть при використанні потужних обчислювальних кластерів [9].

1.4.2 Generative Pre-trained Transformer

Generative Pre-trained Transformer (GPT) – це модель на основі трансформерів, розроблена компанією OpenAI, яка орієнтована на

генерацію тексту. Основна відмінність GPT від інших моделей полягає в його підході до попереднього навчання і трансформерній архітектурі, яка дозволяє ефективно обробляти великі обсяги тексту.

GPT використовує архітектуру трансформера, але на відміну від BERT, вона орієнтована на авто-регресивне моделювання. Це означає, що модель генерує текст послідовно, передбачаючи наступне слово на основі попередніх слів. Основні компоненти архітектури включають: самоувага – механізм самоуваги дозволяє моделі враховувати всі попередні слова у послідовності для генерації наступного слова, фід-форвардні мережі – кожне слово проходить через фід-форвардні нейронні мережі для подальшої обробки.

GPT навчається у два етапи. Попереднє навчання: модель тренується на великому корпусі текстів для передбачення наступного слова в реченні. Це дозволяє моделі зрозуміти структуру мови та семантичні зв'язки між словами. Тонке налаштування (fine-tuning): після попереднього навчання модель додатково налаштовується на специфічних задачах, таких як питання-відповідь, переклад або класифікація текстів [10].

GPT знайшов широке застосування в різних задачах обробки природної мови завдяки своїй здатності генерувати зв'язний та контекстуально-логічний текстю GPT використовується для створення статей, історій та інших типів контенту, що вимагають природного мовлення [34]. Модель застосовується в системах питань-відповідей, забезпечуючи точні відповіді на основі контексту [11]. GPT використовується для машинного перекладу текстів, забезпечуючи високоякісний переклад між різними мовами [30]. Завдяки здатності генерувати природну мову, GPT використовується в чат-ботах для забезпечення інтерактивної взаємодії з користувачами [35].

GPT здатний генерувати зв'язний та контекстно-логічний текст, що робить його корисним для різних задач, пов'язаних з мовленням. Модель може бути налаштована для різних задач NLP без необхідності зміни її

основної архітектури [10]. GPT може бути використаний для широкого спектру задач, від генерації текстів до перекладу та розробки чат-ботів [34].

Навчання та використання GPT вимагає значних обчислювальних ресурсів, що може бути недоступним для багатьох організацій [11]. Модель потребує великого обсягу даних для ефективного навчання, що може бути складним завданням для специфічних доменів [30]. GPT може генерувати текст, який виглядає правдоподібно, але є неправдивим або некоректним, що вимагає додаткового контролю [35].

1.5 Аналіз недоліків класичних методів та методів глибокого навчання

1.5.1 Обмеження в контексті обробки складних лінгвістичних конструкцій

Класичні методи машинного навчання, такі як Naive Bayes і SVM, мають суттєві обмеження в контексті обробки складних лінгвістичних конструкцій. Ці методи базуються на простих статистичних моделях, які не враховують складні залежності між словами у реченні. Наприклад, Naive Bayes припускає, що всі ознаки (слова) є незалежними, що рідко відповідає реальності у природній мові. Це може призводити до проблем з точністю при обробці текстів, які містять складні граматичні структури або багатозначні слова [5].

Методи на основі глибокого навчання, такі як LSTM та GRU, були розроблені для подолання деяких з цих обмежень, зокрема шляхом зберігання та обробки довготривалих залежностей у тексті. Проте, навіть ці моделі мають свої недоліки. Незважаючи на їх здатність враховувати попередній контекст, вони можуть мати труднощі з обробкою дуже довгих послідовностей через проблему зникання градієнта, яка обмежує їх ефективність при роботі з дуже довгими текстами [7].

Крім того, моделі на основі LSTM і GRU не завжди ефективно справляються з обробкою складних синтаксичних структур, таких як вкладені речення або довгі зв'язки між словами. Це пов'язано з тим, що ці моделі, хоча і можуть зберігати контекст на певному рівні, все ж таки залежать від порядку слів і можуть втрачати важливу інформацію при обробці дуже складних лінгвістичних конструкцій [36].

Новіші моделі, такі як трансформери, зокрема BERT і GPT, значно покращили здатність обробляти складні лінгвістичні конструкції завдяки механізму уваги, який дозволяє моделі фокусуватися на різних частинах тексту незалежно від їх позиції. Проте навіть ці моделі можуть стикатися з труднощами при обробці дуже довгих текстів або текстів з дуже складною граматикою, що потребує додаткових ресурсів для навчання та обробки [9].

Таким чином, незважаючи на значний прогрес у розвитку методів обробки природної мови, обмеження у контексті обробки складних лінгвістичних конструкцій залишаються актуальними як для класичних методів, так і для методів глибокого навчання.

1.5.2 Відсутність врахування семантичних та синтаксичних особливостей мови

Класичні методи машинного навчання, такі як Naive Bayes і SVM, часто не враховують семантичні та синтаксичні особливості мови, що є важливим для глибокого розуміння тексту. Методи, засновані на мішку слів (bag-of-words) або TF-IDF, ігнорують порядок слів та зв'язки між ними, що призводить до втрати критичної інформації про контекст і структуру речень. Це може бути проблематичним при обробці складних текстів, де граматичні структури та семантичні зв'язки між словами відіграють ключову роль [37].

Навіть більш просунуті методи глибокого навчання, такі як LSTM та GRU, мають обмеження у врахуванні семантичних і синтаксичних особливостей. Хоча ці моделі можуть зберігати інформацію про попередні

слова у послідовності, вони часто мають труднощі з обробкою складних синтаксичних структур, таких як вкладені речення або довгі залежності між словами [36]. Це пов'язано з тим, що ці моделі, хоча і можуть зберігати контекст на певному рівні, все ж таки залежать від порядку слів і можуть втрачати важливу інформацію при обробці дуже складних лінгвістичних конструкцій.

Новітні моделі на основі трансформерів, такі як BERT і GPT, значно покращили врахування семантичних та синтаксичних особливостей завдяки механізму самоуваги. Цей механізм дозволяє моделі фокусуватися на різних частинах тексту незалежно від їх позиції, що покращує розуміння складних лінгвістичних конструкцій [9]. Проте, навіть ці моделі можуть мати обмеження у випадках, коли необхідно враховувати дуже специфічні синтаксичні структури або складні семантичні зв'язки, які не були присутні у навчальних даних.

Крім того, трансформерні моделі потребують значних обчислювальних ресурсів для навчання і налаштування, що може бути обмежуючим фактором для багатьох організацій. Це також ускладнює їх використання для задач, де доступні обмежені обсяги даних або де необхідна висока швидкість обробки [11].

1.5.3 Проблеми з генералізацією на нові домени

Одним з основних недоліків як класичних методів машинного навчання, так і методів глибокого навчання є їх обмежена здатність до генералізації на нові домени. Це означає, що моделі, натреновані на одному наборі даних, можуть демонструвати значне зниження продуктивності при застосуванні до даних, які суттєво відрізняються від навчальних.

Класичні методи, такі як Naive Bayes і SVM, часто стикаються з проблемами, коли їм потрібно працювати з новими доменами, оскільки вони сильно залежать від характеристик навчального набору даних. Наприклад,

Naive Bayes, що використовує прості ймовірнісні припущення, може не впоратися з новими, більш складними або різноманітними даними, оскільки його гнучкість обмежена [5]. SVM також можуть мати труднощі з генералізацією через свою чутливість до вибору гіперпараметрів і структури даних [6].

Методи глибокого навчання, такі як LSTM і GRU, мають більшу гнучкість порівняно з класичними методами, але вони також не завжди ефективно адаптуються до нових доменів. Хоча ці моделі можуть зберігати контекст і обробляти послідовності, їх ефективність сильно залежить від обсягу та якості навчальних даних. Наприклад, моделі LSTM, натреновані на текстах новин, можуть погано працювати з текстами з інших доменів, таких як технічна документація або медичні звіти, через відмінності в стилі, лексиконі та структурі [38].

Трансформерні моделі, такі як BERT і GPT, також стикаються з проблемами генералізації. Незважаючи на їх здатність враховувати контекст та обробляти складні лінгвістичні конструкції, вони потребують значних обчислювальних ресурсів для перенавчання або тонкого налаштування на нові домени. Це робить їх адаптацію до нових задач ресурсозатратною і часоємною [9]. Крім того, навіть при тонкому налаштуванні, ці моделі можуть не завжди ефективно враховувати специфічні особливості нових доменів, що може призводити до зниження точності [11].

Однією з причин таких проблем є те, що навчальні дані часто не охоплюють всю різноманітність мовних конструкцій і контекстів, які можуть зустрічатися в реальному світі. Це робить моделі вразливими до зниження продуктивності при зустрічі з новими або незвичними даними [39].

1.6 Обмеження моделей LLM без додавання знань

1.6.1 Нестабільність результатів без зовнішнього знання

Великі мовні моделі (LLM), такі як GPT і BERT, демонструють значний прогрес у багатьох задачах обробки природної мови, проте їх результати можуть бути нестабільними без зовнішнього знання. Основна причина цього полягає в тому, що LLM базуються виключно на інформації, отриманій під час попереднього навчання, і не мають доступу до актуальних або специфічних знань, які можуть бути необхідними для точного розуміння та обробки деяких запитів.

Нестабільність результатів LLM може проявлятися у різних формах.

Відсутність актуальності інформації. Оскільки моделі тренуються на статичних наборах даних, вони можуть не враховувати останні події або нові знання, що робить їх відповіді менш точними або застарілими. Наприклад, модель GPT-3, натренована до 2021 року, не має інформації про події, що сталися після цього періоду [11].

Некоректність відповідей: LLM можуть генерувати відповіді, що виглядають правдоподібно, але є неправдивими або некоректними. Це пов'язано з тим, що моделі намагаються створювати зв'язний текст на основі статистичних закономірностей у навчальних даних, але без зовнішнього валідаційного механізму можуть робити помилки [40].

Обмеження в спеціалізованих доменах: моделі часто демонструють недостатню точність у вузькоспеціалізованих областях, де потрібні глибокі знання та експертиза. Наприклад, при обробці медичних текстів або юридичних документів, LLM можуть давати помилкові або поверхневі відповіді без додаткових знань із цих доменів [41].

Вразливість до контекстних змін: LLM можуть некоректно реагувати на запити, які вимагають глибокого контекстного розуміння або знань про

попередні частини розмови. Це обмежує їх здатність до підтримки зв'язних та логічних діалогів у тривалих взаємодіях [42].

Для подолання цих обмежень існують підходи, що включають додавання зовнішнього знання до моделей LLM. Наприклад, інтеграція з базами знань або використання механізмів валідації відповідей за допомогою зовнішніх джерел інформації може значно підвищити точність і стабільність результатів [43].

1.6.2 Проблеми з розумінням контексту та багатозначності

Великі мовні моделі (LLM), такі як GPT і BERT, демонструють вражаючі результати в багатьох задачах обробки природної мови. Однак, вони часто стикаються з труднощами при обробці текстів, де контекст і багатозначність мають критичне значення. Ці проблеми виникають через обмеження в здатності моделей до повного розуміння контексту та точного визначення значень слів у різних ситуаціях.

Контекстна залежність. LLM часто покладаються на локальний контекст для прийняття рішень, що може призводити до помилок у випадках, коли розуміння вимагає врахування ширшого контексту. Наприклад, моделі можуть неправильно інтерпретувати слова, які змінюють своє значення залежно від попередніх речень або абзаців [11]. Це обмежує їх здатність ефективно обробляти складні тексти з довготривалими залежностями.

Багатозначність слів. Моделі часто мають труднощі з розрізненням багатозначних слів, які можуть мати кілька значень залежно від контексту. Наприклад, слово «bank» може означати як фінансову установу, так і берег річки. Без додаткових знань або контексту модель може зробити неправильний вибір значення [44].

Прості синтаксичні структури. LLM іноді мають тенденцію до спрощення синтаксичних структур, що призводить до втрати точності в

розумінні складних граматичних конструкцій. Це може бути особливо проблематичним у технічних текстах або наукових статтях, де точне розуміння структурної взаємодії між словами є критичним [37].

Недостатність доменної специфічності. Моделі, треновані на широких наборах даних, можуть не враховувати специфічні контекстні вимоги вузьких доменів. Наприклад, у медичних текстах або юридичних документах точне розуміння термінології та контексту має вирішальне значення, і без додаткових знань моделі можуть давати некоректні результати [41].

Обмеження попереднього навчання. Навіть найсучасніші моделі, такі як GPT-3, тренуються на великому, але все ж обмеженому наборі даних. Це означає, що вони можуть не мати знань про всі можливі контексти і багатозначності, з якими вони стикаються у реальних застосуваннях [40].

Для подолання цих обмежень існують підходи, що включають інтеграцію зовнішніх баз знань, спеціалізованих моделей та механізмів валідації, що можуть допомогти моделі краще розуміти контекст та розрізняти багатозначні слова [43]. Це забезпечує більш стабільні та точні результати в обробці текстів у різних доменах.

1.6.3 Відсутність можливості логічного висновку без додаткової інформації

Великі мовні моделі (LLM), такі як GPT і BERT, демонструють вражаючі результати у багатьох задачах обробки природної мови, проте вони мають суттєве обмеження – нездатність до логічного висновку без додаткової інформації. Це обмеження стає особливо критичним у випадках, коли необхідно виконати складні логічні операції або розуміти причинно-наслідкові зв'язки.

Недостатність логічних міркувань. LLM переважно базуються на статистичних закономірностях у великих наборах текстових даних, що

дозволяє їм добре виконувати завдання, пов'язані з передбаченням тексту. Проте вони не мають внутрішнього механізму для виконання складних логічних міркувань або дедуктивних висновків. Це обмежує їх здатність до вирішення задач, що вимагають чіткої логіки та розуміння контекстуальних зв'язків [40].

Відсутність каузального розуміння. Моделі не здатні інтерпретувати причинно-наслідкові зв'язки без додаткових знань або контексту. Наприклад, модель може згенерувати текст, що виглядає логічно, але не здатна правильно визначити причинно-наслідкові зв'язки між подіями, що робить її відповіді неточними у складних ситуаціях [45].

Залежність від навчальних даних. Логічні висновки у LLM обмежені тими знаннями, які були закладені у них під час навчання. Це означає, що моделі можуть не враховувати інформацію, яка не була представлена у навчальних даних, що призводить до поверхневих або помилкових висновків у нових контекстах [46].

Недосконаліть у обробці формальних знань. Моделі часто стикаються з труднощами при обробці формальних знань, таких як математичні або логічні задачі. Це обмеження випливає з їхньої архітектури, яка не призначена для виконання формальних логічних операцій, що є необхідними для вирішення таких задач [47].

Обмеження у використанні зовнішніх знань: для подолання цих обмежень існують підходи, що включають інтеграцію LLM з зовнішніми базами знань або системами логічного висновку. Проте ці інтеграції можуть бути складними та вимагати значних обчислювальних ресурсів, що ускладнює їх широке застосування [43].

2 РОЗРОБКА ЗАСОБІВ ОБРОБКИ ПРИРОДНОЇ МОВИ ДОПОВНЕНИХ ЗНАННЯМИ

2.1 Необхідність додавання експертних знань

2.1.1 Аргументація на користь інтеграції знань у процес обробки природної мови

Включення експертних знань у процес обробки природної мови (NLP) є важливим кроком для покращення точності та надійності мовних моделей. Аргументація на користь інтеграції таких знань ґрунтується на декількох ключових аспектах.

Покращення розуміння контексту та багатозначності. Мовні моделі часто стикаються з труднощами при розумінні багатозначних слів та складних контекстів. Інтеграція експертних знань може допомогти моделі більш точно визначати значення слів залежно від контексту. Наприклад, знання про те, що слово «bank» може означати як фінансову установу, так і берег річки, допоможе моделі правильно інтерпретувати текст залежно від контексту [48].

Забезпечення логічного висновку. Мовні моделі без доступу до експертних знань мають обмежену здатність до виконання логічних висновків. Додавання експертних знань, таких як правила логіки або доменні знання, може значно покращити здатність моделі робити коректні висновки. Це особливо важливо для задач, що вимагають глибокого розуміння логічних зв'язків між елементами тексту [43].

Підвищення точності в спеціалізованих доменах. Моделі, що працюють з вузькоспеціалізованими текстами, такими як медичні або юридичні документи, часто потребують глибоких знань з відповідного домену. Інтеграція таких знань допоможе моделі краще розуміти

специфічну термінологію та контекст, що підвищить точність і надійність обробки текстів у цих галузях [49].

Можливість обробки складних лінгвістичних конструкцій. Експертні знання можуть включати правила синтаксису та семантики, які допоможуть моделі обробляти складні лінгвістичні конструкції. Це особливо важливо для текстів, де граматичні структури є складними і де потрібно враховувати різні рівні мовної ієрархії [50].

Покращення генералізації. Інтеграція експертних знань може покращити здатність моделі до генералізації на нові домени. Моделі, що мають доступ до широкого спектру знань, краще адаптуються до нових контекстів і можуть більш точно обробляти раніше невідомі дані. Це важливо для забезпечення стабільності та надійності результатів при зміні умов використання [39].

Отже інтеграція експертних знань у процес обробки природної мови необхідна для подолання обмежень сучасних мовних моделей. Вона дозволяє забезпечити більш високу точність, надійність та адаптивність моделей у різних задачах і доменах.

2.1.2 Переваги використання експертних знань для покращення результатів NLP

Інтеграція експертних знань у процес обробки природної мови (NLP) приносить значні переваги, що дозволяє покращити точність, надійність та загальну ефективність моделей.

Підвищення точності та надійності. Інтеграція експертних знань допомагає моделям NLP краще розуміти контекст і значення слів, що призводить до більш точних результатів. Наприклад, додавання семантичної інформації може покращити результати розпізнавання іменованих сутностей, що є критичним для багатьох застосувань, таких як аналіз тексту в фінансовій або медичній галузях [51]. Це також знижує

ймовірність помилок при інтерпретації багатозначних слів або складних синтаксичних конструкцій.

Поліпшення генералізації. Знання про різні домени дозволяють моделям краще адаптуватися до нових контекстів і задач. Наприклад, моделі, які мають доступ до медичних або юридичних знань, можуть точніше інтерпретувати специфічну термінологію та контексти в цих галузях. Це робить моделі більш універсальними та здатними до генералізації на нові домени [39].

Забезпечення логічних висновків. Додавання експертних знань у вигляді логічних правил або структурованих баз знань покращує здатність моделей робити логічні висновки. Це особливо важливо для задач, які потребують глибокого розуміння причинно-наслідкових зв'язків або формальних міркувань. Наприклад, використання онтологій і правил логіки дозволяє моделям точніше відповідати на складні запити [43].

Оптимізація обробки складних лінгвістичних конструкцій. Експертні знання можуть включати правила синтаксису та семантики, що допомагають моделям ефективніше обробляти складні лінгвістичні конструкції. Це дозволяє моделі краще розуміти структуру речень і взаємодії між словами, що підвищує точність у завданнях, таких як машинний переклад або аналіз настроїв [50].

Покращення інтерпретації результатів. Інтеграція експертних знань робить результати моделей більш інтерпретованими і зрозумілими для користувачів. Це особливо важливо для застосувань у критичних галузях, таких як медицина або право, де точність і прозорість рішень мають вирішальне значення. Наприклад, додавання медичних знань до моделі NLP дозволяє краще зрозуміти результати аналізу медичних текстів [49].

2.2 Концепція Knowledge-Augmented NLP

2.2.1 Основні принципи

Концепція Knowledge-Augmented NLP (KA-NLP) полягає в інтеграції експертних знань з методами обробки природної мови для покращення точності, надійності та інтерпретованості моделей.

Інтеграція зовнішніх знань. Одним з головних принципів KA-NLP є використання зовнішніх джерел знань, таких як бази знань, онтології та спеціалізовані доменні дані. Це дозволяє моделям краще розуміти контекст і значення слів у різних ситуаціях. Наприклад, додавання медичних знань до моделі NLP може значно покращити результати аналізу медичних текстів [49].

Використання семантичних і синтаксичних правил. KA-NLP також включає використання семантичних і синтаксичних правил, що допомагають моделям краще розуміти структуру речень та взаємозв'язки між словами. Це особливо важливо для задач, де точне розуміння граматичних конструкцій має вирішальне значення, таких як машинний переклад або розпізнавання іменованих сутностей [51].

Забезпечення логічного висновку. Ще одним важливим принципом KA-NLP є забезпечення здатності моделей до виконання логічних висновків. Це досягається шляхом інтеграції логічних правил та структурованих баз знань, що дозволяє моделям робити коректні висновки на основі наявної інформації. Такий підхід є критичним для задач, що вимагають глибокого розуміння причинно-наслідкових зв'язків [43].

Гнучкість і адаптивність. KA-NLP моделі повинні бути гнучкими та адаптивними до різних доменів і задач. Це означає, що вони повинні легко інтегрувати нові знання і адаптуватися до змін у контексті використання. Використання методів трансферного навчання дозволяє моделям швидко

адаптуватися до нових даних та умов, підвищуючи їх ефективність і універсальність [39].

Інтерпретованість результатів. Одним з ключових аспектів KA-NLP є інтерпретованість результатів. Інтеграція експертних знань допомагає зробити результати моделей більш зрозумілими і прозорими для користувачів. Це особливо важливо для застосувань у критичних галузях, таких як медицина або право, де точність і прозорість рішень мають вирішальне значення [40].

Завдяки цим принципам, KA-NLP моделі можуть забезпечувати більш точні, надійні та інтерпретовані результати, що робить їх ефективними для широкого спектра застосувань у різних доменах.

2.2.2 Методи інтеграції знань у моделі NLP

Інтеграція знань у моделі обробки природної мови (NLP) є ключовим аспектом концепції Knowledge-Augmented NLP (KA-NLP). Існує кілька методів, які дозволяють ефективно включати знання в моделі NLP, підвищуючи їх точність, надійність та інтерпретованість.

Використання баз знань. Одним з основних методів інтеграції знань є використання баз знань, таких як Wikidata, DBpedia або спеціалізовані доменні бази. Ці бази знань містять структуровану інформацію про різні об'єкти та їх взаємозв'язки, що дозволяє моделям NLP краще розуміти контекст і значення слів. Наприклад, моделі можуть використовувати знання з Wikidata для покращення точності розпізнавання іменованих сутностей [52].

Інтеграція онтологій. Онтології надають формальний опис знань у певній області, включаючи терміни та їх взаємозв'язки. Інтеграція онтологій у моделі NLP допомагає краще структурувати інформацію та забезпечувати логічні висновки. Онтології можуть бути використані для поліпшення

семантичного розуміння тексту та забезпечення коректності результатів [53].

Методи на основі семантичних векторних просторів, такі як word embeddings (Word2Vec, GloVe) та contextual embeddings (BERT, ELMo), дозволяють моделям захоплювати семантичну інформацію про слова на основі їх використання в контексті. Це допомагає моделям розуміти значення слів у різних контекстах та підвищувати точність задач, таких як класифікація тексту та машинний переклад [54].

Інший підхід полягає у використанні комбінованих моделей, що поєднують традиційні методи машинного навчання з правилами, створеними експертами. Це дозволяє використовувати переваги як машинного навчання, так і експертних знань. Наприклад, комбіновані моделі можуть використовувати правила для фільтрації або коригування результатів, отриманих від машинного навчання [55].

Трансферне навчання передбачає перенесення знань, отриманих на одних задачах, для вирішення інших задач. Додавання знань у цьому контексті може покращити здатність моделі до генералізації на нові домени. Наприклад, моделі, попередньо натреновані на загальних текстових корпусах, можуть бути додатково натреновані з використанням спеціалізованих баз знань для досягнення кращих результатів у вузькоспеціалізованих задачах [39].

Інтеграція логічних правил у моделі NLP допомагає забезпечити коректні висновки та підвищити інтерпретованість результатів. Логічні правила можуть бути використані для забезпечення правильності логічних висновків та усунення помилок, що виникають через недостатнє розуміння контексту [43].

2.2.3 Переваги та проблеми підходу Knowledge-Augmented NLP

Інтеграція знань у моделі обробки природної мови (NLP) приносить значні переваги, але також створює певні виклики. Підхід Knowledge-Augmented NLP (KA-NLP) дозволяє покращити точність і надійність мовних моделей, проте його реалізація може бути складною.

Переваги підходу Knowledge-Augmented NLP.

Підвищена точність та надійність. Інтеграція експертних знань допомагає моделям краще розуміти контекст і значення слів, що призводить до більш точних результатів. Це особливо важливо для спеціалізованих доменів, де термінологія та контексти можуть бути складними та специфічними. Наприклад, у медичних додатках додавання знань про медичні терміни та взаємозв'язки може значно підвищити точність діагностичних систем [49].

Покращене розуміння контексту. Використання знань дозволяє моделям краще враховувати контекст і багатозначність слів. Це допомагає уникнути помилкових інтерпретацій та підвищує загальну ефективність моделі. Наприклад, у системах машинного перекладу знання про культурні та історичні контексти можуть покращити якість перекладів [56].

Можливість логічного висновку. Інтеграція логічних правил і структурованих баз знань дозволяє моделям робити коректні висновки на основі наявної інформації. Це критично для задач, які потребують розуміння причинно-наслідкових зв'язків або виконання складних логічних операцій, таких як системи автоматичного заповнення податкових декларацій [43].

Підвищена інтерпретованість. Додавання експертних знань робить результати моделей більш прозорими і зрозумілими для користувачів. Це особливо важливо для критичних галузей, де рішення моделей повинні бути легко інтерпретованими, як у медицині або праві [40].

Проблеми підходу Knowledge-Augmented NLP.

Складність інтеграції. Інтеграція зовнішніх знань у моделі NLP може бути складним процесом, що вимагає значних обчислювальних ресурсів та часу. Це включає в себе збір, структурування та актуалізацію знань, що може бути ресурсозатратним і складним завданням [57].

Застарівання знань. Знання, інтегровані у моделі, можуть швидко застарівати, особливо у швидкозмінних доменах, таких як медицина або технології. Це вимагає постійного оновлення баз знань, що додає додаткові витрати та складності [58].

Сумісність знань. Забезпечення сумісності різних джерел знань може бути викликом. Відмінності у форматах даних, структурах і термінології можуть створювати труднощі при інтеграції знань з різних джерел, що впливає на ефективність моделей [59].

Обмеженість доступу до спеціалізованих знань. У деяких доменах доступ до високоякісних знань може бути обмеженим або дорогим. Це може обмежити можливості для інтеграції знань у моделі NLP і знизити їх ефективність у спеціалізованих задачах [55].

Отже Knowledge-Augmented NLP надає численні переваги, але також створює має певні недоліки, які потребують подальших досліджень і розробок для їх подолання.

2.3 Інтеграція знань у моделі машинного навчання

2.3.1 Використання графів знань

Графи знань є потужним інструментом для збагачення моделей обробки природної мови (NLP) завдяки їх здатності організовувати та структурувати великі обсяги інформації. Вони представляють знання у вигляді вузлів (об'єктів) та ребер (взаємозв'язків), що дозволяє моделям машинного навчання ефективніше використовувати контекст і семантичні зв'язки між даними.

Графи знань структурують інформацію у вигляді триплетів «суб'єкт-предикат-об'єкт» (наприклад, «Париж – столиця – Франція»), що дозволяє створювати складні семантичні мережі. Ці мережі можуть бути використані для різних задач NLP, таких як розпізнавання іменованих сутностей, заповнення пропусків у даних, відповіді на запитання та багато інших.

Графи знань можуть значно покращити системи відповіді на запитання. Наприклад, система DBpedia використовує граф знань для надання точних відповідей на запити користувачів, витягуючи інформацію з Wikipedia. Такий підхід дозволяє системам QA швидко і ефективно знаходити релевантні відповіді, використовуючи структуровані дані з графів знань.

Інтеграція графів знань може покращити точність розпізнавання іменованих сутностей. Наприклад, використання Wikidata для збагачення NER моделей дозволяє краще розпізнавати сутності та їх атрибути, враховуючи семантичні зв'язки між словами [52].

Графи знань також використовуються для покращення семантичного пошуку. Системи, такі як Google's Knowledge Graph, використовують графи знань для забезпечення більш точних і контекстуально релевантних результатів пошуку, що покращує загальну користувацьку взаємодію.

Графи знань можуть бути інтегровані з мовними моделями для покращення їх продуктивності. Наприклад, моделі BERT і GPT-3 можуть бути збагачені знаннями з графів, що дозволяє їм краще розуміти контекст і забезпечувати більш точні результати

Інтеграція графів знань у моделі NLP вимагає значних зусиль щодо збору, обробки та підтримки актуальності даних. Важливо забезпечити сумісність різних джерел знань та ефективну синхронізацію між ними. Проте, переваги, які надають графи знань, роблять ці зусилля виправданими, оскільки вони значно покращують точність, надійність та інтерпретованість моделей NLP.

2.3.2 Онтології та їх роль у NLP

Онтології є важливим інструментом для структурування інформації та покращення розуміння тексту в обробці природної мови (NLP). Вони забезпечують формалізований опис знань у певній області, включаючи терміни та їх взаємозв'язки, що дозволяє моделям машинного навчання ефективніше використовувати ці знання для розв'язання різних задач.

Онтології визначають набір концептів і категорій у певній області, а також специфікують взаємозв'язки між цими концептами. Вони можуть включати такі елементи, як класи (або об'єкти), атрибути, відношення та правила, що дозволяють моделювати складні взаємозв'язки між об'єктами. Наприклад, онтологія в медицині може включати поняття, як-от захворювання, симптоми, процедури та ліки, а також відношення між ними.

Інтеграція онтологій у системи розпізнавання іменованих сутностей допомагає покращити точність розпізнавання. Наприклад, медичні онтології, такі як SNOMED CT або MeSH, використовуються для збагачення моделей NER, дозволяючи їм краще розпізнавати медичні терміни та їх атрибути. Це дозволяє системам точніше ідентифікувати медичні сутності та їх взаємозв'язки у текстах.

Онтології також використовуються у системах питань-відповідей для покращення розуміння контексту запитів та надання точних відповідей. Наприклад, онтологія в галузі права може допомогти системі QA правильно інтерпретувати юридичні терміни та взаємозв'язки між ними, забезпечуючи точні відповіді на юридичні запити.

Онтології використовуються для покращення семантичного пошуку, забезпечуючи більш точні результати пошуку завдяки розумінню семантичних зв'язків між термінами. Наприклад, бібліотека онтологій у сфері біомедицини BioPortal дозволяє покращити результати пошуку в біомедичних текстах, забезпечуючи точне врахування складних наукових термінів і їх взаємозв'язків [53].

Онтології також використовуються для автоматичного анотування тексту, що допомагає структурувати інформацію та робити її більш доступною для аналізу. Наприклад, онтологія Gene Ontology використовується для анотування геномних даних, що дозволяє автоматично ідентифікувати та класифікувати гени за їх функціями.

Інтеграція онтологій у моделі NLP вимагає значних зусиль щодо створення, підтримки та оновлення онтологій. Це включає забезпечення сумісності різних онтологій, синхронізацію їх з актуальними знаннями та інтеграцію у системи NLP. Проте, переваги, які надають онтології, роблять ці зусилля виправданими, оскільки вони значно покращують точність, надійність та інтерпретованість моделей NLP.

2.3.3 Інтеграція доменних знань

Інтеграція доменних знань у моделі обробки природної мови (NLP) є важливою складовою для підвищення точності та надійності моделей у спеціалізованих галузях. Використання спеціалізованих знань дозволяє моделям ефективніше обробляти специфічну термінологію та контексти, що є критичним для успішного вирішення багатьох задач у конкретних галузях.

Одним із найпростіших методів інтеграції доменних знань є використання спеціалізованих корпусів текстів для навчання моделей NLP. Наприклад, у галузі медицини використання корпусів, таких як Medline або PubMed, дозволяє моделям навчатися на текстах, насичених медичною термінологією та контекстами, що значно підвищує точність моделей при обробці медичних документів [49].

Додатковий спосіб інтеграції спеціалізованих знань включає використання доменних словників та тезаурусів. Наприклад, у юридичній галузі використання юридичних словників та тезаурусів допомагає моделям точніше розпізнавати та інтерпретувати юридичні терміни та вирази. Це

підвищує точність моделей у задачах, таких як класифікація юридичних документів або розпізнавання іменованих сутностей.

Інтеграція структурованих баз знань, таких як Wikidata або DBpedia, дозволяє моделям NLP використовувати багаті знання про об'єкти та їх взаємозв'язки. Це особливо корисно для задач, які вимагають глибокого розуміння контексту та логічних взаємозв'язків. Наприклад, у галузі біомедицини використання структурованих баз знань допомагає моделям краще розуміти складні наукові концепти та їх взаємозв'язки [52].

Онтології є потужним інструментом для структурованого представлення знань у певній галузі. Інтеграція онтологій у моделі NLP дозволяє забезпечити точність і логічність висновків, що особливо важливо для спеціалізованих задач. Наприклад, у галузі фінансів використання онтологій фінансових термінів та концептів дозволяє моделям точніше аналізувати фінансові документи та здійснювати автоматичне анотування текстів [53].

Інтеграція знань із експертних систем дозволяє моделям NLP використовувати правила та логіку, розроблену експертами у певній галузі. Це підвищує точність моделей у задачах, що вимагають глибокого розуміння та спеціалізованих знань. Наприклад, у галузі медицини експертні системи можуть допомагати моделям NLP у діагностиці захворювань або в аналізі медичних звітів.

У медицині інтеграція доменних знань значно покращує точність аналізу текстів. Використання медичних баз знань та онтологій, таких як SNOMED CT або MeSH, дозволяє моделям краще розпізнавати медичні терміни та їх взаємозв'язки, підвищуючи точність розпізнавання іменованих сутностей та аналізу текстів.

У юридичній галузі використання юридичних онтологій та словників дозволяє моделям NLP точніше розпізнавати та інтерпретувати юридичні терміни, що покращує точність класифікації юридичних документів та

розпізнавання іменованих сутностей. Це особливо важливо для систем автоматичного аналізу юридичних текстів.

Інтеграція доменних знань у моделі машинного навчання є критично важливою для підвищення точності та надійності моделей у спеціалізованих галузях. Використання спеціалізованих корпусів текстів, словників, онтологій та структурованих баз знань дозволяє моделям NLP ефективніше використовувати ці знання для розв'язання різних задач, що забезпечує більш точні та надійні результати.

2.4 Впровадження та оцінка моделей Knowledge-Augmented NLP

Оцінка ефективності моделей Knowledge-Augmented NLP (KA-NLP) є критично важливою для визначення їх продуктивності та надійності у різних застосуваннях. Впровадження та оцінка моделей KA-NLP вимагає використання різноманітних методів і метрик, які дозволяють повністю зрозуміти, наскільки добре модель виконує свої задачі. У цьому підрозділі буде розглянуто основні методи та метрики, які використовуються для оцінки ефективності моделей KA-NLP.

Крос-валідація є стандартним методом оцінки моделей машинного навчання, включаючи KA-NLP. Вона полягає у розділенні даних на кілька частин (зазвичай 5 або 10), після чого модель тренується на кількох частинах даних і тестується на одній з них. Це дозволяє отримати стабільні та надійні оцінки продуктивності моделі, мінімізуючи вплив випадкових факторів.

Розділення даних на тренувальну та тестову вибірки є ще одним поширеним методом оцінки. Модель тренується на одній частині даних (зазвичай 70–80%) і тестується на іншій частині (20–30%). Це дозволяє оцінити, як модель буде працювати на нових, невідомих даних, що важливо для визначення її узагальнюючої.

Точність є однією з найбільш поширених метрик оцінки моделей класифікації. Вона визначає відсоток правильно класифікованих прикладів

від загальної кількості прикладів. Точність важлива для загального розуміння продуктивності моделі, але може бути недостатньо інформативною у випадку незбалансованих

Для більш детальної оцінки моделей KA-NLP, особливо у випадках незбалансованих класів, використовуються метрики Precision (точність), Recall (повнота) та F1-Score. Precision визначає, який відсоток прикладів, передбачених як позитивні, є дійсно позитивними. Recall визначає, який відсоток дійсно позитивних прикладів був правильно ідентифікований моделлю. F1-Score є гармонійним середнім між Precision та Recall, що дає збалансовану оцінку моделі.

Для оцінки моделей автоматичного резюмування та генерації тексту часто використовується метрика ROUGE (Recall-Oriented Understudy for Gisting Evaluation). ROUGE порівнює автоматично згенеровані резюме або тексти з референтними текстами, оцінюючи схожість на рівні слів або фраз.

BLEU (Bilingual Evaluation Understudy) є стандартною метрикою для оцінки моделей машинного перекладу. Вона порівнює автоматично згенеровані переклади з людськими перекладами на основі збігу н-грам, оцінюючи якість перекладу.

Для оцінки ефективності моделей KA-NLP можуть використовуватися комбінації зазначених методів і метрик. Наприклад, для оцінки моделі класифікації тексту можна використовувати крос-валідацію разом з метриками Precision, Recall та F1-Score, щоб отримати детальнішу оцінку продуктивності моделі. Для моделей автоматичного резюмування або перекладу корисними будуть метрики ROUGE та BLEU, які дозволяють оцінити якість згенерованих текстів у порівнянні з людськими зразками.

Застосування цих методів і метрик забезпечує комплексну оцінку ефективності моделей KA-NLP, що дозволяє визначити їх сильні та слабкі сторони, а також покращити їх продуктивність для реальних застосувань.

2.5 Аналіз популярних фреймворків

2.5.1 TensorFlow

TensorFlow – це потужний фреймворк для машинного навчання, розроблений компанією Google. Він підтримує широкий спектр алгоритмів, включаючи нейронні мережі, і має вбудовані інструменти для роботи з NLP задачами через бібліотеку TensorFlow Text. TensorFlow також добре інтегрується з іншими бібліотеками для обробки природної мови, такими як NLTK і SpaCy.

Переваги:

− висока продуктивність та масштабованість;

− потужна підтримка нейронних мереж та глибокого навчання;

− широке співтовариство користувачів та обширна документація.

Недоліки:

− відносна складність у використанні для новачків;

− велика кількість налаштувань, що може ускладнити початкову реалізацію.

2.5.2 PyTorch

PyTorch – це ще один потужний фреймворк для машинного навчання, розроблений Facebook. Він відомий своєю гнучкістю та легкістю у використанні, що робить його популярним вибором серед дослідників та розробників. PyTorch підтримує широке використання нейронних мереж та глибокого навчання, а також має хорошу інтеграцію з бібліотеками для обробки природної мови [60].

Переваги:

− інтуїтивно зрозумілий та простий у використанні;

− потужна підтримка нейронних мереж та глибокого навчання;

− широка підтримка спільноти та обширна документація.

Недоліки:

− деякі обмеження у масштабованості порівняно з TensorFlow.

2.5.3 Natural Language Toolkit

Natural Language Toolkit (NLTK) – це популярна бібліотека для обробки природної мови на Python. Вона містить великий набір інструментів для текстової обробки, таких як токенізація, стемінг, лематизація, розпізнавання іменованих сутностей та інші. NLTK часто використовується для навчальних цілей та базових задач NLP.

Переваги:

− простий у використанні та добре задокументований;

− великий набір інструментів для обробки тексту.

Недоліки:

− обмежена підтримка глибокого навчання;

− менша продуктивність порівняно з TensorFlow та PyTorch.

2.5.4 SpaCy

SpaCy – це сучасна бібліотека для обробки природної мови на Python, оптимізована для продуктивності та масштабованості. Вона включає інструменти для токенізації, лематизації, розпізнавання іменованих сутностей та інших задач NLP. SpaCy також добре інтегрується з TensorFlow та PyTorch, що дозволяє використовувати її у комплексних NLP задачах [61].

Переваги:

− висока продуктивність та ефективність;

− простий у використанні API;

− хороша інтеграція з TensorFlow та PyTorch.

Недоліки:

− обмежена підтримка деяких функцій порівняно з NLTK.

2.5.5 Вибір фреймворку

З урахуванням необхідності інтеграції знань та вимог до подальшого написання програми, найкращим вибором для практичної реалізації задачі класифікації тексту є PyTorch у поєднанні зі SpaCy. PyTorch забезпечує гнучкість та простоту у використанні для розробки та тренування нейронних мереж, тоді як SpaCy надає ефективні інструменти для попередньої обробки тексту та інтеграції зовнішніх знань.

Гнучкість та простота використання: PyTorch забезпечує інтуїтивно зрозумілий інтерфейс для розробки моделей глибокого навчання. PyTorch має потужну підтримку нейронних мереж, що дозволяє реалізовувати складні моделі для класифікації тексту. SpaCy надає ефективні інструменти для інтеграції зовнішніх знань, що покращує результати NLP моделей. PyTorch та SpaCy мають активне співтовариство та обширну документацію, що полегшує розробку та усунення проблем.

3 ПРАКТИЧНА РЕАЛІЗАЦІЯ ЗАСОБІВ ОБРОБКИ ПРИРОДНОЇ МОВИ ДОПОВНЕНИХ ЗНАННЯМИ

3.1 Програмна реалізація Knowledge-Augmented NLP методів на основі PyTorch у поєднанні зі SpaCy

Розробка та впровадження Knowledge-Augmented NLP методів потребує ефективних інструментів і бібліотек. У цьому підрозділі описується програмна реалізація на основі PyTorch у поєднанні зі SpaCy, що забезпечує інтеграцію зовнішніх знань для покращення результатів обробки природної мови.

Для реалізації проекту необхідно встановити такі бібліотеки: spacy, torch, torchvision, torchaudio та завантажити англійську модель SpaCy:

```
pip install spacy torch torchvision torchaudio
python -m spacy download en_core_web_sm
```

Імпорт необхідних бібліотек приведено в лістингу 3.1.

Лістинг 3.1 – Імпорт необхідних бібліотек приведено в лістингу

```
import torch
import torch.nn as nn
import torch.optim as optim
import spacy
from torch.utils.data import DataLoader, Dataset
from sklearn.model_selection import train_test_split
import numpy as np
```

Код підготовки даних наведено у лістингу 3.2.

Лістинг 3.2 – Підготовка даних

```
reviews = [
    {"text": "I love this product, it is amazing!", "label":
1},
```

Продовження лістингу 3.2

```
        {"text": "This is the worst thing I have ever bought.",
"label": 0},
        {"text": "Absolutely fantastic! Would buy again.",
"label": 1},
        {"text": "Not worth the money. Very disappointing.",
"label": 0}
    ]
    # Поділ даних на навчальну та тестову вибірки
    train_data, test_data = train_test_split(reviews,
test_size=0.2, random_state=42)
    # Завантаження моделі SpaCy
    nlp = spacy.load("en_core_web_sm")
    # Функція для перетворення тексту у вектори
    def tokenize(text):
        doc = nlp(text)
        return [token.vector for token in doc if not
token.is_stop and not token.is_punct]
    # Клас Dataset для роботи з PyTorch
    class ReviewDataset(Dataset):
        def __init__(self, data):
            self.data = data
        def __len__(self):
            return len(self.data)
        def __getitem__(self, idx):
            item = self.data[idx]
            text, label = item["text"], item["label"]
            tokens = tokenize(text)
            tokens = torch.tensor(tokens)
            return tokens, label
    # Створення DataLoader для навчальної та тестової вибірки
    train_dataset = ReviewDataset(train_data)
    test_dataset = ReviewDataset(test_data)

    train_loader = DataLoader(train_dataset, batch_size=2,
shuffle=True)
    test_loader = DataLoader(test_dataset, batch_size=2,
shuffle=False)
```

Код для створення моделі наведено у лістингу 3.3.

Лістинг 3.3 – Створення моделі

```
class TextClassifier(nn.Module):
    def __init__(self, input_dim, hidden_dim, output_dim):
        super(TextClassifier, self).__init__()
        self.fc1 = nn.Linear(input_dim, hidden_dim)
        self.relu = nn.ReLU()
        self.fc2 = nn.Linear(hidden_dim, output_dim)
        self.softmax = nn.Softmax(dim=1)

    def forward(self, x):
        x = self.fc1(x)
        x = self.relu(x)
        x = self.fc2(x)
        x = self.softmax(x)
        return x

# Параметри моделі
input_dim = 96  # Розмір векторного представлення SpaCy
hidden_dim = 50
output_dim = 2

model = TextClassifier(input_dim, hidden_dim, output_dim)
```

Код навчання моделі наведено у лістингу 3.4.

Лістинг 3.4 – Навчання моделі

```
criterion = nn.CrossEntropyLoss()
optimizer = optim.Adam(model.parameters(), lr=0.001)
num_epochs = 10
for epoch in range(num_epochs):
    for inputs, labels in train_loader:
```

Продовження лістингу 3.4

```
            # Перетворення вектори в середнє значення по
токенах
            inputs = torch.stack([torch.mean(input, dim=0) for
input in inputs])
            outputs = model(inputs)
            loss = criterion(outputs, torch.tensor(labels,
dtype=torch.long))
            optimizer.zero_grad()
            loss.backward()
            optimizer.step()
        print(f'Epoch [{epoch+1}/{num_epochs}], Loss:
{loss.item():.4f}')
```

Код для оцінки моделі наведено у лістингу 3.5.

Лістинг 3.5 – Оцінка моделі

```
model.eval()
correct = 0
total = 0
with torch.no_grad():
    for inputs, labels in test_loader:
        inputs = torch.stack([torch.mean(input, dim=0) for
input in inputs])
        outputs = model(inputs)
        _, predicted = torch.max(outputs.data, 1)
        total += labels.size(0)
        correct += (predicted == labels).sum().item()

print(f'Accuracy: {100 * correct / total}%')
```

Повний текст програмної реалізації наведено у додатку А.

3.2 Практичне використання класифікації тексту для аналізу настроїв у соціальних мережах

Аналіз настроїв у соціальних мережах є однією з ключових задач обробки природної мови (NLP), яка дозволяє визначати емоційний тон текстових повідомлень, таких як твіти, коментарі та пости. Використання Knowledge-Augmented NLP методів для аналізу настроїв забезпечує покращення точності та надійності результатів за рахунок інтеграції зовнішніх знань та семантичних контекстів.

Для навчання моделі створимо CSV-файл з даними з соціальних мереж для навчання моделі. Файл містить записи із текстами та відповідними мітками (позитивні – 1, негативні – 0).

Перші 10 строк файлу наведено на рисунку 3.1.

```
"I absolutely love this new phone! It's amazing.",1
"This is the worst service I have ever experienced.",0
"The movie was fantastic and very entertaining.",1
"I hate this weather. It's so depressing.",0
"Great job on the presentation! Very well done.",1
"This food is terrible, I will never eat here again.",0
"I had a wonderful time with my family at the park.",1
"The product broke after one use. Waste of money.",0
"Absolutely delighted with the customer support.",1
"The experience was awful and I regret coming here.",0
```

Рисунок 3.1 – Перші 10 строк файлу

Повний текст файлу наведено у додатку Б.

Приклад 1: Позитивний відгук.

Відгук: «I absolutely love this new phone! It's amazing.».

Аналіз:

− загальний тон: Позитивний;

− ключові слова: «love», «amazing»;

− коментар: цей відгук використовує сильні позитивні слова, такі як «love» та «amazing», щоб виразити високу задоволеність продуктом. Емоційна мова вказує на дуже позитивний досвід користувача.

Приклад 2: Негативний відгук.

Відгук: «This is the worst service I have ever experienced.».

Аналіз:

− загальний тон: Негативний;

− ключові слова: «worst», «ever experienced»;

− коментар: відгук містить сильні негативні вислови, такі як «worst» та «ever experienced», що підкреслює крайнє незадоволення обслуговуванням. Це свідчить про дуже негативний досвід.

3.3 Практичне використання класифікації тексту для фільтрації спаму

Фільтрація спаму є однією з критичних задач обробки природної мови (NLP), що полягає у виявленні та блокуванні небажаних або шкідливих повідомлень. Використання Knowledge-Augmented NLP методів для фільтрації спаму дозволяє покращити точність і надійність виявлення спамових повідомлень шляхом інтеграції зовнішніх знань та семантичного аналізу.

Для навчання моделі створимо CSV-файл з записами для навчання моделі фільтрації спаму. Файл містить текстові повідомлення та відповідні мітки (1 – спам, 0 – не спам). Перші 10 строк файлу наведено на рисунку 3.2.

```
    "Congratulations! You've won a $1,000 Walmart gift card.
Click here to claim your prize.",1
    "Hey, how are you? Just wanted to check in.",0
    "Please confirm your subscription by clicking this link.",1
    "Can we schedule a meeting for tomorrow?",0
    "Your account has been temporarily suspended. Please update
your information.",1
    "Happy Birthday! Hope you have a great day!",0
    "Urgent! Your account has been compromised. Verify your
identity now.",1
    "Looking forward to our lunch date next week.",0
    "Get cheap meds online without a prescription.",1
    "Just checking to see if you received my last email.",0
```

Рисунок 3.2 – Перші 10 строк файлу

Повний текст файлу наведено у додатку В.

Приклад 1: Спамове повідомлення.

Текст: «Congratulations! You've won a $1,000 Walmart gift card. Click here to claim your prize.».

Аналіз:

− загальний тон: Спам;

− ключові слова: «Congratulations», «won», «$1,000», «Click here»;

− коментар: це повідомлення є класичним прикладом спаму. Використання фраз «Congratulations» та «You've won» вказує на спробу заманити користувача. Крім того, обіцянка великої грошової винагороди і посилання «Click here» є типовими ознаками спаму.

Приклад 2: Неспамове повідомлення.

Текст: «Hey, how are you? Just wanted to check in.».

Аналіз:

− загальний тон: Не спам;

− ключові слова: «Hey», «how are you», «check in»;

− коментар: це повідомлення має дружній та невимушений тон, характерний для особистої кореспонденції. Відсутність обіцянок виграшу або посилань вказує на те, що це повідомлення не є спамом.

ВИСНОВКИ

Інтеграція знань у методи обробки природної мови відкриває значні можливості для покращення точності та ефективності систем NLP. В роботі розглянуто різні підходи та технології, від традиційних статистичних методів до сучасних глибоких нейронних мереж, що використовують графи знань і онтології для збагачення мовних моделей.

Огляд літератури та аналіз сучасних практик показали, що знання не лише сприяють підвищенню розуміння мови машинами, але й значно збільшують їх здатність до розуміння контексту та виконання складних завдань, таких як машинний переклад, автоматичне анотування та відповіді на запитання. Основні складності, з якими стикається розробник систем NLP включають складність інтеграції і синхронізації різноманітних джерел знань, а також вимоги до обчислювальних ресурсів.

Майбутні дослідження в цій галузі мають великий потенціал для розвитку, особливо в аспектах автоматизації та збільшення масштабів застосування знань у NLP. Розвиток технологій, таких як вдосконалені алгоритми машинного навчання, краще розуміння природної мови та еволюція баз знань, можуть надати нові рішення для ефективного використання знань.

Продемостровано, що хоча робота з інтеграцією знань в системи NLP несе в собі певні складності, переваги, які вона надає, роблять цей напрям досліджень вкрай важливим для подальшого прогресу в області штучного інтелекту. Зростаюча інтеграція і використання знань у NLP не тільки покращують існуючі системи, але й сприяють розробці нових інноваційних рішень, що в майбутньому зможуть удосконалити взаємодію між людиною та машиною.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1) Hutchins, W. John. "The Georgetown-IBM experiment demonstrated in January 1954." MT News International 8.11 (1997): 15-18.

2) Weizenbaum, Joseph. "ELIZA–a computer program for the study of natural language communication between man and machine." Communications of the ACM 9.1 (1966): 36-45.

3) Rabiner, Lawrence R. "A tutorial on hidden Markov models and selected applications in speech recognition." Proceedings of the IEEE 77.2 (1989): 257-286.

4) Miller, George A. "WordNet: a lexical database for English." Communications of the ACM 38.11 (1995): 39-41

5) McCallum, Andrew, and Kamal Nigam. "A comparison of event models for naive bayes text classification." AAAI-98 workshop on learning for text categorization. Vol. 752. No. 1. 1998.

6) Joachims, Thorsten. "Text categorization with Support Vector Machines: Learning with many relevant features." European conference on machine learning. Springer, Berlin, Heidelberg, 1998.

7) Hochreiter, Sepp, and Jürgen Schmidhuber. "Long short-term memory." Neural computation 9.8 (1997): 1735-1780.

8) Vaswani, Ashish, et al. "Attention is all you need." Advances in neural information processing systems 30 (2017).

9) Devlin, Jacob, et al. "BERT: Pre-training of deep bidirectional transformers for language understanding." arXiv preprint arXiv:1810.04805 (2018).

10) Radford, Alec, et al. "Language models are unsupervised multitask learners." OpenAI Blog 1.8 (2019): 9.

11) Brown, Tom B., et al. "Language models are few-shot learners." Advances in neural information processing systems 33 (2020): 1877-1901.

12) Lewis, David D. "Naive (Bayes) at forty: The independence assumption in information retrieval." European conference on machine learning. Springer, Berlin, Heidelberg, 1998.

13) Rennie, Jason DM, et al. "Tackling the poor assumptions of naive bayes text classifiers." ICML. Vol. 3. 2003.

14) Zhang, Harry. "The optimality of naive Bayes." AAAI. Vol. 1. No. 2. 2004.

15) Cortes, Corinna, and Vladimir Vapnik. "Support-vector networks." Machine learning 20.3 (1995): 273-297.

16) Scholkopf, Bernhard, Alexander J. Smola, and Klaus-Robert Muller. "Kernel principal component analysis." Artificial Neural Networks–ICANN'97. Springer, Berlin, Heidelberg, 1997.

17) Dumais, Susan T., et al. "Inductive learning algorithms and representations for text categorization." Proceedings of the seventh international conference on Information and knowledge management. 1998.

18) Hsu, Chih-Wei, Chih-Chung Chang, and Chih-Jen Lin. "A practical guide to support vector classification." BJU international 101.1 (2008): 1396-1400.

19) Bottou, Léon, and Chih-Jen Lin. "Support vector machine solvers." Large scale kernel machines 3.1 (2007): 301-320.

20) Ben-Hur, Asa, and Jason Weston. "A user's guide to support vector machines." Data mining techniques for the life sciences. Humana Press, 2010. 223-239.

21) Sutskever, Ilya, Oriol Vinyals, and Quoc V. Le. "Sequence to sequence learning with neural networks." Advances in neural information processing systems. 2014.

22) Graves, Alex, Abdel-rahman Mohamed, and Geoffrey Hinton. "Speech recognition with deep recurrent neural networks." 2013 IEEE international conference on acoustics, speech and signal processing. IEEE, 2013.

23) Tang, Duyu, Bing Qin, and Ting Liu. "Document modeling with gated recurrent neural network for sentiment classification." Proceedings of the 2015 conference on empirical methods in natural language processing. 2015.

24) Nallapati, Ramesh, et al. "Abstractive text summarization using sequence-to-sequence RNNs and beyond." arXiv preprint arXiv:1602.06023 (2016).

25) Lipton, Zachary C. "The mythos of model interpretability." Queue 16.3 (2018): 31-57.

26) Cho, Kyunghyun, et al. "Learning phrase representations using RNN encoder-decoder for statistical machine translation." arXiv preprint arXiv:1406.1078 (2014)

27) Chung, Junyoung, et al. "Empirical evaluation of gated recurrent neural networks on sequence modeling." arXiv preprint arXiv:1412.3555 (2014).

28) Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio. "Neural machine translation by jointly learning to align and translate." arXiv preprint arXiv:1409.0473 (2014).

29) Amodei, Dario, et al. "Deep speech 2: End-to-end speech recognition in english and mandarin." International conference on machine learning. PMLR, 2016.

30) Wu, Shijie, et al. "BERT-based ranking for biomedical entity normalization." Bioinformatics 36.4 (2020): 1234-1240.

31) Sun, Chi, Luyao Huang, and Xipeng Qiu. "Utilizing BERT for aspect-based sentiment analysis via constructing auxiliary sentence." arXiv preprint arXiv:1903.09588 (2019).

32) Alberti, Chris, et al. "BERT for coreference resolution: Baselines and analysis." arXiv preprint arXiv:1908.09091 (2019).

33) Tang, Raphael, and Jimmy Lin. "Training neural machine translation models with monolingual data." arXiv preprint arXiv:1804.08299 (2018).

34) Zellers, Rowan, et al. "Defending against neural fake news." Advances in Neural Information Processing Systems 32 (2019).

35) Roller, Stephen, et al. "Recipes for building an open-domain chatbot." arXiv preprint arXiv:2004.13637 (2020).

36) Tai, Kai Sheng, Richard Socher, and Christopher D. Manning. "Improved semantic representations from tree-structured long short-term memory networks." arXiv preprint arXiv:1503.00075 (2015).

37) Manning, Christopher D., Prabhakar Raghavan, and Hinrich Schütze. Introduction to information retrieval. Cambridge University Press, 2008.

38) Yogatama, Dani, et al. "Generative and discriminative text classification with recurrent neural networks." arXiv preprint arXiv:1703.01898 (2017).

39) Ruder, Sebastian, Matthew E. Peters, Swabha Swayamdipta, and Thomas Wolf. "Transfer learning in natural language processing." In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Tutorials, pp. 15-18. 2019.

40) Marcus, Gary, and Ernest Davis. "GPT-3, Bloviator: OpenAI's language generator has no idea what it's talking about." Technology Review (2020).

41) Lee, Ji Young, Franck Dernoncourt, and Peter Szolovits. "Mitigating bias in gender, race, and ethnicity in natural language processing: a survey of research and evaluation methods." Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP) (2020): 1012-1027.

42) Henderson, Matthew, et al. "POMDP-based statistical spoken dialog systems: A review." Proceedings of the IEEE 101.5 (2013): 1160-1179.].

43) Chen, Danqi, et al. "Reading Wikipedia to answer open-domain questions." Proceedings of the 55th annual meeting of the association for computational linguistics (volume 1: Long papers). 2017.

44) Hirst, Graeme. "Anaphora in natural language understanding: A survey." Springer-Verlag, 1981.

45) Pearl, Judea, and Dana Mackenzie. The Book of Why: The New Science of Cause and Effect. Basic Books, 2018.

46) Bender, Emily M., Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. "On the dangers of stochastic parrots: Can language models be too big?." Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency. 2021.

47) Saxton, David, et al. "Analysing mathematical reasoning abilities of neural models." arXiv preprint arXiv:1904.01557 (2019).

48) Navigli, Roberto. "Word sense disambiguation: A survey." ACM Computing Surveys (CSUR) 41.2 (2009): 1-69.

49) Wang, Yaqing, et al. "Clinical information extraction applications: a literature review." Journal of Biomedical Informatics 77 (2018): 34-49.

50) Kiperwasser, Eliyahu, and Yoav Goldberg. "Simple and accurate dependency parsing using bidirectional LSTM feature representations." Transactions of the Association for Computational Linguistics 4 (2016): 313-327.

51) Nadeau, David, and Satoshi Sekine. "A survey of named entity recognition and classification." Lingvisticae Investigationes 30.1 (2007): 3-26.

52) Vrandečić, Denny, and Markus Krötzsch. "Wikidata: a free collaborative knowledgebase." Communications of the ACM 57.10 (2014): 78-85.

53) Noy, Natalya F., et al. "Ontology development 101: A guide to creating your first ontology." Stanford knowledge systems laboratory technical report KSL-01-05 and Stanford medical informatics technical report SMI-2001-0880 32.1 (2001): 1-25.

54) Mikolov, Tomas, et al. "Efficient estimation of word representations in vector space." arXiv preprint arXiv:1301.3781 (2013).

55) Etzioni, Oren, et al. "Open information extraction: The second generation." IJCAI. Vol. 7. 2011.

56) Koehn, Philipp. Statistical machine translation. Cambridge University Press, 2009.

57) Paulheim, Heiko. "Knowledge graph refinement: A survey of approaches and evaluation methods." Semantic web 8.3 (2017): 489-508.

58) McCrae, John P., et al. "The open linguistics working group: Developing the linguistic linked open data cloud." Proceedings of the 10th International Conference on Language Resources and Evaluation (LREC 2016). 2016.

59) Hogan, Aidan, et al. "Knowledge graphs." arXiv preprint arXiv:2003.02320 (2020).

60) Paszke, Adam, et al. "PyTorch: An imperative style, high-performance deep learning library." Advances in Neural Information Processing Systems 32 (2019): 8026-8037.

61) Honnibal, Matthew, and Ines Montani. "spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing." To appear 7 (2017): 411-420.