ДЕРЖАВНИЙ УНІВЕРСИТЕТ ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

КАФЕДРА ІНФОРМАЦІЙНИХ СИСТЕМ ТА ТЕХНОЛОГІЙ

КВАЛІФІКАЦІЙНА РОБОТА

на тему: «ТЕХНОЛОГІЯ NLP ДЛЯ ВИКОРИСТАННЯ В ІНФОРМАЦІЙНИХ СИСТЕМАХ»

на здобуття освітнього ступеня бакалавра
зі спеціальності 126 Інформаційні системи та технології
освітньо-професійної програми Інформаційні системи та технології

Кваліфікаційна робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело

_____ *Дмитро КУЧЕРЯВИЙ*

Виконав:
здобувач вищої освіти
група ІСД-42
Дмитро КУЧЕРЯВИЙ

Керівник:
науковий ступінь, вчене звання
Оксана ТКАЛЕНКО
к.т.н., доцент

Рецензент:
науковий ступінь, вчене звання
_____
Ім’я, ПРІЗВИЩЕ

Київ – 2025

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

Навчально-науковий інститут Інформаційних технологій

Кафедра Інформаційних систем та технологій

Ступінь вищої освіти Бакалавр

Спеціальність Інформаційні системи та технології

Освітньо-професійна програма Інформаційні системи та технології

ЗАТВЕРДЖУЮ
Завідувач кафедрою ІСТ

_____Каміла СТОРЧАК
«____» _____20___ р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ СТУДЕНТУ
Кучерявому Дмитру Олексійовичу
(прізвище, ім'я, по батькові здобувача)

1.Тема кваліфікаційної роботи: «Технології NLP для використання в інформаційних системах»

керівник кваліфікаційної роботи Оксана ТКАЛЕНКО, к.т.н., доцент
(ім'я, ПРІЗВИЩЕ, науковий ступінь, вчене звання)

затверджені наказом вищого навчального закладу від «24» лютого 2025 року № 56.

2. Строк подання кваліфікаційної роботи: 30 травня 2025 року.

3. Вихідні дані до кваліфікаційної роботи: навчальний корпус слів;
матриця ознак;
методи векторизації текстів BoW, TF-IDF, Word2Vec;
типи ознак екземплярів датасетів;
науково-технічна література з питань, пов'язаних з AI, ML, NLP.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
1. Дослідження процесів обробки природної мови та аудіо.

2. Дослідження технологій та методів обробки природної мови.
3. Розробка рекомендацій для організації навчання та досліджень в області NLP.

5. Перелік ілюстративного матеріалу: *презентація*
   1. Задачі, пов'язані з текстом на природній мові.
   2. Принцип обробки аудіо.
   3. Етапи вирішення NLP-завдань.
   4. Метод векторизації тексту Bag of Words.
   5. Метод векторизації тексту TF-IDF.
   6. Методи векторизації тексту Word2Vec / GloVe.
   7. Визначення навиків спеціаліста з NLP.
   8. Python як інструмент для NLP.
   9. Платформа Anaconda.
   10. Рекомендації по навчанню та дослідженням в області NLP.

6. Дата видачі завдання: 24 лютого 2025 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Аналіз наявної науково-технічної літератури	24.02 – 01.03.25	
2	Дослідження методів машинного навчання	02.03 – 14.03.25	
3	Дослідження особливостей глибокого навчання	15.03 – 30.03.25	
4	Дослідження особливостей побудови нейронних мереж	31.03 – 07.04.25	
5	Вивчення особливостей фреймворків TensorFlow, Keras, PyTorch, Theano	08.04 – 20.04.25	
6	Реалізація моделі ML для обробки даних	21.04 – 26.04.25	
7	Оформлення роботи: вступ, висновки, реферат	27.04 – 18.05.25	
8	Розробка демонстраційних матеріалів	19.05 – 30.05.25	

Здобувач вищої освіти _____ Дмитро КУЧЕРЯВИЙ
(підпис) *(Ім'я, ПРІЗВИЩЕ)*

Керівник роботи
кваліфікаційної роботи _____ Оксана ТКАЛЕНКО
(підпис) *(Ім'я, ПРІЗВИЩЕ)*

РЕФЕРАТ

Текстова частина кваліфікаційної роботи на здобуття освітнього ступеня бакалавра: 64 стор., 37 рис., 13 табл., 27 джерел.

Мета роботи – дослідити підходи до обробки природної мови (NLP) в контексті сучасних інформаційних систем.

Об'єкт дослідження – процеси аналізу та обробки текстової інформації в інформаційних системах.

Предмет дослідження – методи та алгоритми обробки природної мови, зокрема моделі машинного навчання та підходи до векторизації тексту (Bag of Words, TF-IDF, Word2Vec), які застосовуються для автоматизованої інтерпретації текстових даних в інформаційних системах.

Короткий зміст роботи: Досліджені методи та технології обробки даних в інформаційних системах, визначені рекомендації для вивчення технології NLP та пов’язаного з нею машинного навчання, досліджені математичні інструменти для аналізу даних. Досліджені особливості технології NLP, технології та методи обробки природної мови, мультимодальні задачі, які вирішуються за допомогою NLP-технології. Визначені підходи до векторизації тексту (Bag of Words, TF-IDF, Word2Vec), які застосовуються для автоматизованої інтерпретації текстових даних в інформаційних системах. Розроблені рекомендації для організації навчання та досліджень в області NLP.

КЛЮЧОВІ СЛОВА: ДАНІ, МАШИННЕ НАВЧАННЯ, НЕЙРОННА МЕРЕЖА, МОВА ПРОГРАМУВАННЯ PYTHON, ВЕКТОРИЗАЦІЯ, ТЕХНОЛОГІЯ, АЛГОРИТМ, МЕТОД, КОЛІРНА МОДЕЛЬ, ІНТЕРАКТИВНЕ СЕРЕДОВИЩЕ.

ЗМІСТ

ВСТУП

Актуальність теми: У сучасному інформаційному суспільстві щодня генерується велика кількість текстової інформації: електронні листи, повідомлення, новини, запити у пошукових системах, соціальні медіа, документи тощо. Ефективна обробка цієї інформації є критично важливою для прийняття рішень, автоматизації бізнес-процесів і підвищення якості обслуговування користувачів. Саме тому, технологія обробки природної мови (Natural Language Processing, NLP) стає одним із ключових напрямів розвитку інформаційних систем. NLP дозволяє машинам розуміти, аналізувати, інтерпретувати та генерувати людську мову. Це відкриває можливості для створення інтелектуальних програм, здатних автоматично:

- класифікувати тексти;
- визначати емоційне забарвлення повідомлень;
- витягати ключову інформацію;
- вести діалог із користувачем;
- здійснювати машинний переклад;
- розпізнавати запити в природній мові.

Інформаційні системи, що інтегрують NLP-модулі, значно розширюють свою функціональність і стають набагато зручнішими для кінцевого користувача. Зокрема, NLP-технології успішно використовуються в системах підтримки клієнтів (чат-боти), аналітики даних, контент-модерації, електронного навчання, правових експертних системах, медичних ІС тощо.

Крім того, стрімкий розвиток відкритих бібліотек (таких як spaCy, Hugging Face Transformers), платформ для навчання моделей (Google Colab, Jupyter Notebook) та доступ до попередньо навчених мовних моделей (BERT, GPT) робить цю технологію доступною для широкого кола розробників, студентів і науковців. Це відкриває нові горизонти для практичної реалізації NLP-рішень навіть у рамках дипломного проєкту.

Отже, вивчення та впровадження NLP у контексті інформаційних систем є високорелевантним та перспективним напрямом, що поєднує як наукову новизну, так і практичну значущість. Дослідження у цій сфері дозволяють поглибити розуміння сучасних мовних технологій, отримати цінні практичні навички та зробити внесок у розвиток інтелектуальних інформаційних систем. Це свідчить про актуальність обраної теми бакалаврської кваліфікаційної роботи та подальших досліджень.

Мета роботи - дослідити підходи до обробки природної мови (NLP) в контексті сучасних інформаційних систем.

Для досягнення поставленої мети необхідно виконати наступні *завдання*:

- дослідити особливості технології NLP;
- дослідити технології та методи обробки природної мови, мультимодальні задачі;
- визначити інструменти для обробки та аналізу даних;
- розробити рекомендації для організації навчання та досліджень в області NLP.

Об'єкт дослідження - процеси аналізу та обробки текстової інформації в інформаційних системах.

Предметом дослідження є методи та алгоритми обробки природної мови, зокрема моделі машинного навчання та підходи до векторизації тексту (Bag of Words, TF-IDF, Word2Vec), які застосовуються для автоматизованої інтерпретації текстових даних в інформаційних системах.

Наукова новизна отриманих результатів: підготовлено рекомендації щодо вибору інструментів та технологій (мови програмування, бібліотек, середовищ), що доцільні для впровадження NLP-функціоналу в інформаційні системи початкового та середнього рівня складності.

Апробація результатів бакалаврської роботи:

Кучерявий Д. О. «Алгоритми сортування в Python». Тези доповіді на VI Міжнародній науково-технічній конференції «Сучасний стан та перспективи розвитку IoT». – Київ, 15 квітня 2025 року.

1 ДОСЛІДЖЕННЯ ПРОЦЕСІВ ОБРОБКИ ПРИРОДНОЇ МОВИ ТА АУДІО

1.1 Задачі, які пов'язані з обробкою природної мови

Починаючи досліджувати обробку природної мови та аудіо, було визначено, що спочатку потрібно дослідити задачі, які виникають у цій області. Потім потрібно розуміти як слова і тексти представляються у комп'ютері, щоб їх можна було подавати на вхід моделі машинного навчання. Для цього потрібно розглянути варіанти векторного представлення слів, потім розглянути, що таке ембедінги слів: чим вони корисні, як їх отримувати і як їх використовувати.

На першому етапі дослідження були визначені задачі, які виникають при обробці природньої мови та аудіо.

Обробка природньої мови (Natural Language Processing, NLP) включає в себе задачі, які пов'язані з текстами на природній мові. Це як задачі розуміння мови, так і задачі генерування текстів на природній мові. А деякі задачі включають обидва аспекти. Розуміння тексту – коли по вхідному тексту потрібно щось зрозуміти і видати відповідь. На сьогоднішній день актуальними є задачі генерації тексту на природній мові. Бувають задачі, які включають обидва аспекти: коли по вхідному тексту потрібно щось зрозуміти і на основі цього тексту згенерувати новий текст.

Досліджуючи технологію NLP, можна визначити класичні задачі, які виникають в цій області. До таких задач відносяться класифікація тексту (наприклад, по авторству, жанру) та аналіз настрою (sentiment analysis) (рис. 1.1).

Першою задачею, яка є важливою в області Natural Language Processing є класифікація як і для будь-яких інших типів даних. Можна класифікувати тексти по авторству, за жанром та по багатьом іншим класам. Серед класифікації тексту доцільно виділити окрему задачу, яка застосовується найчастіше. Це аналіз настрою або аналіз тональності (sentiment analysis) – коли по вхідному тексту потрібно визначити настрій даного тексту – позитивний, нейтральний, негативний. Розглянемо приклади практичного застосування цієї задачі.

Наприклад, у нас є додаток Web Story і ми хочемо оцінити, на скільки користувачам подобається або не подобається наш додаток. Ми можемо вивантажити автоматично всі коментарі, всі відгуки, які користувачі залишають і прогнати їх через модель sentiment analysis і вона видасть нам для кожного коментаря оцінку, на скільки цей коментар позитивний, нейтральний або негативний.

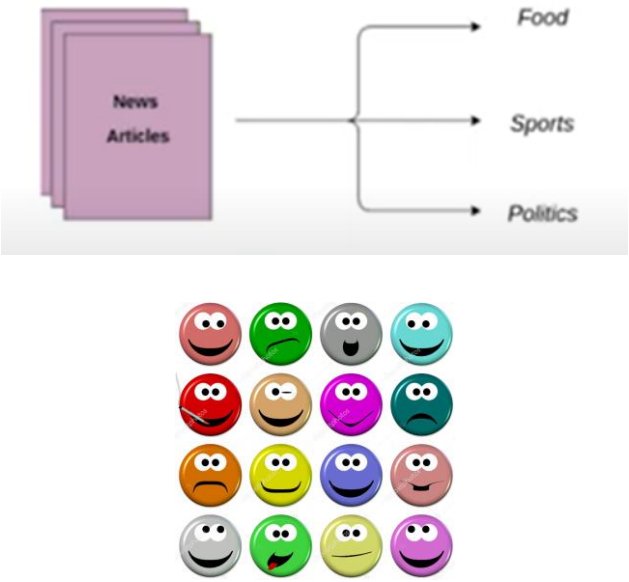

Рис. 1.1 Класичні задачі в області NLP

Інший приклад. Припустимо, що ми соціолог і досліджуємо вплив політичної події на настрій людей. Припустимо в якійсь країні якась людина була обрана президентом. Після цієї новини всі починають це обговорювати у соціальних мережах. Ми можемо вивантажити всі ці коментарі, прогнати через модель sentiment analysis і отримати приблизний настрій громадськості у зв’язку з цією подією.

Наступними задачами є генерація тексту і машинний переклад (рис. 1.2). Задача генерації тексту полягає в тому, щоб навчити модель машинного навчання (Machine Learning, ML) генерувати правдоподібний текст, тобто такий текст, який би не відрізнявся від того, що пише людина. Ми знайомі з такими моделями як ChatGPT або GPT4, випущені OpenAI. Бачимо приклад генерації ChatGPT. Це такі моделі, з якими можна спілкуватися, які можуть генерувати текст, який дійсно практично не відрізняється від людського.

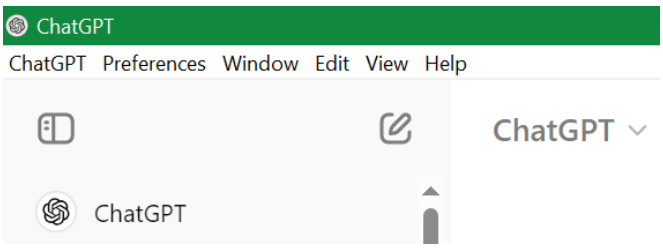

Рис. 1.2 Генерація тексту

Задача машинного перекладу полягає в тому, що потрібно побудувати модель, якій на вхід ми подаватимемо текст на одній мові, а вона нам генеруватиме переклад цього тексту на іншій мові. На перший погляд задачі генерації тексту і машинного перекладу не дуже схожі але насправді вони мають багато чого спільного. Задачу машинного перекладу можна розглядати як деяке розширення задачі генерації тексту. Насправді це логічно, тому що при машинному перекладі модель повинна генерувати текст, який є перекладом іншого тексту на іншій мові. Приклад використання машинного перекладу зрозумілий – Google-перекладач всередині використовує нейронну мережу для своїх задач.

У результаті проведених досліджень були визначені ще три задачі, які пов’язані з обробкою природної мови та аудіо. До них можна віднести: системи «питання-відповіді» (question answering); сумаризація тексту; діалогові системи (рис. 1.3).

Рис. 1.3 Chat-bot, Text summarization

Системи «питання-відповіді» – це коли нам потрібно побудувати модель машинного навчання, якій на вхід ми подаємо деяке питання, а вона генерує на нього відповідь. Ця задача також пов’язана з задачею генерації тексту, але відрізняється. Тут наша модель повинна генерувати відповідь на наше питання.

Задача сумаризація тексту полягає в тому, щоб подати на вхід моделі великий текст, модель повинна виділити з нього ключовий зміст і згенерувати текст поменше, який буде відображати сумаризовану ідею цього великого тексту, який був поданий на вхід. Приклад практичного застосування цієї задачі: Google помістив об’яву у Google Docs, де якщо у нас є великий текст у Google-документах, то він автоматично буде сумаризуватися у невеликий текст в лівому верхньому куті. Будуть виведені ключові аспекти цього тексту і якщо нова людина зайде на цю сторінку, він буде знати про що цей великий документ, на який він натрапив.

Діалогові системи – це задача побудови чат-ботів. Ця задача також пов’язана з генерацією тексту, але ідейно відрізняється у тому плані, що моделі потрібно не просто генерувати текст, а й вміти спілкуватися з людиною наче це інша людина.

Всі ці задачі схожі. Вони включають як розуміння тексту, так і його генерацію. Такі великі моделі для генерації тексту як ChatGPT і GPT-4 в процесі свого навчання навчаються вирішувати й інші задачі: question answering, сумаризація тексту, діалогові системи. Всі ці задачі мають багато чого спільного, у багатьох з них можна одночасно і розуміти текст, і генерувати текст, просто відрізняються деякі аспекти цих задач.

Ще одна задачею NLP є розпізнавання іменованих об’єктів (Named Entity Recognition, NER) (рис. 1.4).

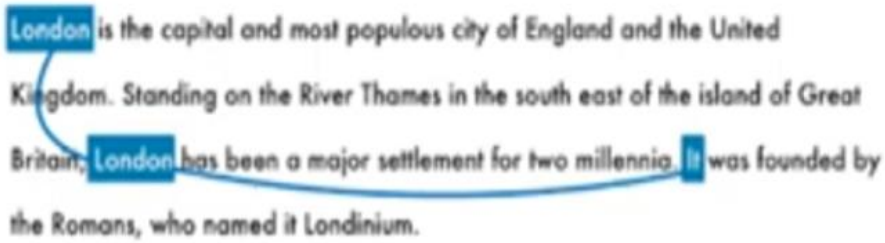

Рис. 1.4 Розпізнавання іменованих об’єктів

Ця задача не має особливого змісту сама по собі, але вона використовується як проміжна задача для вирішення інших більш складних задач. Припустимо, у нас є декілька сутностей, які ми хочемо отримувати з тексту. Нехай цими сутностями будуть люди, місця і організації. Тоді, задачею нашої моделі ML буде прийняти

на вхід текст, виділити з тексту всі слова, які відносяться до якої-небудь сутності з перерахованих (слова, які відносяться до людей, слова, які відносяться до організацій, слова, які відносяться до місць) і підписати кожне з цих слів – до чого це слово відноситься. Бачимо на скріншоті, що тут модель зрозуміла, що Aman Kharwal – це людина, India – це місце, Google – це організація [1-3].

Крім задач, які пов'язані виключно з текстами, можна вирішувати задачі як з картинками, так і з текстами. І взагалі, пов'язані з різними доменами, не тільки з картинками і текстами, а з чим небудь іншим (рис. 1.5).

Мультимодальність (CV + NLP):

- опис зображення/відео (image/video captioning);
- Visual question answering;
- розпізнавання тексту на зображенні (Optical Character Recognition, OCR).

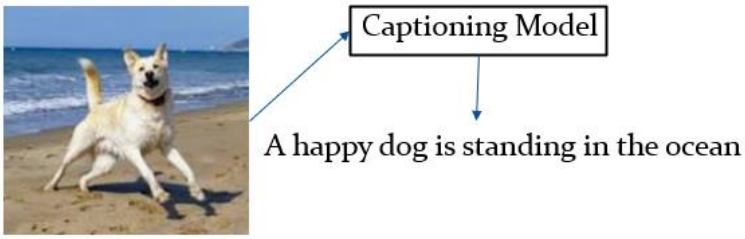

Рис. 1.5 Мультимодальність (CV + NLP)

Перша – це опис зображення або відео. На скріншоті бачимо приклад вирішення такої задачі моделлю. Моделі на вхід подається картинка, а задача моделі – згенерувати опис цієї картинки на природній мові.

Visual question answering – це той самий question answering тільки включає в себе ще візуальну компонету. Моделі на вхід подається не тільки питання на природній мові, але й картинка, з якою пов'язане це питання. Задача моделі – згенерувати відповідь на природній мові, яка була б відповіддю на це питання і зрозуміло, що в цьому випадку моделі потрібно як обробити картинку – візуальну компоненту, так і вхідне питання, тобто текст.

Третя задача – розпізнавання тексту на зображенні. Це дуже важлива задача у практичному застосуванні. Основана на тому, що моделі на вхід подається картинка, на якій написаний текст (це може бути картинка будівлі з вивісками з

текстами) або картинка меню ресторану. Задача моделі – розпізнати, де знаходиться текст і перевести текст в літери, тобто видати цей текст на природній мові користувачу. Ця задача вже може вирішуватися багатьма смартфонами, айфонами точно, тобто ми можемо просто взяти айфон, навести камеру на вивіску на вулиці, наприклад, камера автоматично розпізнає цей текст, виведе нам і зможе навіть перевести його на іншу мову. Це дуже зручно, наприклад, коли ми йдемо у ресторан у незнайомій країні, мову якої ми не знаємо, і нам потрібно перекласти меню.

1.2 Мультимодальні задачі. Обробка аудіо

Важливою мультимодальною задачею, яка пов’язана одночасно з розумінням тексту і картинок, є генерація зображення або відео по тексту. Останнім часом ця задача стала досить популярною, тому що з’явилися нові моделі, які можуть генерувати відмінні картинки за текстовим описом. На рис. 1.6 бачимо, що картинка дуже добре згенерувалася по текстовому опису.

Фотографія їжака, який сидить посередині озера. На ньому гавайська сорочка і солом’яна шляпа. Їжак читає книгу. На його фоні – листя.

Рис. 1.6 Генерація зображень/відео по тексту

Виконуючи дослідження задач для аудіо, було визначено, що з обробкою аудіо пов’язані наступні задачі (рис. 1.7):

- класифікація аудіо, аналіз настрою;
- Audio-to-text, text-to-audio;
- покращення якості аудіо (speech enhancement);
- розділення джерел звуку.

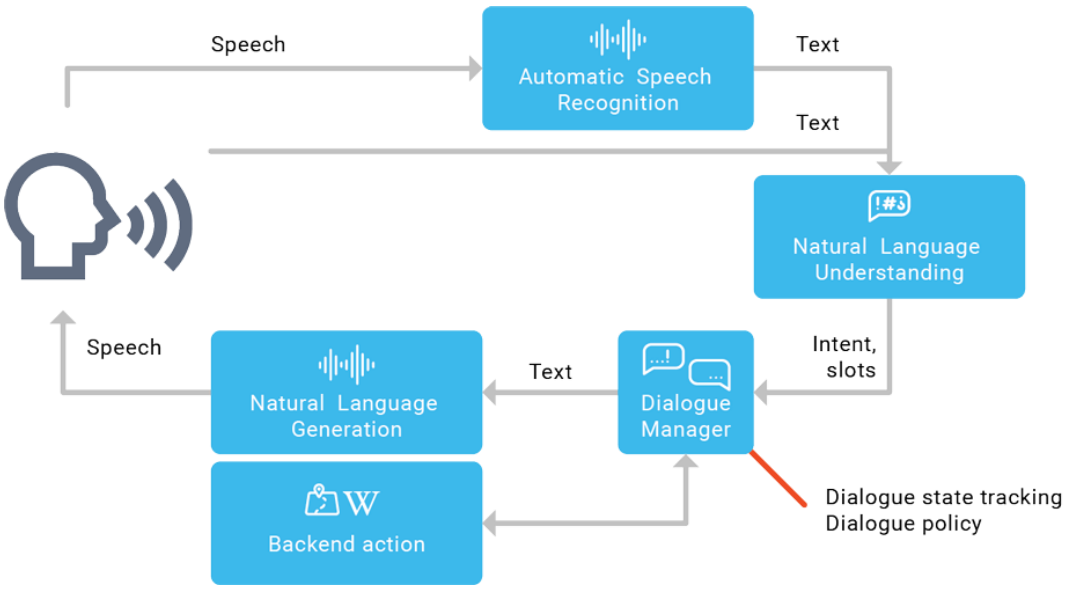

Рис. 1.7 Обробка аудіо

Покращення якості аудіо – це коли на вхід моделі ML подається аудіо, яке зашумлене, з якимись завадами, тобто не дуже гарної якості, задача нашої моделі – почистити це аудіо, тобто на виході видати той самий аудіофайл, тільки без шумів і похибок.

Розділення джерел звуку. На вхід подається аудіофайл, в якому змішані декілька джерел звуку, на виході модель повинна видати декілька файлів, в кожному з яких виділене окремо кожне джерело звуку.

OpenAI – це дослідницька компанія та лабораторія штучного інтелекту, яка заснована у грудні 2015 року. Мета OpenAI – розвивати та використовувати штучний інтелект (Artificial Intelligence, AI) на користь людства. В результаті досліджень були визначені основні напрямки діяльності OpenAI, а саме: дослідження; розробка технологій; етика та безпека; співпраця з іншими організаціями.

OpenAI займається фундаментальними та прикладними дослідженнями у сфері штучного інтелекту, розробляючи нові моделі та алгоритми. Компанія створює різноманітні технології на базі штучного інтелекту, включаючи генеративні моделі, мовні моделі (такі як GPT-3 і GPT-4), інструменти для машинного навчання та інші. OpenAI приділяє значну увагу етичним аспектам і питанням

безпеки штучного інтелекту, прагнучи забезпечити, щоб він був безпечним, прозорим і корисним для всіх.

Компанія активно співпрацює з іншими дослідницькими інститутами, урядовими органами та приватними компаніями, щоб прискорити прогрес у сфері штучного інтелекту та забезпечити його етичне використання. OpenAI відома своїми досягненнями у створенні потужних мовних моделей, таких як серія GPT (Generative Pre-trained Transformer), які можуть виконувати різноманітні завдання, включаючи генерацію тексту, переклад, аналіз тексту та багато іншого.

OpenAI виконує широкий спектр завдань, пов'язаних з розвитком та застосуванням штучного інтелекту. Основні з них включають: розробку спеціалізованих моделей для конкретних завдань, таких як комп'ютерний зір, розпізнавання мови, ігри; фундаментальні дослідження, спрямовані на розуміння основних принципів роботи штучного інтелекту; прикладні дослідження, що мають на меті створення нових алгоритмів та технологій для практичного застосування.

Використання моделей штучного інтелекту доцільне для автоматизації бізнес-процесів, покращення обслуговування клієнтів, аналізу великих обсягів даних та інших завдань. Інтеграція штучного інтелекту на теперішній час затребувана у всі галузі індустрії. Для спільного розвитку технологій штучного інтелекту доцільно організовувати співпрацю з академічними інститутами, урядовими організаціями та приватними компаніями, розробляти освітні програми, проводити семінари, воркшопи для поширення знань про штучний інтелект та його можливості.

Проаналізувавши процес розробки інструментів та платформ для штучного інтелекту, було визначено, що доцільним є створення доступних інструментів і платформ для розробників, які дозволять використовувати потужності штучного інтелекту у своїх проєктах. Також потрібно забезпечувати підтримку відкритого коду та надання доступу до своїх моделей через API. Адже саме ці завдання та можливості дозволять забезпечити безпечний та корисний розвиток штучного інтелекту для суспільства та всього людства.

1.3 Застосування технології NLP у поєднанні з колірними моделями для аналізу та генерації текстових описів візуальних даних

Технологія NLP у поєднанні з візуальними даними може аналізувати, описувати або навіть розпізнавати кольори через текст з використанням колірних моделей RGB, HSV, Cielab і такі застосування особливо корисні у дизайні, маркетингу, комп'ютерному зорі та мультимодальних моделях.

У задачах візуалізації текстових даних можуть використовуватися колірні моделі:

- Word Cloud (Хмара слів) - різні слова можуть мати різні кольори для кращого сприйняття;
- візуалізація кластерів слів (наприклад, у Word2Vec, FastText) - слова, що мають схожий зміст, можуть бути позначені різними кольорами;
- теплові карти (Heatmaps) - використовуються у аналізі уваги (Attention Mechanisms) у моделях Transformer, де кольори відображають рівень важливості слів.

У NLP можуть бути задачі, де текст містить описи кольорів:

- обробка відгуків і описів товарів - аналізують тексти, які містять згадки про кольори (наприклад, у моді, дизайні);
- генерація текстів із зображень (Image Captioning) - моделі можуть описувати кольори об'єктів на зображеннях;
- розпізнавання кольорів у тексті - конвертація назв кольорів у числові значення RGB або HSV.

У сучасному штучному інтелекті NLP часто поєднується з комп'ютерним зором (CV), де вже безпосередньо використовуються колірні моделі:

- мультимодальні моделі (CLIP, Flamingo, VisualGPT) - поєднують текстові дані з візуальними зображеннями;
- аналіз зображень та текстових описів - для автоматичної генерації текстових підписів до зображень;

- переклад кольорів у текст - моделі можуть пояснювати кольорові градієнти або відповідність кольорів у дизайні.

Виконаємо дослідження, яким чином графічний файл передає нам кольори і яким чином користувач сприймає колір. Клітини сітківки людського ока неоднорідні. Вони складаються з декількох груп клітин, чутливих до певних діапазонів видимого спектру. Одні відповідають за один колір, інші – за інший колір і насправді ми сприймаємо колір дуже дискретно. Це по перше. По друге, оскільки будова людського організму зокрема і нервової системи в особистості, а зір – це частина нервової системи – індивідуальні, то виявляється кожна людина і сам колір теж сприймаємо індивідуально.

У фізичному світі для представлення кольорів використовуються барвники. Комп'ютер оперувати фізичними барвниками не може, тому представляє кольори у вигляді певних чисел. Колірний простір – це опис визначеної числової системи, яку пристрій або програма використовують для представлення кольорів. Фактично це координатний простір, в якому чітко можна визначити колір та відрізнити один колір від іншого.

Кожна людина забарвлення бачить по своєму, але забарвлення сприймається і визначається однаково. Це називається соціалізація. Починаючи від нашого народження, нас навчають певним шаблонам, у тому числі і в плані сприйняття кольорів. Нам кажуть – зелений колір (листок на дереві - то зелений колір). А на світлофорі ось та верхня лампочка горить червоним кольором, а потім загориться жовтим, а потім зелений. І ми все це сприймаємо із самого дитинства і зрештою наші індивідуальні кольори, які ми з вами бачимо, ми їх починаємо називати так, як називають їх усі інші. Знову ж таки, не зважаючи на те, що всі ми кольори в природі бачимо по іншому. Результат нашого світосприйняття є наслідком деяких фізико-хімічних реакцій, які починають формуватися в сітківці нашого ока і передаються у вигляді електричних імпульсів на потиличну частину нашого головного мозку. Там у нас знаходяться центри сприйняття зору.

У комп'ютера не має фізико-хімічних реакцій на колір, але комп'ютеру якимось чином потрібно цей колір для нас людей передавати. Передавання

кольору в комп’ютері базується на цифрах. Тобто якщо наші сприйняття того чи іншого кольору – це є наслідком хімічної реакції в стовпчикових клітинах наших нашої сітківки, то в комп’ютері це просто шифр у вигляді цифр. А от що стосується комбінацій цього шифру – комбінації цього шифру можуть бути різні і підходи до того, що шифрувати теж можуть бути різні. Власне кажучи комбінації цього шифру і підходи до шифрування і визначають, яку колірну схему використовує комп’ютер.

Колірна модель – це математична модель, що описує представлення кольорів у вигляді кортежів чисел (зазвичай з трьох, рідше – чотирьох значень), так званих колірних компонент або колірних координат. Усі можливі значення кольорів, що задаються колірною моделлю, визначають колірний простір. Кортежі – це те, що ми вказуємо у круглих дужках через кому. От саме кольори ми передаємо у вигляді кортежів. Колірна модель задає відповідність між кольорами, як їх сприймає людина, та кольорами, що відображаються на цифрових пристроях (наприклад, на моніторах ноутбуків).

Якщо записати колір у наступним способом: (179, 12, 33) – це кортеж наших чисел. Колір записаний відповідно до правил колірної моделі RGB (рис. 1.8).

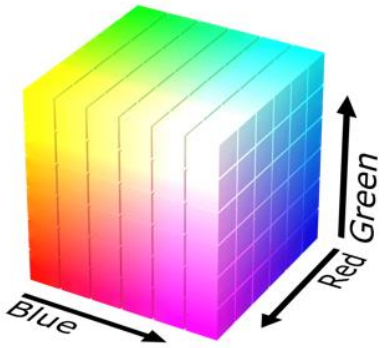

Рис. 1.8 Колірна модель RGB

Колірна модель RGB – це мікс трьох базових кольорів. Сама абревіатура назви RGB походить від цих кольорів. У результаті міксу трьох кольорів отримується певний відтінок. Цей мікс може бути в межах діапазону від 0 до 255 і таким чином по кожному із цих кольорів (червоному, зеленому і синьому) відбувається варіація в межах від 0 до 255, тобто 256 варіантів [5-7].

Порахуємо, скільки може бути варіантів з урахуванням того, що у нас три кольори: 256 * 256 * 256 = 16777216. В межах даної колірної схеми у нас можливі 16777216 унікальних наборів кольорів. Кожен із цих наборів кольорів змінюється в межах цього діапазону.

0 – 255
0 – 255 - в межах цих варіацій шифрується колір
0 – 255

Якщо взяти вихідний варіант - три нулі, колір буде в результаті змішування трьох нулів – отримаємо чорний колір. І навпаки, якщо взяти набір із цифр 255 255 255 - отримаємо колір білий. В межах між чисто чорним і чисто білим кольором знаходяться всі можливі відтінки із 16 777 216 варіацій кольору у форматі RGB. Представити графічно можна все це у вигляді куба, як на рис. 1.8.

Проведемо дослідження, яким чином все це працює. Для цього потрібно зайти у пошуковій системі в CSS RGB (рис. 1.9).

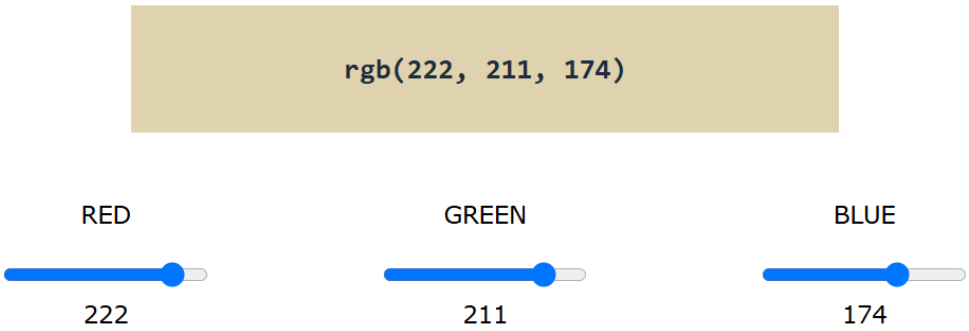

Рис. 1.9 CSS RGB

Це калькулятор кольорів у форматі RGB. Якщо показники спектру (або каналу) RED, GREEN, BLUE перевести в нуль - отримуємо чисто чорний колір. Коли змінювати канал RED – зображення починає червоніти. І в самому своєму найвищому варіанті, тобто при значенні кортежу (255, 0, 0) – отримуємо чисто червоний колір. Відповідно якщо ми поставимо значення нашого кортежу (0, 255, 0) – отримаємо яскраво-зелений колір. І якщо поставити значення кортежу (0, 0, 255) – отримаємо синій колір.

Всі інші кольори знаходяться в межах цих варіацій. Наприклад, комбінація (94, 60, 154) – дасть деякий варіант фіолетового кольору. Трохи нижче на сайті CSS RGB можна знайти більш сталі варіанти схеми RGB для інших кольорів. Чисто зелений колір можна отримати шляхом встановлення показників кортежу (0, 255, 0).

Всі графічні файли на сьогоднішній день представлені двома основними моделями представлення комп'ютерної графіки: векторна і растрова моделі. Наприклад, коли використовується смартфон для того, щоб щось сфотографувати, отримуємо растрову картинку тієї чи іншої колірної схеми. Коли на смартфон знімається відеоролик, насправді отримуємо не відеоролик, а набір певної кількості растрових картинок, які створюються у певній кількості за певний проміжок часу. Якщо у нормальному режимі проводити відеозйомку, то отримуємо картинку, яка за кожну секунду створює 30 растрових графічних файлів. У результаті швидкого перемотування цих файлів створюється ілюзія анімації.

Растрові файли мають піксельну структуру. Припустимо, що у нас це якась картинка. У растрових файлах верхній лівий кут вважається точкою відліку. Можемо провести аналогію з матрицею. У матриці верхній лівий кут – це теж точка відліку. І далі у нас іде сіточка. Ця сіточка складається з невеликих комірок, які називаються пікселями. Кожен піксель у растровому зображенні не простий – це не просто якась комірка, а це комірка, яка містить в собі певну інформацію. У кожного пікселя є унікальний ідентифікатор, тобто ID. У кожного пікселя є відповідна координати x, y (рис. 1.10).

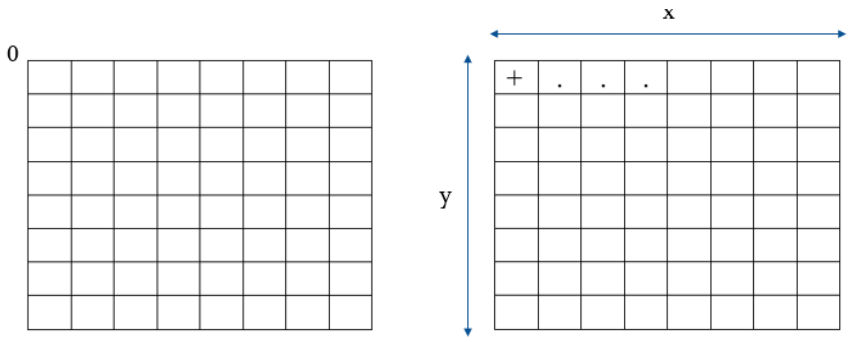

Рис. 1.10. Інформація про піксель

Піксель з координатами 0, 0 – це самий перший піксель, який знаходиться у верхньому лівому кутку. Всі інші відповідно мають похідні від нього координати: 0,1, 0,2, 0,3, 0,4 і т.д. Крім *x, y* – характеристик, кожен піксель ще має характеристики кольору відповідно до своєї колірної схеми. Якщо картинка у форматі RGB, то відповідно кожен піксель має параметр R, параметр G, параметр B – навпроти яких вказуються цифри у діапазоні від 0 до 255. Дослідимо наступне позначення:

ID
x - 1
y – 1
R – 0
G – 255
B - 0

Це означає, що піксель, який має ID з координатими x = 1 і y = 1 і має зелений колір. Таким чином, кожен піксель має унікальне забарвлення без відтінку, яке є однорідним без градієнтів.

1.4 Колірні моделі RGBA, HSV, Cie XYZ, Cie LAB

Є інші колірні схеми. Наприклад, колірна схема, яка складається не із трьох елементів, а з чотирьох. Один із варіантів такої схеми називається RGBA. Окрім трьох інтових значень основних кольорів (RED, GREEN, BLUE), є ще флотовий канал ALPHA. Якщо RED, GREEN, BLUE визначають кольори, то ALPHA визначає прозорість або інтенсивність кольору. Якщо ALPHA = 1, то це означає, що колір стовідсотково не прозорий і при показниках RGB (0, 255, 0) отримаємо той самий зелений колір, який ми отримали в колірній схемі RGB без ALPHA. ALPHA змінюється з кроком на 0.1 і коли ми змінюємо показник ALPHA, у нас колірна схема залишається та сама, але змінюється відтінок. При ALPHA = 0.2 і при показниках, які відповідають чисто зеленому кольору (0, 255, 0) маємо от відтінок зеленого кольору. ALPHA дає додаткові відтінки. Відповідно, якщо ми

отримаємо відтінки в десяти варіаціях, тобто 16 777 216 * 10 = 167 772 160 кольорів, тобто приріст у 10 разів (рис. 1.11).

Рис. 1.11 CSS RGBA

ALPHA, даючи більшу різноманітність для кольорів, є більш привабливою у використанні з точки зору промислової комп'ютерної графіки, наприклад, тієї комп'ютерної графіки, де потрібно друкувати бігборди – великі графічні зображення там де плавність переходу окремих відтінків повинна бути максимальною для того, щоб ці бігборди привабливо виглядали.

Колірна схема RGB на сьогоднішній день е комп'ютерній графіці найпоширеніша. Ті фотографії, які ми отримуємо за допомогою смартфонів, мають саме цю колірну схему.

Є така колірна схема, як HSV (рис. 1.12).

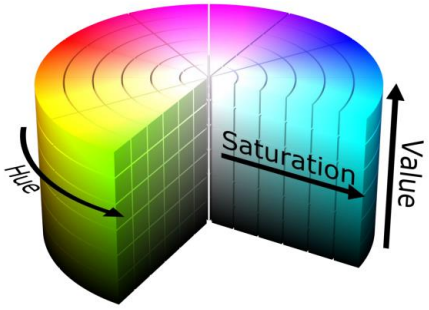

Рис. 1.12 Колірна модель HSV

Якщо RGB оперувало показниками трьох спектрів (червоного, зеленого, синього), то HSV оперується трьома характеристиками кольору (тоном, насиченістю, яскравістю). Ці три характеристики визначають, якого кольору піксель ми будемо бачити в тому чи іншому місці на нашій картинці (рис. 1.13).

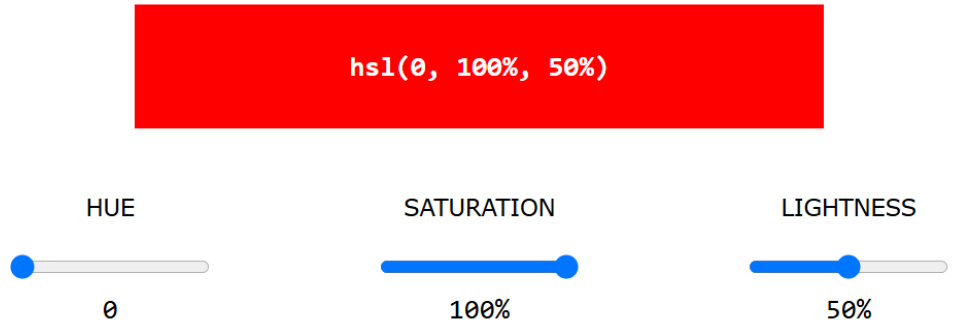

Рис. 1.13 CSS HSL

У даному конкретному випадку, коли ми маємо HUE = 0, SATURATION = 100 і LIGHTNESS = 50, отримаємо зелений колір, який би отримали при даних RGB (250, 0, 0). Подивимось, як тут відбувається варіація. Це у нас також кортеж, який складається із трьох чисел. Але перше число – це ціле число і воно змінюється в межах від 0 до 359. Друге число дає відсотки, третє число також дає відсотки. В цілому може бути варіацій 360 * 100 * 100 = 3 600 000 кольорів. Менше, ніж у форматі RGB.

Перша характеристика HUE є самою головною, тому що вона відображає коло, кожен градус з якого має певний відтінок і за допомогою показника цього градусу ми цей відтінок вибираємо. Після відтінку ми можемо вибрати Saturation – він змінюється від центру до периферії, де в центрі він самий світлий, а на периферії він самий темний. І теж саме стосується Value – насиченості (або яскравості) – вгорі у нас колір самий яскравий, внизу у нас колір самий темний. Варіюючи цими трьома показниками, отримуємо колірну схему HSV (або HSL).

Cie XYZ – еталонна колірна модель, задана у строгому математичному сенсі організацією CIE (Міжнародна комісія з освітлення). Вона грунтується на функціях колірної відповідності, що корелюють із функціями спектральної

чутливості колбочок – рецепторів людського ока, що відповідають за сприйняття кольорів (рис. 1.14).

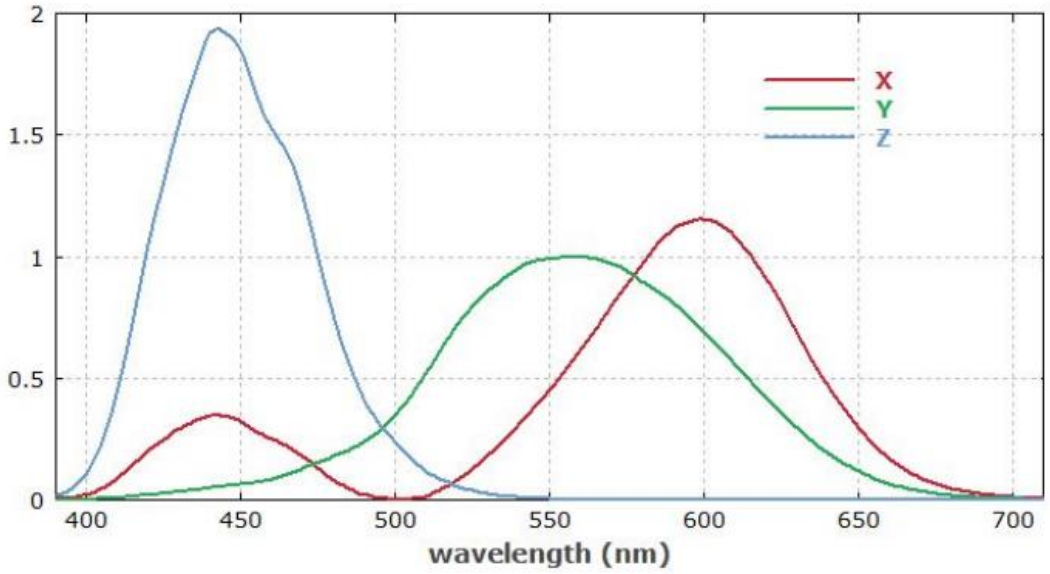

Рис. 1.14 Колірна модель Cie XYZ

Ця модель була винайдена для того, щоб стандартизувати кольори і відтінки світіння електричних лампочок. Схема Cie XYZ грунтується на функціях колірної відповідності, що корелюється з функціями спектральних чутливих колбочок, тобто рецепторів людського ока. Тобто цю схему розробляли у співпраці з лікарями-офтальмологами і лікарі-офтальмологи підказували, яким чином змінюється сприйняття того чи іншого діапазону кольору. Тут їх три і вони теж як в RGB відповідають червоному, зеленому і синьому. Відмінність полягає в протяжності цих спектрів. Якщо в RGB червоний, синій, зелений мають однакову протяжність, то як ми можемо побачити з цієї діаграми у схемі Cie XYZ вони мають протяжність не однакову. Дана схема вираховується по наступному принципу: беруться показники X, Y, Z – сума цих показників і діляться значення X, Y, Z відповідно:

$$\begin{cases} x = \dfrac{X}{X+Y+Z} \\ y = \dfrac{Y}{X+Y+Z} \\ z = \dfrac{Z}{X+Y+Z} \end{cases} \qquad (1.1)$$

Окрім стандартних колірних координат, також часто використовують поняття відносних колірних координат, що відповідають формулі (1.1). Тут $X+Y+Z=1$, а це означає, що для однозначного завдання відносних координат достатньо будь-якої пари значень, а відповідний колірний простір може бути представлений у вигляді двовимірного графіка. Сума всіх цих показників повинна відповідати одиниці. А пропорції X, Y, Z будуть визначати відтінок того чи іншого кольору. Представляється дана схема у вигляді ось такого зрізаного конусу (рис. 1.15).

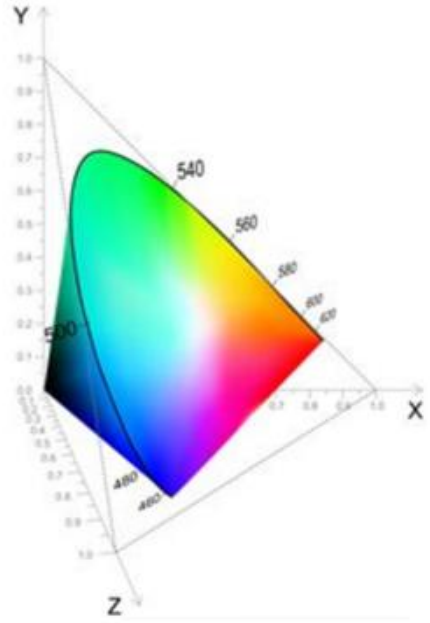

Рис. 1.15 Представлення кольорів у системі Cie XYZ

Є продовження цієї схеми, яка називається Cie LAB (рис. 1.16). Вона розроблена на основі схеми Cie XYZ.

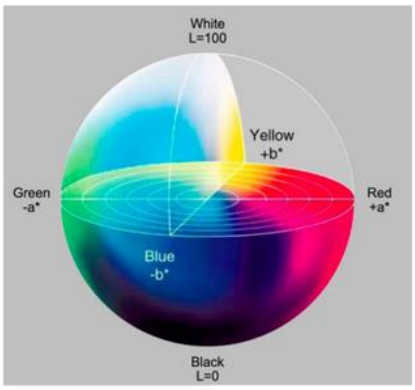

Рис. 1.16 Колірна модель Cie LAB

Модель CIE LAB – видозміна моделі Cie XYZ, в якій будь-яка зміна кольору буде більш лінійною з точки зору людського сприйняття. Фактично CIE LAB розроблена для того, щоб однакова зміна значень координат кольору у різних областях колірного простору спричиняла у людини однакове відчуття зміни кольору. Таким чином, математично коригується нелінійність сприйняття кольору людиною.

На практиці найчастіше мають справу з наступними колірними схемами: RGB, RGBA, HSV, CMYK. CMYK дуже часто використовується комп'ютерними графіками, які створюють графічні файли високої роздільної здатності високої якості. CMYK - це одна з основних колірних моделей, яку широко використовують у поліграфії та дизайні.

2 ДОСЛІДЖЕННЯ ТЕХНОЛОГІЙ ТА МЕТОДІВ ОБРОБКИ ПРИРОДНОЇ МОВИ

2.1 Вирішення завдань з NLP

Технологія NLP дозволяє автоматично аналізувати, розуміти, інтерпретувати й генерувати природну мову. Вона застосовується для вирішення широкого спектру завдань у бізнесі, науці, IT-сфері, освіті, медицині та інших галузях.

Дослідивши основні завдання, які вирішуються за допомогою NLP у сучасному світі, можна виділити з них наступні:

- класифікація тексту;
- інформаційний пошук і витягування інформації;
- машинний переклад;
- генерація тексту;
- питання–відповіді (Question Answering);
- аналіз настроїв;
- автоматичне резюмування;
- синтаксичний та семантичний аналіз;
- розпізнавання мовлення та синтез;
- діалогові системи (чат-боти).

Класифікація тексту передбачає розподіл текстів за категоріями. Наприклад, це може бути визначення тональності (позитивний/негативний відгук), тематична класифікація (спорт, політика, наука та інші), фільтрація спаму [10, 11].

Інформаційний пошук і витягування інформації забезпечує знаходження потрібної інформації у текстах. Це може бути витягування імен, організацій, дат (Named Entity Recognition), автоматичне заповнення таблиць з текстів, витяг ключових слів або фраз.

Машинний переклад – це переклад тексту з однієї мови на іншу. В якості інструментів можна використовувати: Google Translate, DeepL, OpenNMT, MarianMT.

Генерація тексту – це автоматичне створення зв’язного тексту. Приклади: генерація статей, звітів, листів; автоматичне резюмування (узагальнення текстів); написання коду або коментарів.

Ключову роль технологія NLP відіграє у відповідях на запитання на основі тексту або бази знань (питання–відповіді (Question Answering)). Приклади: чат-боти, віртуальні асистенти (Siri, Alexa, ChatGPT); медичні або технічні помічники.

Аналіз настроїв – це оцінка емоційного забарвлення тексту. Це можуть бути відгуки на товари або послуги, соціологічний аналіз дописів у соцмережах.

Автоматичне резюмування - скорочення великих текстів до основних тез. Прикладами можуть бути резюме статей, юридичних документів; новинні дайджести.

Синтаксичний та семантичний аналіз - розбір структури та змісту речень: аналіз граматики, частин мови; визначення ролей слів у реченні (розбір за залежностями).

Розпізнавання мовлення та синтез включає:

- Speech-to-text: розпізнавання усної мови;
- Text-to-speech: озвучення тексту.

Діалогові системи (чат-боти) забезпечують інтерактивну взаємодію з користувачем через текст або голос.

Ці завдання часто комбінуються у реальних системах - наприклад, віртуальний асистент може виконувати класифікацію, аналіз настроїв і відповідати на питання одночасно.

Вирішення завдань з NLP відбувається у кілька етапів:

- збір і підготовка даних;
- перетворення тексту у числове представлення;
- побудова та тренування моделі;
- оцінка моделі;
- тестування і вдосконалення;
- розгортання та використання.

На етапі *збору і підготовки даних* формуються вхідні дані, які будуть використані для побудови NLP-моделі. Відбувається:

- збір текстів з вебсайтів, документів, соцмереж, чатів;
- здійснюється очистка текстів (видалення HTML, знаків пунктуації, чисел, стоп-слів);
- токенізація - поділ тексту на окремі слова або речення;
- лематизація/стемінг - приведення слів до початкової форми;
- анотація даних, якщо задача з навчанням із вчителем (приклад: тональність тексту = позитивна/негативна).

Перетворення тексту у числове представлення. Комп'ютер не працює напряму з текстами - їх потрібно перетворити у вектори чисел. Основними методами, які виконують цю функцію, є:

- Bag of Words (BOW) - частота кожного слова у тексті;
- TF-IDF - зважене врахування частоти слова у документі та в корпусі;
- Word2Vec/GloVe - векторне представлення слів з урахуванням контексту;
- Embeddings із трансформерів - наприклад, з BERT або GPT.

У *процесі побудови та тренування моделі* залежно від задачі обирається відповідний алгоритм.

Точність і якість моделі потрібно оцінювати за допомогою метрик. Модель перевіряється на нових або edge-case прикладах. Потрібно провести: тестування на валідаційній вибірці, покращення за допомогою гіперпараметрів, порівняння різних моделей або підходів.

У процесі розгортання та використання модель потрібно інтегрувати у додаток або API: наприклад, для чат-ботів, пошукових систем, рекомендаторів, модерації контенту, використовуючи засоби: FastAPI, Flask, Docker, Hugging Face Transformers.

Отже, весь процес вирішення завдань з NLP можна звести до 6 кроків: Збір → Попередня обробка → Векторизація → Навчання → Оцінка → Розгортання.

2.2 Застосування машинного навчання до NLP-задач

Застосування машинного навчання до NLP-задач відбувається через побудову моделей машинного навчання, які вчаться розпізнавати закономірності у текстових даних. Досліджуючи покроковий процес вищевказаного застосування, в рамках дипломної роботи було визначено, яким чином він відбувається.

Спочатку відбувається постановка NLP-задачі та визначаються цілі. З вище проведеного аналізу було визначено, що найрозповсюдженішими задачами є:

- класифікація тексту (теми, емоції, спам);
- аналіз тональності;
- розпізнавання сутностей (імен, дат тощо);
- переклад, резюмування, генерація тексту;
- витяг інформації або відповіді на питання.

Після цього необхідно виконати збір і розмітку текстових даних. Дані - це головне у ML. Будь-яке завдання машинного навчання починається з даних. Джерелами даних є контент, створений користувачами, соцмережі, відгуки, новини, бази даних, діагностична інформація (запити користувачів).

Попередня обробка тексту забезпечує нормалізацію даних: очистку (видалення спецсимволів, HTML); токенізацію (розбиття на слова); лематизацію або стемінг (зведення до основи); видалення стоп-слів (“і”, “в”, “на”); перетворення регістру (все в нижній), іноді балансування класів, усунення дублів.

Далі забезпечується перетворення тексту у числову форму, так як машина не працює напряму з текстом - потрібно його векторизувати. Методи векторизації тексту приведені у табл. 2.1.

Таблиця 2.1

Методи векторизації тексту

№ п/п	Метод векторизації	Призначення
1	Bag of Words (BOW)	Представлення тексту частотами слів
2	TF-IDF	Враховує важливість слова в тексті й у корпусі
3	Word2Vec / GloVe	Контекстні вектори слів
4	Transformers (BERT, GPT)	Високоякісні ембеддінги для слів і речень

На сьогоднішній день практично не існує повноцінних рекомендацій та порад на тему того, як ефективно справлятися із завданнями NLP, все треба проводити за допомогою досліджень.

2.3 Дослідження методу векторизації тексту Bag of Words (BoW)

У межах бакалаврської кваліфікаційної роботи виконаємо дослідження методу векторизації тексту Bag of Words (BoW).

BoW розглядає текст як "мішок слів" (bag - мішок), ігноруючи порядок слів, граматику, пунктуацію, а натомість рахує, скільки разів кожне слово зустрічається у тексті.

Метод BoW включає наступні етапи: збір корпусу (набір документів) – крок 1, побудову словника – крок 2, побудову векторів – крок 3.

Припустимо на кроці 1 у нас є три речення:

1) "Я люблю NLP".
2) "NLP - це цікаво".
3) "Я люблю машинне навчання".

На кроці 2 здійснюється побудова словника, тобто потрібно взяти всі унікальні слова з усіх речень і це буде нашим словником (vocabulary):

["я", "люблю", "nlp", "це", "цікаво", "машинне", "навчання"]

На кроці 3 кожне речення представляється у вигляді вектора з чисел, де кожен елемент - це кількість разів, скільки слово із словника зустрічається у реченні (табл. 2.2).

Таблиця 2.2

Побудова векторів

Речення	я	люблю	nlp	це	цікаво	машинне	навчання
Я люблю NLP	1	1	1	0	0	0	0
NLP — це цікаво	0	0	1	1	1	0	0
Я люблю машинне навчання	1	1	0	0	0	1	1

Bow - це матриця ознак, яку можна використовувати у машинному навчанні. Кожне речення - це рядок з чисел (вектор) і тепер комп’ютер "розуміє", що таке текст у числовій формі.

Перевагами методу Bow є:

- простота реалізації;
- добра робота з простими задачами;
- добре працює з невеликим набором даних;
- швидке використання;
- дозволяє зрозуміти, як слова впливають на зміст;
- сумісний із класичними моделями ML.

На ряду з перевагами доцільно виділити і недоліки методу Bow:

- ігнорування порядку слів;
- не враховування значення/контексту слів або синонімів;
- великі словники → великі та розріджені матриці (sparse) / велика розмірність словника при великому корпусі.

Метод BoW можна використовувати для класифікації текстів (наприклад, спам/не спам), аналізу тональності, у пошукових системах (для пошуку за схожістю ключових слів) та для початкового навчання моделей для невеликих задач.

Виконаємо дослідження задачі NLP - класифікація тональності тексту (Sentiment Analysis), метою якої є автоматично визначати, чи має текст позитивну або негативну тональність.

В якості прикладу сформулюємо два речення:

1) "Цей фільм чудовий" → позитивна тональність.
2) "Цей фільм жахливий" → негативна тональність.

Задачею є навчити модель передбачати тональність нового речення, наприклад: "Фільм був чудовий".

Для того, щоб вирішити задачу за допомогою BoW (Bag of Words), необхідно:

- створити словник (вхідні ознаки);
- побудувати вектор ознак (векторизація);

- виконати навчання класифікатора (математично);
- здійснити передбачення для нового тексту.

Для того, щоб створити словник (вхідних ознак) необхідно знайти всі унікальні слова у навчальному корпусі:

[цей, фільм, чудовий, жахливий, був]

Для побудови вектора ознак (векторизації) потрібно кожне речення подати як вектор із частотами входження слів зі словника.

Вектор для словосполучення "Цей фільм чудовий":

- цей (1), фільм (1), чудовий (1), жахливий (0), був (0)
  → **[1, 1, 1, 0, 0]**

Вектор для словосполучення "Цей фільм жахливий":

- цей (1), фільм (1), чудовий (0), жахливий (1), був (0)
  → **[1, 1, 0, 1, 0]**

Вектор для словосполучення "Фільм був чудовий":

- цей (0), фільм (1), чудовий (1), жахливий (0), був (1)
  → **[0, 1, 1, 0, 1]** ← новий текст для передбачення

У процесі навчання класифікатора (математично) ці вектори можна використати як вхідні ознаки для моделі машинного навчання, наприклад, логістичної регресії або наївного Байєсівського класифікатора.

Виконаємо формалізацію задачі. Для цього введемо позначення:

- $\mathbf{x}_1 = [1, 1, 1, 0, 0]$ — позитивний (мітка $y_1 = 1$)
- $\mathbf{x}_2 = [1, 1, 0, 1, 0]$ — негативний (мітка $y_2 = 0$)

(2.1)

Задача - знайти таку функцію:

$$\hat{y} = f(\mathbf{x}) = \sigma(\mathbf{w}^\top \mathbf{x} + b) \ , \tag{2.2}$$

де w – вектор вагів для кожного слова;

b - зсув (bias);

σ - сигмоїдна функція:

$$\sigma(z) = \frac{1}{1 + e^{-z}} \quad . \tag{2.3}$$

Навчання полягає у підборі вагів *w* так, щоб для позитивних прикладів вихід ý був близький до 1, а для негативних — до 0.

Виконаємо дослідження передбачення для нового тексту. Речення "Фільм був чудовий" → вектор:

$$\mathbf{x}_3 = [0, 1, 1, 0, 1] \quad . \tag{2.4}$$

Обчислюємо:

$$\hat{y}_3 = \sigma(\mathbf{w}^\top \mathbf{x}_3 + b) \quad . \tag{2.5}$$

Якщо ý$_3$ > 0.5 → позитивне;

Якщо ý$_3$ < 0.5 → негативне.

Виконаємо дослідження цієї ж задачі класифікації тональності тексту, але простішим і більш інтуїтивним методом — підрахунком “позитивних” і “негативних” слів у реченні, без формальної математики.

Задачею є визначити, чи має речення позитивну або негативну тональність (табл. 2.3).

Таблиця 2.3

Визначення тональності речення

№ п/п	Речення	Тональність
1	"Цей фільм чудовий"	*Позитивна*
2	"Цей фільм жахливий"	*Негативна*

Крок 1. Побудуємо словник тональності. Створимо вручну список позитивних і негативних слів, який називається словником тональностей:

- позитивні слова: чудовий;

- негативні слова: жахливий.

Це простий спосіб дати моделі знання про "емоційне значення" слів.

Крок 2. Аналіз нового речення. Нове речення: "Фільм був чудовий". Розбиваємо речення на слова:

["фільм", "був", "чудовий"]

Рахуємо позитивні слова:

- у реченні є слово "чудовий", яке у нашому списку позитивне;
- позитивних слів: 1.

Рахуємо негативні слова:

- негативних слів: 0.

Крок 3. Прийняття рішення. Правило:

- якщо позитивних слів > негативних → позитивна тональність;
- якщо негативних слів > позитивних → негативна тональність;
- якщо рівна кількість слів або жодного слова - нейтральна тональність (за потреби).

У нашому прикладі:

- позитивні: 1;
- негативні: 0.

Висновок: речення позитивне.

У табл. 2.4 приведений приклад визначення тональності тексту.

Таблиця 2.4

Визначення тональності тексту

№ п/п	Речення	Чудовий	Жахливий	Результат
1	Цей фільм чудовий	1	0	Позитивна
2	Цей фільм жахливий	0	1	Негативна
3	Фільм був чудовий	1	0	Позитивна
4	Фільм був жахливий	0	1	Негативна
5	Фільм був нормальний	0	0	Нейтральна

Отримано повноцінний підхід до машинного навчання на основі BoW, який можна реалізувати в Excel, Jupyter Notebook або у Google Colab. У результаті буде отримана оцінка точності моделі на тестових даних, можна буде побачити звіт, де буде видно, як добре модель передбачає позитивну й негативну тональність.

2.4 Дослідження методу векторизації тексту TF-IDF

Метод TF-IDF (Term Frequency - Inverse Document Frequency) - це популярна техніка векторизації тексту, яка дозволяє представити документи у вигляді числових векторів з урахуванням важливості слів у межах одного документа та всього корпусу текстів.

TF-IDF оцінює важливість слова документі так, щоб:

- часті слова у цьому документі мали вищу вагу;
- загальновживані слова (наприклад, “це”, “і”, “в”) мали меншу вагу.

TF-IDF для слова t у документі d з корпусу D:

$$\text{TF-IDF}(t,d) = \underbrace{\text{TF}(t,d)}_{\text{частота терміна}} \times \underbrace{\text{IDF}(t,D)}_{\text{зворотна частота в документах}} \tag{2.6}$$

TF (Term Frequency) - частота слова у документі:

$$\text{TF}(t,d) = \frac{\text{Кількість появ слова } t \text{ у документі } d}{\text{Загальна кількість слів у документі } d} \tag{2.7}$$

IDF (Inverse Document Frequency) - зниження ваги частих слів:

$$\text{IDF}(t,D) = \log\left(\frac{N}{1+n_t}\right) \ , \tag{2.8}$$

де N - загальна кількість документів;

n_t - кількість документів, у яких є термін t.

Дослідимо приклад практичної NLP-задачі, яку можна вирішити за допомогою методу TF-IDF, наприклад, автоматична класифікація текстів студентських відгуків.

Формулювання. Маємо набір коротких студентських відгуків про викладачів або курси. Потрібно автоматично визначити, позитивний це відгук чи негативний, використовуючи векторизацію тексту методом TF-IDF і класифікацію логістичною регресією або SVM.

Вхідні дані представлено у табл. 2.5.

Таблиця 2.5

Вхідні дані

Відгук	Мітка
"Викладач пояснює доступно та цікаво."	1 (позитивний)
"Матеріал неструктурований, постійно відволікається."	0 (негативний)
"Дуже корисний курс, раджу всім."	1
"Не зрозуміло, навіщо цей предмет взагалі."	0

Етапи вирішення вищепоставленої задачі:

1) Препроцесинг текстів: перевести у нижній регістр, видалити знаки пунктуації, токенізувати, лематизувати/стемінг (опціонально). Наприклад:

"Дуже корисний курс, раджу всім" → ["дуже", "корисний", "курс", "раджу", "всім"]

2) Обчислюємо TF-IDF матрицю: кожен документ (відгук) перетворюється у вектор, де кожен елемент - це значення TF-IDF конкретного слова. Розмір матриці:

[кількість відгуків × кількість унікальних слів]

Наприклад: "дуже" може мати низьке значення (загальновживане слово), "раджу" - високе (рідше і корисніше для класифікації).

3) Навчання моделі. Подаємо вектори TF-IDF + мітки (0 або 1) в алгоритм машинного навчання (наприклад, логістична регресія). Модель вчиться визначати шаблони між словами і типом відгуку.

4) Прогнозування. Новий відгук → TF-IDF → модель прогнозує: позитивний чи негативний.

Переваги задачі для студентів: простий і зрозумілий приклад із реального життя; видно всю логіку: від тексту до математики; можна реалізувати як в Excel, так і в Jupyter Notebook / Google Colab.

2.5 Методи векторизації Word2Vec / GloVe

Методи векторизації тексту Word2Vec і GloVe - це просунуті техніки представлення слів у вигляді числових векторів, які зберігають семантичні й синтаксичні зв'язки між словами. Це значно глибші методи порівняно з BoW або TF-IDF, оскільки вони враховують контекст.

Word2Vec створює векторне представлення слів на основі їх контексту. Воно навчається за допомогою нейромережі й формує вектори так, щоб слова, які часто з’являються разом, були близько одне до одного у просторі. Основними архітектурами даного методу є:

- CBOW (Continuous Bag of Words): передбачає слово за його контекстом (оточуючими словами) (рис. 2.1);
- Skip-Gram: навпаки передбачає контекст за словом (рис. 2.2).

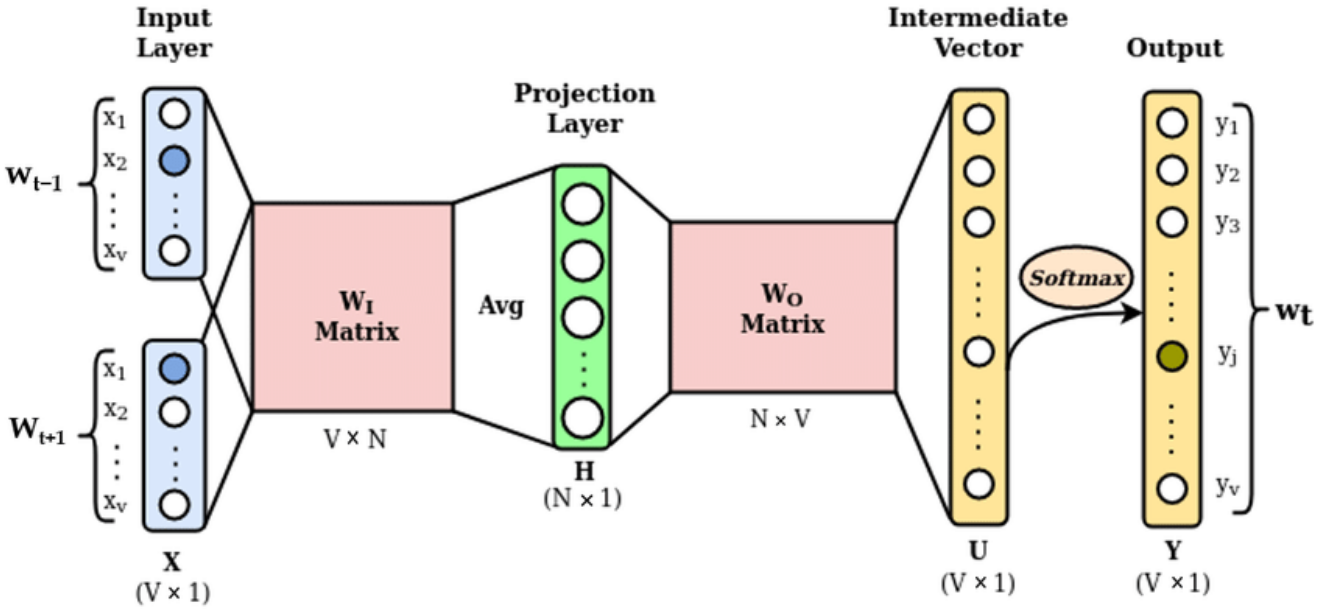

Рис. 2.1 Архітектура CBOW

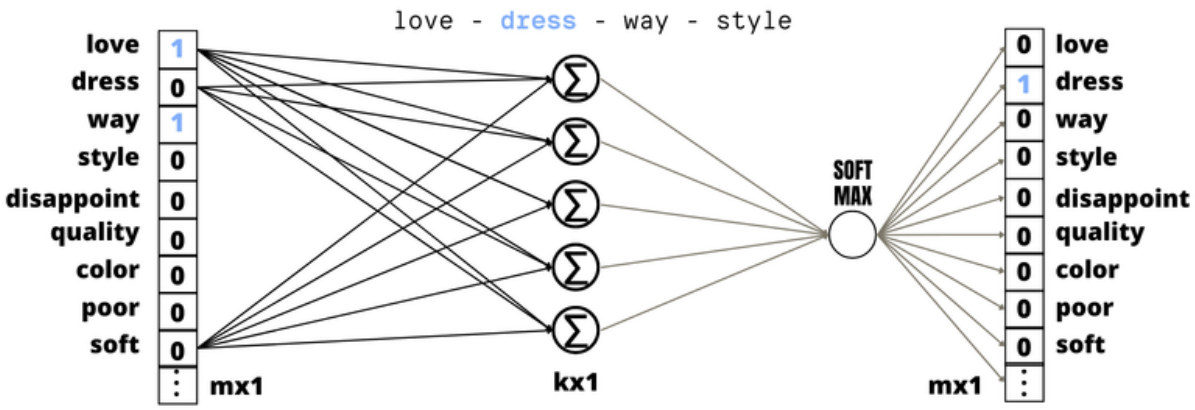

Рис. 2.2 Архітектура Skip-Gram

На відміну від Word2Vec, GloVe поєднує глобальну статистику тексту - частоти співзустрічей слів у текстах - із локальним контекстом. Це матричний підхід, де вивчається співзустріч слів по всьому корпусу. Даний метод зберігає глобальні закономірності. Вектори формуються після аналізу всієї статистики, а не лише локального контексту [14, 16].

Порівняння методів Word2Vec та GloVe приведене у табл. 2.6.

Таблиця 2.6

Порівняння Word2Vec та GloVe

Властивість	Word2Vec	GloVe
Підхід	Передбачення (нейромережа)	Факторизація матриці
Контекст	Локальний	Глобальний
Навчання	Online	Offline (раз і назавжди)
Швидкість	Швидке	Повільніше
Якість представлень	Висока	Висока

Вищевказані вектори використовуються для класифікація текстів, пошуком за змістом, у перекладах, чат-ботах, для виявлення схожості між словами/фразами та підживлення моделей LSTM, CNN, Transformer.

3 РОЗРОБКА РЕКОМЕНДАЦІЙ ДЛЯ ОРГАНІЗАЦІЇ НАВЧАННЯ ТА ДОСЛІДЖЕНЬ В ОБЛАСТІ NLP

3.1 Визначення навиків спеціаліста з NLP

Фахівець у галузі Natural Language Processing (NLP) має володіти поєднанням технічних, лінгвістичних та аналітичних навичок. Визначимо ключові навики спеціалістів у даному напрямку.

Фахівці в галузі NLP повинні володіти навиками програмування. Основною мовою для реалізації задач в області NLP є мова програмування Python. Іноді для оптимізації рішень використовуються мови програмування C++, Java або R.

Для базової обробки тексту необхідно вивчати роботу з бібліотеками NLTK, spaCy, TextBlob. Для обробки даних і класичного ML доцільно вивчити роботу з бібліотеками Scikit-learn, NumPy, Pandas. Нейронні мережі та глибоке навчання доцільно вивчати на фреймворках TensorFlow, PyTorch, Transformers (Hugging Face).

З технологій машинного навчання необхідно знати:

- задачі класифікації, кластеризації, регресії;
- нейронні мережі, регуляризацію, крос-валідацію;
- NLP-моделі: Word2Vec, BERT, GPT, LSTM.

Також спеціалісту з NLP потрібно розумітися на:

- алгоритмах обробки тексту: токенізація, стемінг, лематизація, видалення стоп-слів;
- методах векторизації: Bag of Words, TF-IDF, Word2Vec, GloVe, BERT embeddings;
- математичному апараті: лінійна алгебра, ймовірність, статистика;
- теорії мов та граматиці: синтаксис, морфології, семантиці.

З практичних навиків потрібно вміти: працювати з датасетами, анотацією текстів; вміти виконувати NLP-задачі: класифікацію тексту, пошук інформації, генерацію тексту, переклад, аналіз тональності; вміти здійснювати оцінку якості моделей: accuracy, precision, recall, F1-score, confusion matrix.

Затребуваному спеціалісту в NLP-області необхідні навики роботи з великими обсягами даних (Big Data, Spark); хмарними сервісами: Google Colab, AWS, Azure для запуску моделей; потрібно розумітися на етичних аспектах NLP: упередженості даних, приватності.

Для фахівця з NLP-технологій важливими є критичне мислення, здатність працювати з неоднозначними задачами, комунікаційні навички для того, щоб простими методами пояснювати складні поставлені завдання.

3.2 Обгрунтування вибору мови програмування

Проводячи дослідження NLP-технологій в межах бакалаврської кваліфікаційної роботи було визначено, що поєднання Python та Jupyter Notebook забезпечуватиме оптимальне середовище для розробки NLP-моделей, яке дозволить ефективно експериментувати, будувати та тестувати різноманітні методи обробки природної мови [23].

Ще одним середовищем, яке доцільно використовувати для розробки моделей в ML/NLP – галузях є Google Colab.

Python є провідною мовою програмування в галузі машинного навчання та обробки природної мови з наступних причин:

- містить потужні та зручні бібліотеки для NLP, такі як NLTK, spaCy, TextBlob, gensim, transformers;
- підтримує моделі глибокого навчання: TensorFlow, PyTorch, Keras.

Код на Python легко читається та швидко розробляється, що особливо важливо для досліджень та прототипування NLP-моделей. Платформи типу Hugging Face, OpenAI, AllenNLP підтримують Python як основну мову (рис. 3.1).

Методика вивчення мови програмування Python для подальшого використання у машинному навчанні та задачах, які пов’язані із застосування NLP-технологій представлена нижче (табл. 3.1):

1. Введення в Python.
2. Примітивні типи даних та змінні.
3. Умовні оператори.

4. Цикли.

5. Списки та зрізи.

6. Словники.

7. Множини та кортежі.

8. Функції 1.

9. Функції 2.

10. Виключення та їх обробка.

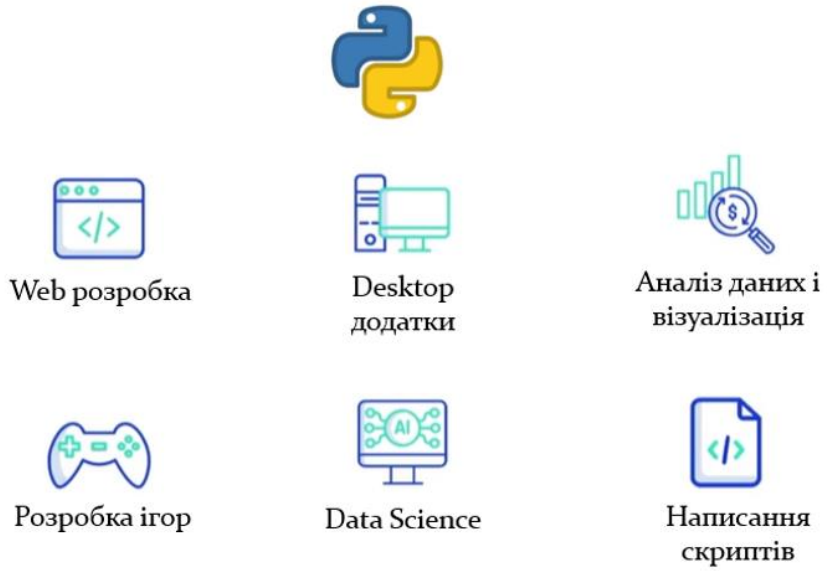

Рис. 3.1 Розповсюдженні області застосування мови програмування Python

Таблиця 3.1

Python для NLP- задач

№ п/п	Тема	Питання
1	Введення в Python	1. Використання Python. 2. Встановлення Python на Windows та Linux. 3. Встановлення середовища розробки.
2	Примітивні типи даних та змінні	1. Змінні та ключові слова. 2. Рядковий тип даних. 3. Числовий тип даних. 4. Логічні типи даних.
3	Умовні оператори	1. Умовний оператор if-else. 2. Тернарні оператори. 3. Приклади умовних операторів.

Продовження таблиці 3.1

№ п/п	Тема	Питання
4	Цикли	1. Поняття циклів. 2. Цикли while та for. 3. Оператори break та continue. 4. Приклади роботи з циклами.
5	Списки та зрізи	1. Поняття списків. 2. Зрізи списків. 3. Методи списків. 4. Обхід списків. 5. Приклади роботи зі списками.
6	Словники	1. Методи словників. 2. Обхід словників. 3. Приклади роботи із словниками.
7	Множини та кортежі	1. Методи множин. 2. Методи кортежів. 3. Приклади роботи з множинами та кортежами.
8	Функції 1	1. Створення та виклик функцій. 2. Передавання параметрів у функції. 3. Повернення результатів функцій. 4. Приклади роботи з функціями.
9	Функції 2	1. Вкладені функції. 2. Область видимості функцій. 3. Стандартні функції Python. 4. Анонімні функції. 5. Рекурсія. 6. Приклади роботи з функціями.
10	Виключення та їх обробка	1. Поняття виключень. 2. Обробка виключень. 3. Види виключень. 4. Приклади виключень та їх обробка.

Потрібно знати принципи об’єктно-орієнтованого програмування (ОПП), інкапсуляцію у Python, реалізацію поліморфізму, реалізацію абстракції у Python,

принципи роботи з рекурсією, ускладнені варіації комбінацій структур даних, вбудовані модулі, PEP, стандарти оформлення коду у мові Python, метакласи та сфери їх застосування, можливості використання типізації в Python, специфікації PEP 8, придбання навиків написання простого та читабельного коду.

3.3 Обгрунтування вибору середовища роботи з NLP-моделями

Станом на сьогоднішній день затребуваними середовищами роботи з NLP-моделями є: Google Colab та Jupyter Notebook.

Google Colab – безкоштовне інтерактивне хмарне середовище розробки від Google. Фізично воно знаходиться на просторах Google, на їх серверах. Починаючи роботу з Google Colab, ми отримуємо доступ до віддаленої машини, де розгортається віртуальна машина з піднятим на ній Jupyter Notebook (рис. 3.2), а це і є інтерактивне середовище розробки.

Рис. 3.2 Віртуальна машина з Jupyter Notebook

Тут є комірки з кодом, які можемо виконувати і також тут є комірки з текстом, що дуже зручно для фіксації своїх ідей та результатів експериментів.

Для того, щоб запустити Google Colab - можна у будь-якій пошуковій системі набрати Google Colab і перейти буквально за першим посиланням. Щоб створити ноутбук, потрібно в меню вибрати файл і потім створити блокнот. У вкладці з’явиться новий чистий ноутбук, який створюється на гугл диску (рис.3.3). Щоб дізнатися його розміщення, достатньо у вкладці «Файл» перейти на «Показати місцеположення на диску» (рис. 3.4). Створений ноутбук можна переміщати в межах диску. Ноутбуком у Google Colab дуже легко ділитися з іншими людьми. Достатньо натиснути на кнопку «Поділитися» і вибрати, з ким ми хочемо

поділитися ноутбуком. Можна зробити доступ за посиланням для всіх, якщо натиснути на «Дозволити доступ всім у кого є посилання». Тоді з'явиться посилання на наш ноутбук і права доступу.

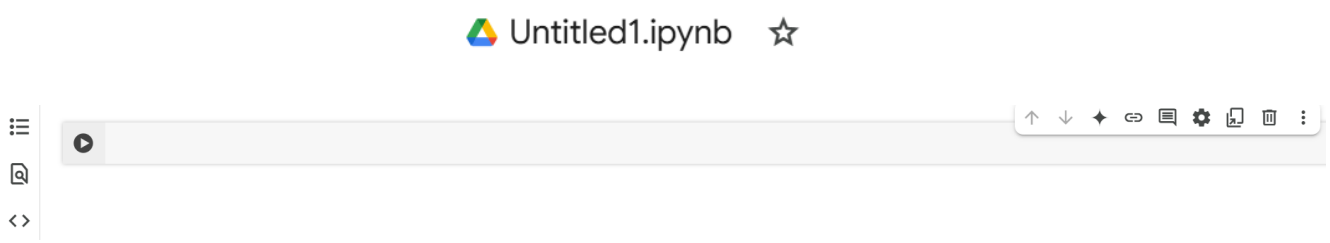

Рис. 3.3 Середовище Google Colab

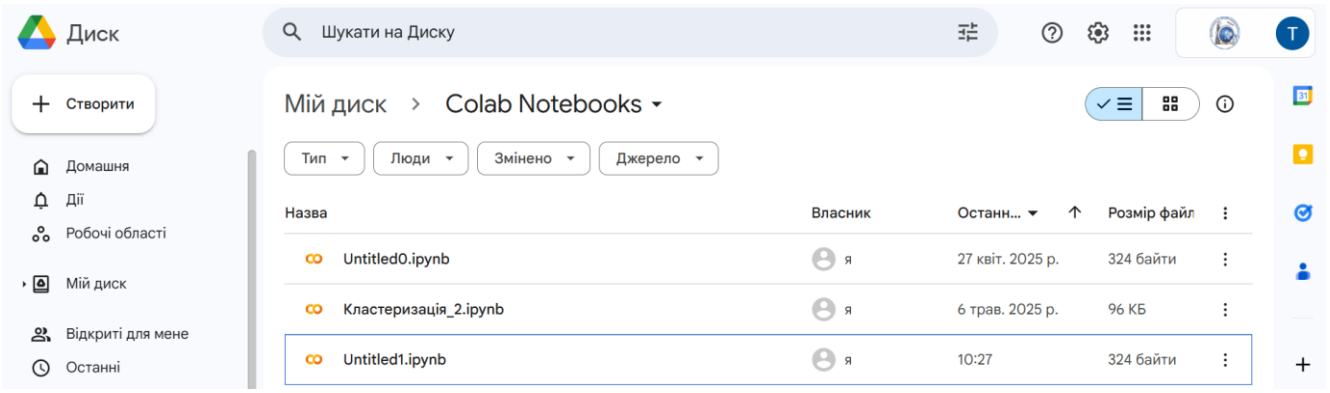

Рис. 3.4 Місцеположення ноутбука на диску

Можна зробити людей не читачами, а коментаторами або редакторами. Але ніколи не потрібно надавати права доступу на редагування неперевіреним людям. Якщо нам надають доступ як читачу, то редагувати наданий ноутбук у нас не вийде. Тому тут ми можемо створити свою копію на Google Disk. Робиться це через Файл → створити копію на Диску. Щоб запустити код в інтерактивному середовищі, потрібно натиснути на кнопку «Підключитися» або запустити будь-яку комірку.

Щоб виконати комірку – треба натиснути на кнопку Play поряд з коміркою, або ж натиснути поєднання клавіш:

Ctrl + Enter – виконується комірка і виділення переходить на комірку нижче
Shift + Enter – виконується комірка і виділення залишається у цій комірці
Alt + Enter – виконується комірка і створюється нова під нею, на неї і переходить виділення

Щоб створити нову комірку з кодом, можна натиснути на випливаючу кнопку + код. Або ж можна натиснути на клавішу *A* для створення комірки над поточною. Головне – знаходитися у потрібному режимі: режимі *введення команд*, коли можна спокійно переміщатися по коміркам, а не в *режимі редагування*, коли можна переміщатися по вмісту однієї комірки. *A* – створення нової комірки зверху. *B* – створення нової комірки знизу. Якщо необхідно із кодової комірки зробити текстову – можна натиснути на поєднання клавіш *Ctrl+m+m*, тоді кодова комірка стане текстовою. Переведення комірки в код здійснюється комбінацією клавіш: *Ctrl+m+y*. Для видалення комірок можна використовувати команду: *Ctrl+m+d*.

Особливістю середовища Google Colab є те, що це віддалена віртуальна машина, з якою ми фізично не контактуємо, а значить ми не можемо взяти флешку й перенести на неї набір даних. Ми не знаємо, де знаходиться цей комп'ютер. Цю проблему можна вирішити трьома способами.

Перший спосіб – це перенести наші локальні файли з папки у ліву область у Google Colab.

Другий варіант - скористатися модулем Files із бібліотеки Google Colab.

Щоб не втратити файли, потрібно користуватися третім і самим надійним варіантом – приєднання гугл диску до віртуальної машини. Рекомендовано користуватися саме цим підходом для роботи з файлами, так як він є самим надійним та перевіреним.

Ще одним продуктивним середовищем для роботи як з технологіями ML/DL, так і в області NLP є середовище Jupyter Notebook. Jupyter Notebook – інтерактивне середовище, де є комірки з текстом і є комірки з кодом. Код можна виконувати, комірки з текстом можна змінювати. При цьому у комірці з текстом можна вказувати різні параметри форматування, наприклад, робити заголовки, заголовки вимірюються решітками. Також можна робити текст жирним, можна виділяти його курсивом, можна добавляти код, вставляти різні посилання, картинки, відступи, таблиці, списки та інше з HTML та CSS.

Jupyter Notebook є одним із кращих інструментів для машинного навчання та досліджень в областях Deep Learning, Data Science, NLP.

Основна мова програмування для ML – це Python. Для програмування на ньому потрібне середовище розробки. Щоб використовувати Jupyter Notebook, потрібно завантажити і встановити набір бібліотек для ML – Anaconda, куди якраз і входить Jupyter Notebook (рис. 3.5).

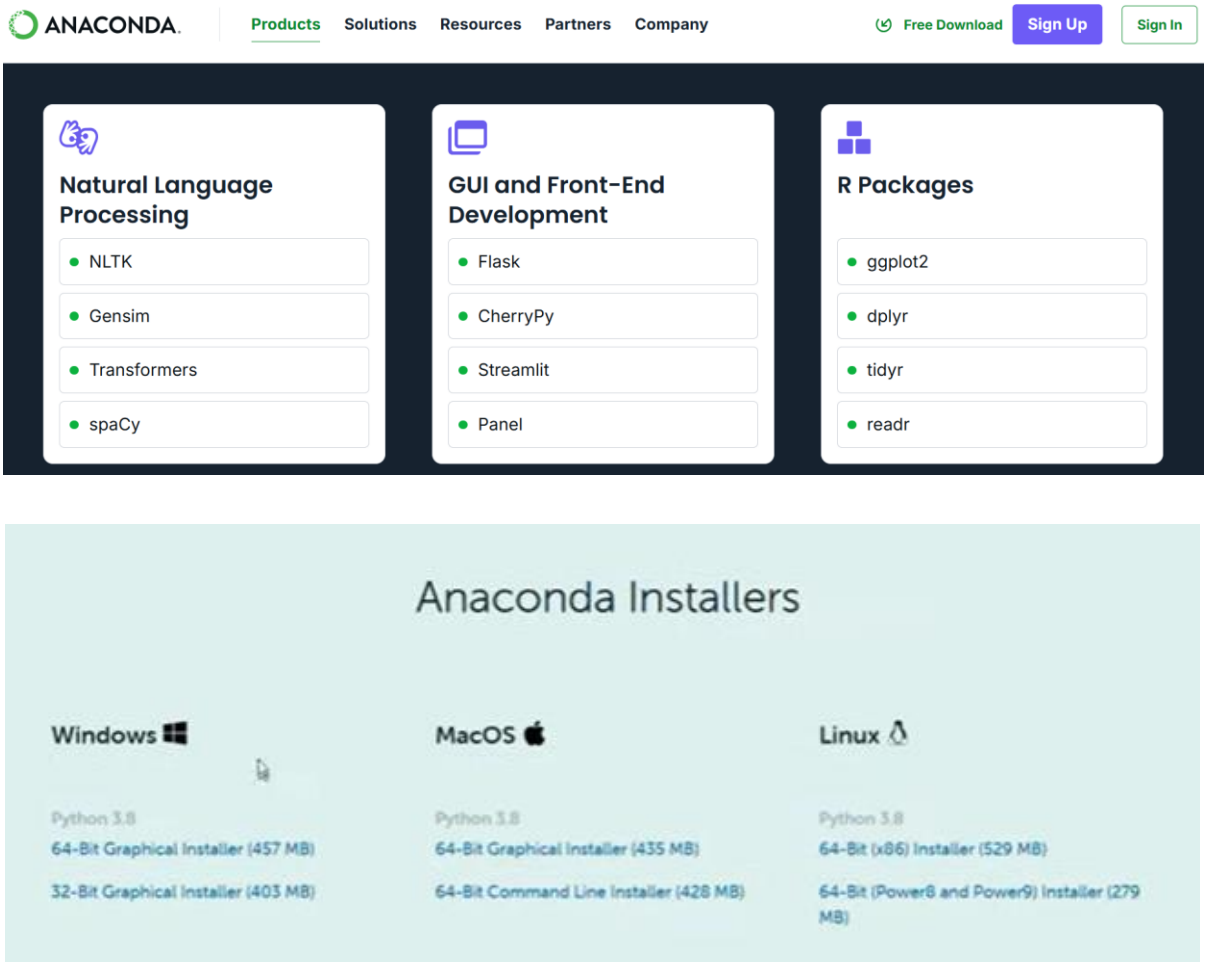

Рис. 3.5 Платформа Anaconda

Ноутбук відкривається у новій вкладці. Вгорі – назва ноутбука. За замовчуванням ноутбук називається Untitled. Назву можна змінити (рис. 3.6).

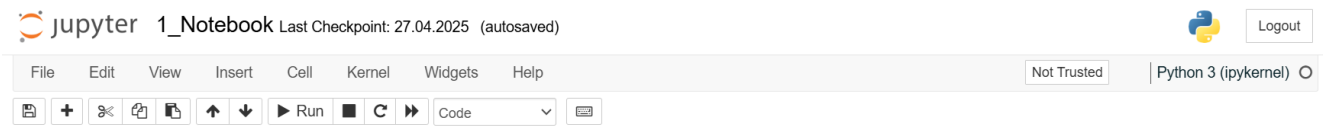

Рис. 3.6 Середовище Jupyter Notebook

Під назвою розміщується рядок меню для дій з ноутбуком, комірками та ядром, а також меню допомоги (Help). Нижче розміщується панель інструментів з кнопками для найчастіших дій. Якщо затримати курсор на кнопці, то з'явиться функція цієї кнопки. Робочу область ноутбука складають комірки. Їх можна скільки завгодно вставляти, видаляти (+, ножиці), копіювати. Є 2 режими роботи ноутбука: перший – це режим редагування, коли ми всередині комірки вводимо код. У цей час комірки виділяються зеленим кольором, а у правому верхньому куті з'являється олівець (рис. 3.7).

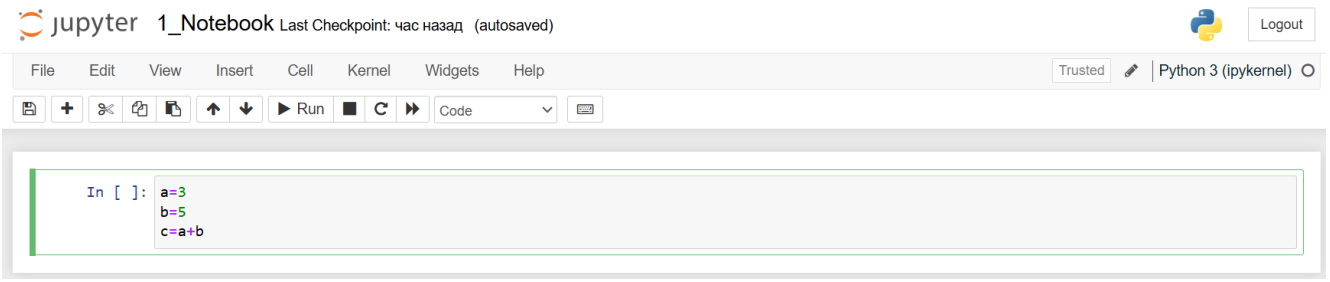

Рис. 3.7 Робота в середовищі Jupyter Notebook (1)

Для того, щоб код виконався, потрібно запустити комірку. Робиться це або клавішею «Запуск» (Run) або комбінацією клавіш Shift+Enter. Коли комірка виконалася, їй присвоюється порядковий номер і виділення переходить на наступну комірку (рис. 3.8).

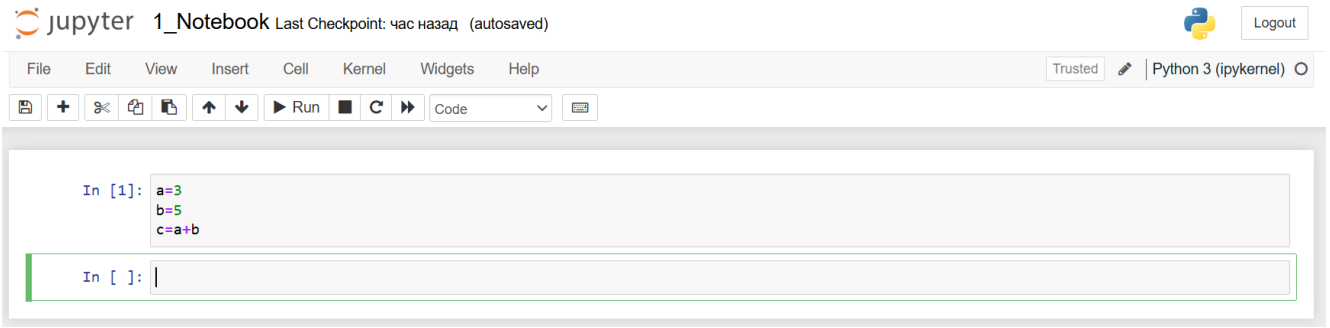

Рис. 3.8 Робота в середовищі Jupyter Notebook (2)

Також у ноутбуці можна дивитися виведення, якщо вони є. Другий режим роботи ноутбука – це командний режим. Щоб у нього перейти з режиму

редагування, потрібно натиснути Esc. Тоді у нас комірка виділяється синім кольором і у правому верхньому куті зникає значок олівця (рис. 3.9).

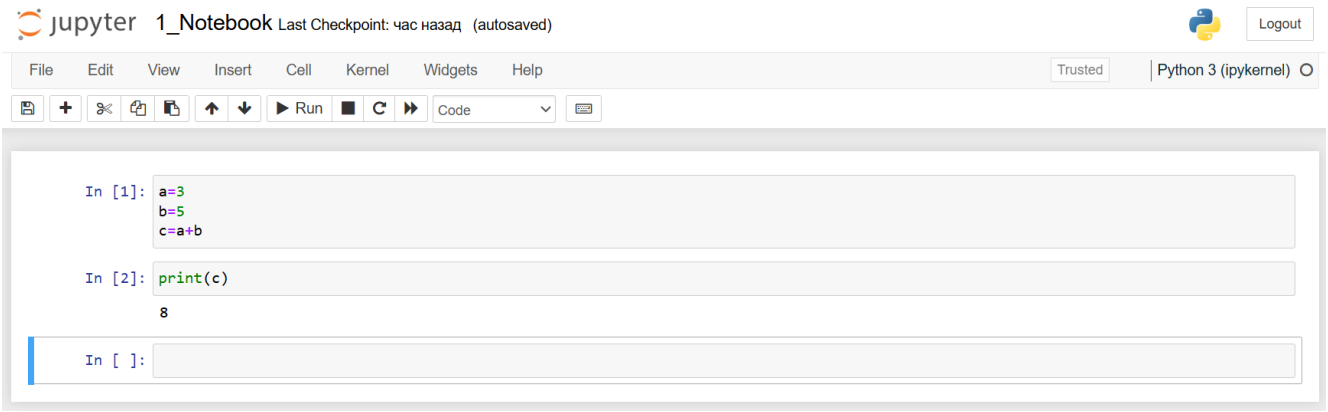

Рис. 3.9 Командний режим роботи ноутбука

У цьому режимі можна вставляти нові комірки за допомогою + на панелі інструментів, або комбінацією клавіш *Ctrl+B*. Для того, щоб видалити комірки, потрібно натиснути комбінацію клавіш *CTRL+DD*. Для того, щоб скопіювати комірку, потрібно її вибрати, натиснути *Ctrl+C*, перейти на потрібну позицію і натиснути *Ctrl+V*. Також у командному режимі можна виділяти декілька комірок зразу, затиснувши при цьому *Shift* і натискаючи на стрілочки вгору або вниз: Shift+↑↓ (рис. 3.10).

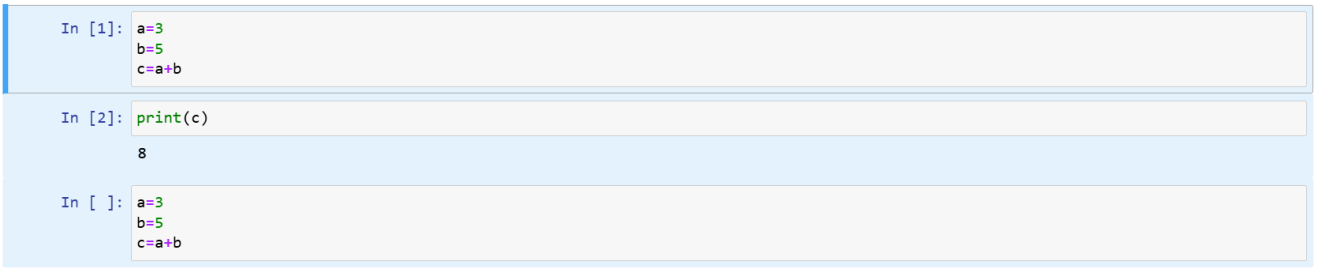

Рис. 3.10 Робота в середовищі Jupyter Notebook (3)

Також можна переміщати комірки, натискаючи на панелі інструментів ↑↓ (рис. 3.11).

Рис. 3.11 Переміщення комірок в середовищі

Можна писати коментарі до коду всередині комірок за допомогою знаку #. Коментар виділяється зеленим кольором та курсивом (рис. 3.12).

```
In [4]: # коментар до коду
        a*=10
        a
Out[4]: 30
```

Рис. 3.12 Коментарі в коді

У ноутбуці можна виводити різні графіки. Для виведення графіків потрібно імпортувати бібліотеку Matplotlib (рис. 3.13).

```
In [5]: import matplotlib.pyplot as plt
```

Рис. 3.13 Імпорт бібліотеки Matplotlib

Matplotlib — це одна з найпопулярніших бібліотек для візуалізації даних у Python, яка корисна в процесі аналізу, обробки та візуалізації текстових даних. В NLP бібліотека Matplotlib використовується для:

- візуалізації частоти слів;
- побудови гістограм довжини речень або слів;
- візуалізації точок у векторному просторі;
- аналізу результатів класифікації.

Основними функціями бібліотеки Matplotlib є: побудова графіків, налаштування осей, підписів, сітки, збереження графіків у зображеннях, створення декількох графіків одночасно, підтримка векторної графіки та високої якості друку.

У процесі вирішення NLP-задач Matplotlib дає змогу візуально оцінити розподіли, закономірності, результати класифікацій; полегшує аналіз текстових даних до та після попередньої обробки; працює у зв’язці з Pandas, Numpy, Seaborn, що розширює її можливості.

Якщо задати координати, можна зобразити графік (рис. 3.14).

```
In [6]: x=[-2,-1,0,1,2]
        y=[4,1,0,1,4]
```

```
In [7]: plt.plot(x,y)
Out[7]: [<matplotlib.lines.Line2D at 0x19fe28f8460>]
```

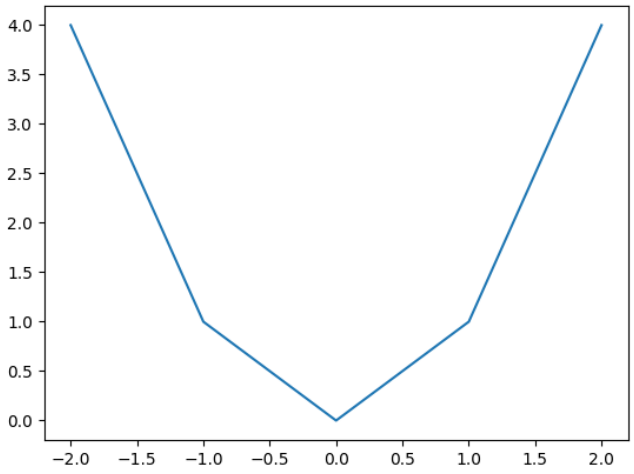

Рис. 3.14 Побудова графіку з використанням Matplotlib

У ноутбуці можна вводити консольні команди. Особливо це зручно в операційній системі Linux. Щоб ввести команду, потрібно надрукувати знак оклику, а потім набирати команду (рис. 3.15).

```
In [8]: ! ping www.google.com

        ЋЎ¬ґ Ї Єґв ¬Ё б www.google.com [142.250.203.196] б 32 Ў ©в ¬Ё ¤ ле:
        Ћвўґв ®в 142.250.203.196: зЁб«® Ў ©в=32 ўаґ¬п=17¬б TTL=114
        Ћвўґв ®в 142.250.203.196: зЁб«® Ў ©в=32 ўаґ¬п=18¬б TTL=114
        Ћвўґв ®в 142.250.203.196: зЁб«® Ў ©в=32 ўаґ¬п=18¬б TTL=114
        Ћвўґв ®в 142.250.203.196: зЁб«® Ў ©в=32 ўаґ¬п=18¬б TTL=114

        ‘в вЁбвЁЄ  Ping ¤«п 142.250.203.196:
            Џ Єґв®ў: ®вЇа ў«ґ® = 4, Ї®«гэґ® = 4, Ї®вґап® = 0
            (0% Ї®вґам)
        ЏаЁЎ«Ё§Ёвґ«м®ґ ўаґ¬п ЇаЁґ¬ -ЇґаґЊ зЁ ў ¬б:
            ЊЁЁ¬ «м®ґ = 17¬бґЄ, Њ Єбё¬ «м®ґ = 18 ¬бґЄ, ‘аґЊґґ = 17 ¬бґЄ
```

Рис. 3.15 Консольні команди

У комірок ноутбука є 2 основних типи: Code і Markdown. Для цього за замовчуванням комірка знаходиться у режимі коду. Змінити тип можна у панелі інструментів (рис. 3.16).

У режимі Markdown можна писати текст, заголовки, формули. Markdown – мова розмітки для форматування тексту. У цьому режимі можна писати заголовки різних рівнів. Перший рівень – найбільший шрифт. Другий рівень, який є меншим, починається з двох знаків решітки. Третій рівень – ще менший і т.д.

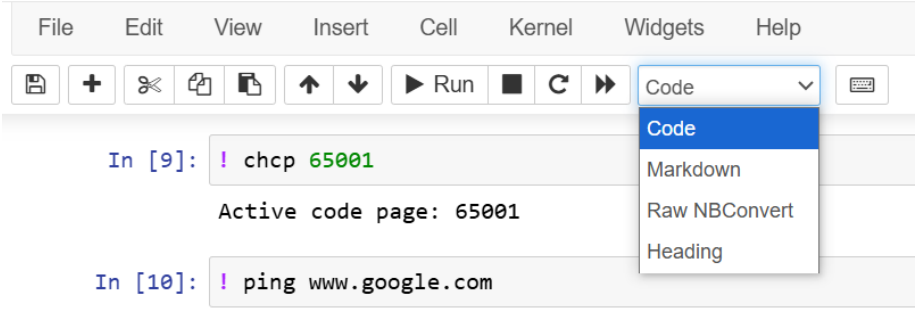

Рис. 3.16 Режими роботи Code і Markdown

Те, що ми бачимо – це ноутбук, де ми пишемо код, де код з'являється, а сам код виконується в ядрі, яке також називається Kernel. При закритті ноутбуку Kernel не закривається. Щоб повністю закрити ноутбук – закриваємо головне вікно і закриваємо консоль. У прикладі готового ноутбука є заголовки різних рівнів, текст, картинки, комірки в режимі коду і в режимі Markdown, є виводи таблиць, все підписано, є коментарі, виведення графіків, різних діаграм.

Робота з Jupyter Notebook у задачах NLP має низку особливостей, які роблять цей інструмент ідеальним середовищем для дослідження, навчання та прототипування моделей обробки природної мови.

Можна покроково обробляти текст, змінювати параметри токенізації, стоп-слів й одразу бачити результат. У зручному форматі виводяться таблиці, графіки, словники, списки. Підтримує Matplotlib, Seaborn, Plotly, що зручно для аналізу частот, хмар слів, векторних представлень. Можна вбудовувати графіки прямо в ноутбук без потреби зберігати окремі файли.

Зручно експериментувати з обробкою: текст → очищення → токенізація → лематизація → векторизація → класифікація. Можна повертатися назад, не запускаючи весь код з нуля. Можна документувати код, пояснювати кроки, вставляти формули (наприклад, tf-idf, формули ймовірностей). Корисно для навчання або презентації результатів. Можна завантажувати моделі, обробляти тексти, виводити ключові слова та семантичні зв'язки.

Задачами NLP у Jupyter є:

- попередня обробка тексту: очищення, токенізація, лематизація;

- аналіз частоти слів, побудова wordcloud;
- векторизація: Bag of Words, TF-IDF, Word2Vec, BERT;
- побудова та оцінка моделей класифікації тексту (логістична регресія, SVM, наївний Баєсівський класифікатор);
- візуалізація результатів (матриця плутанини, PCA/t-SNE на ембеддінгах);
- робота з великими корпусами даних (наприклад, IMDb, новини, Twitter).

3.4 Рекомендації для організації навчання та досліджень у сфері NLP

На основі проведеного аналізу, мною визначені рекомендації для організації навчання та досліджень в області NLP.

Таблиця 3.2

Рекомендації: як побудувати своє навчання

№ п/п	Дії	Приклад
1	Визначитися, що вам подобається в IT	Це backend/Data Science/NLP. Аналітика або виконання чітких задач
2	Вивчити мову програмування	Опанувати Python. Вивчити основи мови від «що таке цикл» до ОПП та побудови власних модулів
3	Вивчити фреймворки	В залежності від першого рішення вивчити хоча б 1-2 фреймворки
4	Зробити резюме та почати пошук роботи	Проаналізувати вакансії, що потрібно від кандидатів, зробити декілька варіантів резюме, відіслати їх, очікувати співбесіди

Таблиця 3.3

Рекомендації по навчанню: Python

№ п/п	Дії	Ресурси
1	Основи мови. Як працювати у IDE, як працює Python. Базові структури даних та синтаксичні конструкції	http://www.learnpython.org/en/Welcome - усі основи
2	Основи ООП, модулі та написання власних об'ємних програм	http://realpython.com/python3-object-oriented-programming/ - про ООП
3	Закріплення знань шляхом вирішення різних задач	http://leetcode.com/problemset/all/

Python відіграє центральну роль у сфері Natural Language Processing (NLP). Python має велику кількість бібліотек, які значно спрощують розробку NLP-рішень. Python має лаконічний і зрозумілий синтаксис, що дозволяє: швидко створювати прототипи NLP-моделей, легко читати та супроводжувати код, зосередитися на логіці обробки, а не на технічних деталях. Python тісно пов'язаний з ML/AI через бібліотеки:

- Scikit-learn - класичне машинне навчання;
- TensorFlow, PyTorch - глибоке навчання та нейронні мережі.

Усі ці бібліотеки легко інтегруються з NLP-фреймворками. Python має активну спільноту фахівців з NLP. Jupyter Notebook, Google Colab - інтерактивна розробка NLP-проєктів.

Таблиця 3.4

Рекомендації по навчанню: Python Backend

№ п/п	Дії	Ресурси
1	Вивчити основи як працює Інтернет (клієнт-сервер взаємодія, HTTP(S)). Вивчити HTML	An overview of HTTP - HTTP | MDN (mozilla.org) (що таке HTTP) How does the Internet work? - Learn web development | MDN (mozilla.org) (як працює Інтернет) HTML Basic (w3schools.com) (HTML основи)
2	Вивчити фреймворк. Зробити «по інструкції» будь-який веб-сервіс	Welcome to Flask — Flask Documentation (2.1.x) (palletsprojects.com) (Flask) The web framework for perfectionists with deadlines | Django (djangoproject.com) (Django)
3	Розробити власний веб-сервіс. Щоб точно можна було вважати себе розробником, то бонусом (для себе звісно) задеплоїти його (розмістити на хостингу)	Cloud Application Platform | Heroku (Heroku – безкоштовний хостинг)

Python є незамінним інструментом у сфері NLP завдяки своїй гнучкості, багатій екосистемі, простоті та активній підтримці спільноти. Саме тому він обирається як основна мова в академічних дослідженнях, прикладних задачах та промислових рішеннях у галузі обробки природної мови.

Бібліотеки Scikit-learn, TensorFlow та PyTorch відіграють ключову роль у створенні та навчанні моделей для задач обробки природної мови (NLP). Вони надають інструменти для реалізації як класичних алгоритмів машинного навчання, так і сучасних глибоких нейронних мереж, які широко застосовуються у NLP.

Таблиця 3.5

Рекомендації по навчанню та дослідженням в області Data Science

№ п/п	Дії	Ресурси
1	Вивчити основні фреймворки для роботи з даними. Вивчити математичну базу.	https://numpy.org/ https://pandas.pydata.org/ https://matplotlib.org/ https://scipy.org/
2	Вивчити класичні моделі машинного навчання та фреймворків. Математична база.	https://scikit-learn.org/stable/
3	Вивчення фреймворків для глибинного навчання. Математична база.	https://keras.io/ https://pytorch.org/ https://www.tensorflow.org/

TensorFlow – фреймворк для створення нейронних мереж, особливо глибоких моделей, які здатні захоплювати контекст і послідовність у тексті (рис. 3.17). Використовується для побудови:

- RNN, LSTM, GRU;
- Seq2Seq моделей;
- трансформерів (власна реалізація або через TensorFlow Hub).

Data Science (наука про дані) та NLP тісно пов’язані, оскільки NLP є однією з ключових підгалузей Data Science, яка працює саме з текстовими (неструктурованими) даними. У Data Science працюють із числовими, категоріальними, зображеннями, аудіо та текстовими даними. NLP - це підхід до

перетворення тексту у форму, придатну для аналізу (наприклад, у числові вектори).

Аналіз робочих процесів Data Science та NLP представлений у табл. 3.6.

Таблиця 3.6

Робочі процеси Data Science та NLP

№ п/п	Крок	Data Science	NLP
1	Збір даних	Дані з API, БД, файлів	Текст з сайтів, соцмереж, PDF, чату
2	Попередня обробка	Очистка, нормалізація	Токенізація, видалення стоп-слів, лематизація
3	Аналіз	Статистика, візуалізація	Частоти слів, колокації, класи слів
4	Моделювання	ML/AI моделі	Класифікація тексту, тематичне моделювання, генерація
5	Валідація	Точність, F1, ROC	Точність, F1, ROC

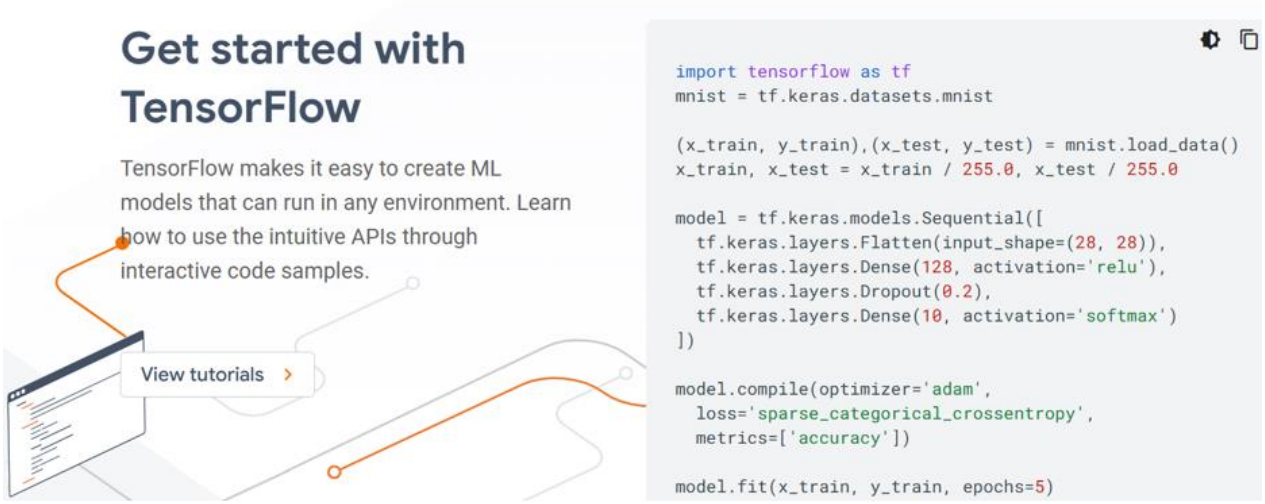

Рис. 3.17 Фреймворк TensorFlow

Keras (як частина TensorFlow) дозволяє створювати моделі з високим рівнем абстракції. Підтримка TensorFlow Text та TensorFlow Hub — модулярні NLP-компоненти (рис. 3.18).

Фреймворк TensorFlow часто використовується у промислових NLP-рішеннях, які потребують масштабування, продуктивності та розгортання.

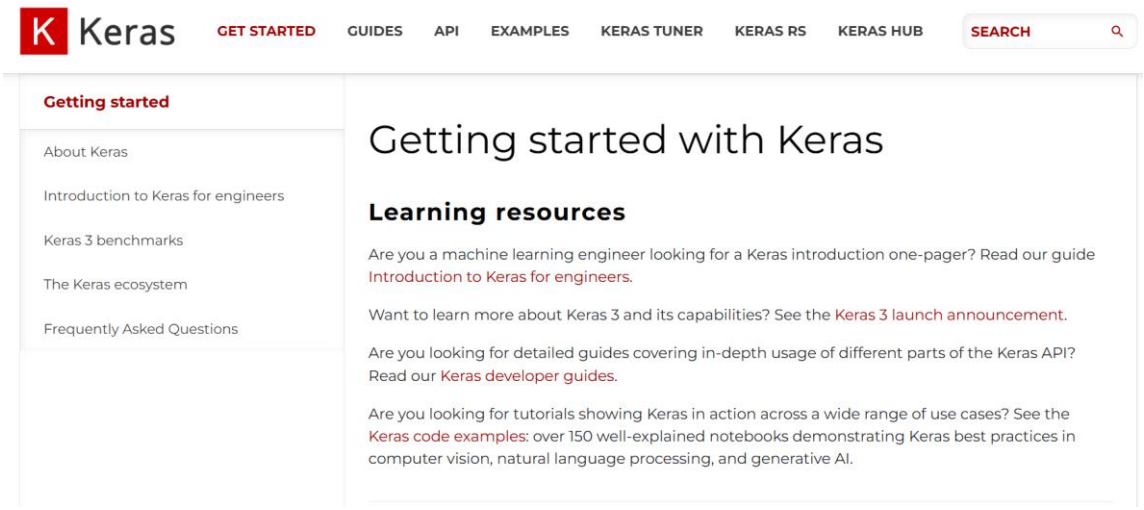

Рис. 3.18 Фреймворк Keras

PyTorch є гнучким фреймворком глибокого навчання з динамічними обчисленнями. Популярний серед дослідників через простоту експериментів із новими NLP-архітектурами. Є основною платформою для бібліотеки Hugging Face Transformers (моделі BERT, GPT, RoBERTa). Найчастіше використовується для роботи з передовими NLP-моделями, трансформерами та дослідженнями.

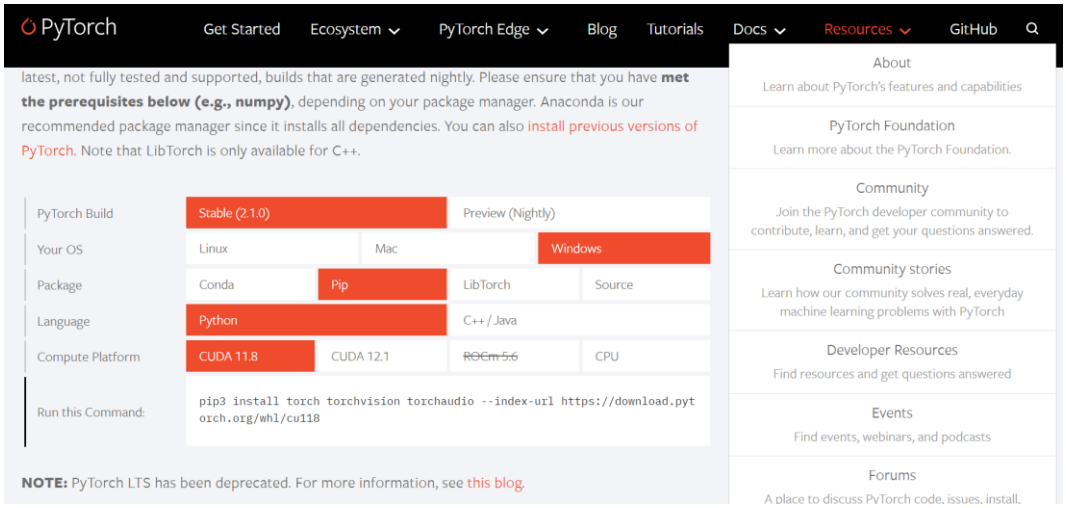

Рис. 3.19 Фреймворк PyTorch

Застосування бібліотек та фреймворків в NLP-задачах представлене у табл. 3.7.

У результаті проведеного аналізу визначені рекомендації для ефективної організації навчання та досліджень у сфері обробки природної мови (NLP).

Першою рекомендацією є вивчення основ програмування та машинного навчання. В якості основного інструменту в NLP є мова програмування Python

(табл. 3.2, 3.3, 3.4). З машинного навчання потрібно знати його типи, задачі та алгоритми до поставлених задач.

Таблиця 3.7

Застосування бібліотек та фреймворків в NLP-задачах

№ п/п	Бібліотека	Основне призначення	Рекомендоване застосування
1	Scikit-learn	Класичні ML-моделі, базовий NLP	Прості задачі класифікації, тематичне моделювання
2	TensorFlow	Глибоке навчання, масштабовані NLP-моделі	Seq2Seq, трансформери, великі проєкти
3	PyTorch	Гнучке глибоке навчання, підтримка Transformers	Сучасні трансформери, дослідження, Hugging Face

Другою рекомендацією для вивчення технології NLP в інформаційних системах є розуміння NLP-задач та методи їх вирішення: токенізація, стемінг, лематизація, POS-теги. Популярними задачами в області NLP є: класифікація тексту, аналіз тональності, генерація тексту, машинний переклад. Також потрібно вміти застосовувати інструменти, такі як NLTK, spaCy, TextBlob.

Наступним кроком є знайомство з методами векторизації тексту. Методи векторизації тексту поділяються на прості (Bag of Words (BoW), TF-IDF) та складні (Word2Vec, GloVe, трансформери (BERT, GPT)). Доцільно навчатися векторизовувати текст у Jupyter Notebook або Google Colab та побачити як виглядає текст у числах.

Важливим кроком організації навчання та досліджень у сфері обробки природної мови є проходження практики на реальних даних. Бажано починати з простих датасетів: IMDb (аналіз тональності), SMS Spam Collection (класифікація), 20 Newsgroups (тематична класифікація). Для практики доцільно використовувати платформу Kaggle.

Подальшим кроком є вивчення бібліотек глибокого навчання: TensorFlow / Keras або PyTorch - для створення власних NLP-моделей; Hugging Face Transformers - для роботи з сучасними моделями (BERT, RoBERTa, GPT та іншими).

Наступними важливими етапами для розвитку знань в NLP-галузі є проведення досліджень та аналізу. Рекомендовано вивчати, які методи краще працюють для певних задач; аналізувати ефективність векторизації, вибір оптимізатора, структури моделі; порівнювати прості та складні моделі на одних і тих самих даних.

Для саморозвитку рекомендовано стежити за новинами та дослідженнями в областях ML/DL/DS/NLP: переглядати статті з arXiv.org, переглядати публікації в Hugging Face, Google AI, DeepMind, слухати виступи з конференцій: ACL, EMNLP, NeurIPS.

Навчання NLP - це поєднання теорії, практики та досліджень. Починати рекомендовано з простого, поступово заглиблюючись у методи та проводячи експерименти з моделями, щоб стати впевненим фахівцем.

ВИСНОВКИ

Обробка природної мови (NLP) є однією з ключових, що дозволяє створювати інтелектуальні інформаційні системи, здатні розуміти, аналізувати та генерувати людську мову. Її застосування значно підвищує ефективність взаємодії користувача з програмними системами.

У ході дослідження були розглянуті основні задачі NLP (класифікація текстів, аналіз тональності, витягування сутностей) та методи їх вирішення: від простих (Bag of Words, TF-IDF) до сучасних векторних та глибинних підходів (Word2Vec, BERT). Доведено ефективність використання алгоритмів машинного навчання у вирішенні NLP-задач. Інтеграція NLP-моделей у інформаційні системи дозволяє автоматизувати обробку великих обсягів текстової інформації, зменшити навантаження на користувачів і покращити функціональність (наприклад, фільтрація контенту, підтримка діалогових інтерфейсів, персоналізація).

Для впровадження NLP у прикладні системи рекомендовано використовувати Python та такі бібліотеки як Scikit-learn, spaCy, Hugging Face Transformers. Google Colab є доцільним середовищем для прототипування й експериментів.

Перспективними є дослідження адаптивних NLP-моделей, мультимовних моделей, моделей зі зниженими обчислювальними витратами (DistilBERT, quantized models) та їх застосування у спеціалізованих галузях (медицина, право, освіта).

Технологія NLP у поєднанні з візуальними даними може аналізувати, описувати або навіть розпізнавати кольори через текст з використанням колірних модедей RGB, HSV, Cielab і такі застосування особливо корисні у дизайні, маркетингу, комп’ютерному зорі та мультимодальних моделях.

Вирішення завдань з NLP відбувається у кілька етапів: збір і підготовка даних; перетворення тексту у числове представлення; побудова та тренування моделі; оцінка моделі; тестування і вдосконалення; розгортання та використання.

ПЕРЕЛІК ПОСИЛАНЬ

1. https://www.python.org/
2. https://www.nltk.org/
3. https://spacy.io/
4. https://textblob.readthedocs.io/en/dev/
5. https://scikit-learn.org/stable/
6. https://openai.com/index/clip/
7. https://radimrehurek.com/gensim/
8. https://huggingface.co/docs/transformers/index
9. https://developer.mozilla.org/en-US/docs/Web/API/SpeechRecognition

10. Sebastian Raschka, Yuxi (Hayden) Liu, Vahid Mirjalili. Machine Learning with PyTorch and Scikit-Learn, Packt Birmingham-Mumbai, 2022. – 741p.

11. Knickerbocker D. Network Science with Python, Packt Birmingham-Mumbai, 2023. – 395p.

12. Dr. Deepali R Vora, Dr. Gresha S Bhatia. Python Machine Learning Projects, 2023. – BPB Online, 302p.

13. Nguyen, V., Khanh, T. T., Nam, P. V., Thu, N. T., Hong, C. S., Huh, E.-N. Towards flying mobile edge computing. Proc. Int. Conf. Inf. Netw. (ICOIN), Jan. 2020. — 723-725 c.

14. Zhou, F., Hu, R. Q., Li, Z., Wang, Y. Mobile edge computing in unmanned aerial vehicle networks. IEEE Wireless Commun., 27(1), 2020. — 140-146 c.

15. Mishra, D., Natalizio, E. A survey on cellular-connected UAVs: Design challenges enabling 5G/B5G innovations and experimental advancements. Comput. Netw., 182, Dec. 2020.

16. W. Yu, W. Cheng, C. Aggarwal, K. Zhang, H. Chen, and Wei Wang. NetWalk: A flexible deep embedding approach for anomaly Detection in dynamic networks, ACM KDD Conference, 2018.

17. Jake VanderPlas Python Data Science Handbook: Essential Tools for Working with Data, 2022, 551p.

18. Heller, Nicholas et al. (2020). The KiTS19 Challenge Data: 300 Kidney Tumor Cases with Clinical Context, CT Semantic Segmentations, and Surgical Outcomes. arXiv: 1904. 00445 [q-bio.QM].

19. Hollo, Kaspar (2019). Exploring the Value of Weakly-Supervised Deep Learning Approaches for Artefact Segmentation in Brightfield Microscopy Images. URL: https://comserv. cs . ut . ee / home / files / hollo _ softwareengineering _ 2021.

20. Wang, Haofan et al. (2020). Score-CAM: Score-Weighted Visual Explanations for Convo-lutional Neural Networks. arXiv: 1910.01279 [cs.CV].

21. Yang, Guanyu et al. (2020). “Weakly-supervised convolutional neural networks ofrenal tumor segmentation in abdominal CTA images”. In: BMC Medical Imaging 20.1, p. 37. ISSN: 1471-2342. DOI: 10.1186/s12880- 020- 00435- w. URL: https://doi.org/10.1186/s12880-020-00435-w.

22. Selvaraju, Ramprasaath R. et al. (2019). “Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization”. In: International Journal of ComputerVision 128.2, pp. 336–359. DOI: 10 . 1007 / s11263 - 019 - 01228 - 7. URL: https://doi.org/10.1007\%2Fs11263-019-01228-7.

23. Лосєв М.Ю. Програмування мовою Python: навчальний посібник / М. Ю. Лосєв, В. М. Федорченко. – Харків, - Львів: Видавництво ПП «Новий Світ - 2000», 2023. – 178 с.

24. Золотухіна О.А., Негоденко О.В., Резник С.Ю., Разіна С.Я. Якість та тестування інформаційних систем. Навчальний посібник підготовлено до друку для самостійної роботи студентів вищих навчальних закладів. Київ: ННІІТ ДУТ, 2020. – 128 с.

25. Зінченко О.В., Фесенко М.А., Березівський М.Ю., Кисіль Т.М. Технології смарт-систем. – Навчальний посібник. – К.: ДУІКТ, 2023. – 162 с.

26. Нікольський Ю. В., Пасічник В. В., Щербина Ю. М. Системи штучного інтелекту: навчальний посібник. – Львів: «Магнолія-2006», 2024. – 279 с.

27. M. Gagolewski. Minimalist Data Wrangling with Python. Warsaw University of Technology, Poloand. Copyright © 2024, 440p.

ДЕМОНСТРАЦІЙНІ МАТЕРІАЛИ

(Презентація)