МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Сумський державний університет
Факультет електроніки та інформаційних технологій
Кафедра комп'ютерних наук

«До захисту допущено»
В.о. завідувача кафедри

_____ Ігор ШЕЛЕХОВ
(підпис)
2024 р.

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня бакалавр

зі спеціальності 122 – Комп'ютерні науки,
освітньо- професійної програми «Інформатика»
на тему: **«ІНФОРМАЦІЙНА СИСТЕМА РОЗПІЗНАВАННЯ РЕЧЕНЬ УКРАЇНСЬКОЇ МОВИ У ПРОЦЕСІ КЛАСТЕРНОГО АНАЛІЗУ»**
здобувача групи ІН-04р, Тайирова Мехрігул Хазраткуловна

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

_____ Мехрігул ТАЙИРОВА
(підпис)

Керівник доцент,
кандидат фізико-математичних наук Сергій ШАПОВАЛОВ _____
(підпис)

Суми – 2024

Сумський державний університет
Факультет електроніки та інформаційних технологій
Кафедра комп'ютерних наук

«Затверджую»
В.о. завідувача кафедри
_____ Ігор ШЕЛЕХОВ
(підпис)

ЗАВДАННЯ НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття освітнього ступеня бакалавра

зі спеціальності 122 - Комп'ютерні науки, освітньо-професійної програми «Інформатика»
здобувача групи ІН-04р, Тайирова Мехрігул Хазраткуловна

1. Тема роботи: *«Інформаційна система розпізнавання речень української мови у процесі кластерного аналізу»*
затверджую наказом по СумДУ від *«22» квітня 2024 р. №0414-VI*
2. Термін здачі здобувачем кваліфікаційної роботи *до 09 червня 2024 року*
3. Вхідні дані до кваліфікаційної роботи _____
4. Зміст розрахунково-пояснювальної записки (перелік питань, що їх належить розробити)
1) Сучасний стан використання систем аналізу природньої мови 2) Інформаційний огляд методів обробки природньої мови. 3) Формування вхідного математичного опису. 4) Дослідження вхідних даних 5) Програмна реалізація алгоритму кластеризації для задачі NLP
5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)_____
6. Консультанти до проєкту (роботи), із значенням розділів проєкту, що стосується їх

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв

7. Дата видачі завдання «____» _____ 20 ___ р.

Завдання прийняла до виконання _____ (підпис)

Керівник _____ (підпис)

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Назва етапів кваліфікаційної роботи	Термін виконання	Примітка
1	*Сучасний стан використання систем аналізу природньої мови*		
2	*Інформаційний огляд методів обробки природньої мови*		
3	*Формування вхідного математичного опису*		
4	*Дослідження вхідних даних*		
5	*Програмна реалізація алгоритму кластеризації для задачі NLP*		

Здобувачка вищої освіти _____ (підпис)

Керівник _____ (підпис)

АНОТАЦІЯ

Записка: 43 стор., 12 рис., 1 додаток, 23 використаних джерел.

Обґрунтування актуальності теми роботи – розроблення системи кластеризації даних на базі NLP є актуальним, адже вона дозволить виконувати автоматичне категоріювання великих об’ємів даних з визначенням найбільш інформативних елементів вхідного тексту.

Об’єкт дослідження – процес підтримки прийняття рішень.

Мета роботи – розробка системи класифікації семантичних конструкцій в українській мові.

Методи дослідження – моделі та методи інтелектуального аналізу природньої мови.

Результати – комплексна система кластеризації великих масивів україномовного тексту.

NLP, ІНТЕЛЕКТУАЛЬНА СИСТЕМА, ІНФОРМАЦІЙНА ПІДТРИМКА, МАШИННЕ НАВЧАННЯ, СЕМАНТИЧНИЙ АНАЛІЗ, КЛАСТЕРНИЙ АНАЛІЗ

ЗМІСТ

ВСТУП

Актуальність теми дослідження. В сучасному інформаційному суспільстві головним способом передачі, зберігання й обробки інформації та досвіду є мова. Будь-які ідеї, предмети чи процеси описуються словами певного алфавіту. На протязі кожної хвилини світова база даних доповнюється кілобайтами нових записів. Із-за експоненціального збільшення її об'ємів, проблематика сприйняття інформації стає все більш актуальною.

Алгоритмічна обробки природньої мови посіла одне з провідних місць в інформаційно-комунікативному просторі [1-10]. Дуже часто людина навіть не помічає, що отримана нею текстова інформація, в тій чи іншій мірі, була попередньо опрацьована спеціалізованими алгоритмами. Починаючи від автоматичного виправлення орфографічних помилок чи машинного перекладу, закінчуючи автоконспектуванням цілих книг.

І починаючи з галузі теорії алгоритмів та структур даних ця проблема поступово заповнила собою галузі штучного інтелекту та машинного навчання [12-20].

Однак при роботі з природньою мовою є ряд науково-методологічних проблем, які до сих пір потребують вирішення. До них відносять:

- багатозначність семантичних конструкцій, які залежать від контексту та індивідуального досвіду автора;
- філологічна специфіка мовник груп, що породжує велике різноманіття мов; безперервне оновлення мов. Додавання до них нових правил та тлумачень;

Розбудова України, як суверенної правової держави і формування інформаційного суспільства висувають на першочерговість вирішення перерахованих вище проблем. Саме українська мова багата на семантичні конструкції та має багатозначність контекстів й підконтекстів. Тому розробка інформаційних технологій розпізнання україномовних речень та систем кластеризації наддасть україномовній інформації властивість знаннєвості та перетворить в інформаційний ресурс для використання.

Класифікація в її найширшому розумінні потрібна для розвитку мови, яка складається зі слів, які допомагають нам розпізнавати та обговорювати різні типи подій, предметів і людей, з якими ми стикаємося. Кожен іменник у мові, наприклад, по суті є ярликом, який використовується для опису класу речей, які мають вражаючі особливості [2, 3, 11].

Об’єкт дослідження. Процес підтримки прийняття рішень.

Предмет дослідження. Система класифікації семантичних конструкцій в українській мові.

Гіпотеза дослідження полягає в тому, що інформаційна технологія розпізнавання речень української мови у процесі кластерного аналізу розв’яже низку проблем, пов’язану з обробкою великих масивів україномовної інформації і дозволить встановлювати і виявляти нові знання.

Наукова новизна. Досліджено основні науково-методологічні проблеми розробки інтелектуальних компонент при розпізнанні речень українською мовою в процесі кластерного аналізу. Під час дослідження було запропоновано і реалізовано варіанти вирішення проблеми високої розмірності простору ознак NLP систем.

Структура. Кваліфікаційна бакалаврська робота складається зі вступу, трьох розділів, висновків, списку використаних джерел та додатку, викладених на 43 сторінках.

1 ІНФОРМАЦІЙНИЙ ОГЛЯД

1.1 Сучасний стан використання систем аналізу природньої мови.

Natural Language Processing (NLP) – це набір алгоритмів і методів, які дозволяють машинам аналізувати природню мову людини. На практиці сфера використання NLP дуже широка і включає в себе безліч не очевидних варіантів застосування. Тим не менш всіх їх об'єднує спільна мета – це аналіз багатозначних синтаксичних конструкцій [1-7].

На практиці, можна виділити наступні рівні реалізації систем підтримки прийняття рішень (СППР) для вирішення задач NLP:

- системи статистичного аналізу, які дозволяють, наприклад, визначати найбільш вживані слова чи словосполучення конкретного користувача. Зазвичай, статистичний огляд – це інструмент для поверхового аналізу тексту й передбачається, що далі ці результати буде розглядати деякий спеціаліст, який і буде приймати кінцеве рішення.
- аналіз природньої мови з використанням машинного навчання (ML) – цей підхід застосовується, коли необхідно наділити систему можливістю приймати рішення, хоча і в обмеженому діапазоні. Наприклад: автоматично виправляти помилки в тексті, переклад, пошук синонімів, тощо.
- системи штучного інтелекту (ШІ) – це набір алгоритмів і методів, які в комплексі дозволяють отримати ефект присутності людини. Найпопулярнішим варіантом використання ШІ в NLP – це створення чат-ботів та голосових помічників, які можуть, у деякому сенсі, імітувати текст, а отже і мову людини.

Основна проблематика NLP – це довільність умов формування вхідного тексту, бо семантика речень містить його значення, але не мету. Хоч при формуванні речень люди користуються загальноприйнятими правилами, але це стосу-

ється лише структури. Змістовна частина може бути побудована довільним чином, адже кожна людина вкладає власний нечіткий контекст у свої слова і дуже часто він відрізняється від словникового визначення.

Хоч далеко не кожен користувач сучасними технологіями про це замислюється, але ті чи інші NLP компоненти вже давно імплементовані в повсякденні девайси. Більшість людей помилково вважає, що розвиток NLP почався лише в останні декілька років, коли з'явилися чат-боти по типу ChatGPT, але насправді робота над методами аналізу людської мови велася з моменту створення перших електронно обчислювальних машин (ЕВМ). Як мінімум, спец службовці були одразу зацікавлені в розробці алгоритмів автоматичного детектування ключових слів в розмові чи переписці. Алгоритми NLP не помітно стали не від'ємною частиною сучасних девайсів та програмних додатків, тому варто зробити огляд найбільш поширених варіантів застосування цієї технології в повсякденному житті.

Один із найпоширеніших способів застосування NLP є автокорекція тексту, що є частковим варіантом реалізації алгоритму «автозаміни». Працює за рахунок узагальнення всіх надрукованих слів і пошуку найбільш близьких словникових відповідників. На даний момент ці алгоритму перебувають у стані еволюції від синтаксису до семантики, та від механіки до змісту. Передбачається, що у найближчому майбутньому будуть все більше поширені методи виправлення граматики, які враховуватимуть не тільки правопису, а й змістовну частину тексту.

Другим за популярністю варіантом використання NLP – є машинний переклад, де за допомогою ідентифікації слів, їх словникового визначення, граматики та контексту вдається виконати переклад з однієї мови на іншу. Звісно, що отриманий переклад не завжди граматично правильний, але цього більш ніж достатньо для багатьох прикладних або побутових задач.

Варто згадати й фільтрацію спаму, адже електронною поштою користуються усі. Вона потрібна для реєстрації в кожному онлайн додатку. Тим не менш, для більшості користувачів проблема спаму є не помітною, якраз із-за постійної

роботи NLP фільтрів, які за допомогою статистичної інформації здатні ідентифікувати мусорне повідомлення. При аналізі приймають участь структура листа, слова та словосполучення, які в ньому використовуються, метадані та зворотний зв'язок від інших користувачів, які отримували такий самий або ж ідентичний лист.

Також відносно не помітним варіантом використання NLP є пошукові системи, які здатні визначити головний зміст довільного запиту та видати найбільш релевантний результат, який відповідає меті пошуку. Це слабоформалізована задача ідентифікації семантичних сутностей.

Окрім вище зазначеної формалізації мети запиту варто виділити ще автоматизований пошук семантичних конструкцій у великих об'ємах даних, що також є частиною пошукових систем. Знаходження найбільш відповідної інформації згідно текстового запиту користувача.

Однак більш досконалі пошукові системи здатні враховувати й попередні запити користувача, щоб результат пошуку був більш коректним і відповідним. Цей самий принцип використовується, коли, на основі активності клієнту вебдодатку, йому рекомендується товар чи послуга, яка подібна до тої, яку він шукав раніше. Зі сторони NLP тут виконується аналіз речень з пошуком синонімічних конструкцій та шаблонів формування тексту, який характерний для конкретного користувача.

Достатньо великою сферою інтересу NLP є створення ефективних чат-ботів, які дозволяють вирішити велику кількість бізнес-задач. У найпростішому вигляді – це консультація клієнтів. Річ у тім, що більшість питань, які може задати пересічний користувач не є специфічними, а самі вони дуже часто повторюються, тому для автоматизації процесу спілкування залучають чат-ботів, які здатні ефективно проконсультувати людини з приводу найпоширеніших питань. Тим не менш, це далеко не тривіальна задача, адже хоч більшість питань є попередньо

відомими, проте їх формулювання є унікальним і залежить від конкретної людини. Тому задача ідентифікації природньої мови ускладнюється довільним формуванням вхідного тексту.

У таких випадках стають у нагоді алгоритми ідентифікації емоційного забарвлення тексту. Це корисно, коли менеджеру потрібно проаналізувати великий об'єм коментарів чи рецензій, щоб зрозуміти ступінь задоволеності клієнта. Також це дає додаткову інформацію про сильні та слабкі сторони продукту з точки зору споживача.

Цікаво, що деякі компанії інтегрують чат-боти з можливістю розпізнавання голосових повідомлень у реальному часі. Фактично, це також спосіб реалізації NLP алгоритму, де семантичні конструкції виокремлюються зі звуку, конвертуються в текст і обробляються за тим самим сценарієм, що й в звичайному чат-боті.

Можна вважати, що вершиною розробки NLP систем є створення розумних помічників, таких як: Siri, Alexa та Cortana, які в повній мірі реалізують усі можливі методи роботи з природньою мовою. Насамперед вони здатні визначати шаблоні вирази та команди в багатозначній людській мові, адже один й той самий зміст може бути сформований безліччю способами, які залежать від конкретної людини, його словникового запасу, досвіду та звичок. Для цього подібні системи використовують методи NLP максимально комплексно.

Також алгоритми NLP знайшли своє місце в глибокій текстовій аналітиці [4], де за допомогою машинного навчання текст інтерпретується шляхом подібним до того, як це робить людина. Тобто враховується не лише словникове визначення слів, а й конкретний контекст і зміст, який хоче передати автор. На практиці, це корисно, коли цінна інформація прихована за великою кількістю менш важливої. Зазвичай, спеціалістам потрібно витратити чи малу кількість часу, щоб виокремити такі слабоформалізовані ідеї з початкового тексту. У свою чергу NLP дозволяє відсікти найменш інформативні частини, що спрощує подальший аналіз.

Варто зазначити, що будь-яка сучасна соціальна мережа або крупний онлайн сервіс, у деякому сенсі, вимушений використовувати методи NLP під час своєї роботи, адже вони зберігають таку кількість текстових даних, які, при всьому бажанні, не можливо опрацювати мануальна.

Вважається, що сучасні комплексні системи NLP наділені властивістю «критичного мислення», яке проявляється при відповідях чат-ботів, коли запит до них містить інформацію, яка потребує розуміння тонкощів контексту. Наприклад, регіональні, расові, культурні чи релігійні особливості людини, адже знати синтаксичне визначення слів – це одна задача, а розуміти їх не формальне значення – інша.

Можна помітити, що сучасні тенденції розвитку NLP спрямовані на створення максимально автономної системи роботи з текстом, яка здатна розуміти контекст слів та формувати осмисленні, не шаблоні речення. Складно спрогнозувати, коли з'явиться можливість реалізувати щось подібне, адже на даний момент часу, навіть найрозвинутішим розробкам в сфері NLP, бракує точності в розумінні змісту. Насамперед, це пов'язано з тим, що речення, які сформовані людиною не завжди підпорядковуються чітким синтаксичним чи граматичним правилам, адже людська мова – це «живий» інструмент, який змінюється в залежності від свого носія. Люди, в залежності від власного унікального досвіду, можуть наділяти деякі слова мета значеннями, які ніяким чином не пов'язані з семантикою. До того ж, існують такі конструкти, як іронія, сарказм, сталі вирази і т.п., які не можливо інтерпретувати за словником.

1.2 Інформаційний огляд методів обробки природньої мови.

Обробка природньої мови – це комплексний процес, порядок дій в якому залежить від конкретних умов і вимог задачі [18]. Проектування таких систем починається з аналізу предметної області та декларування її можливих варіантів використання. Оскільки вони напряму впливають на бажаний результат роботи кінцевого продукту.

Універсального методу обробки та взаємодії з природньою мовою немає, адже самі задачі NLP відносяться до слабоформалізованих. Як мінімум, варто зазначити, що для кожної мови характерні свої особливості побудови речень, орфографічні та синтаксичні правила. Наприклад, коли людина вивчає нову для себе мову, то більшу частину часу витрачається не на поповнення словникового запасу, а на розуміння граматичних правил та їх коректного використання. Це ж відноситься і до машин.

Як вже було зазначено, одним з найпоширеніших варіантів використання NLP – є машинний переклад. Хоча ця технологія все ще розвивається, але загальна її структура залишається не зміною. На першому кроці, вхідне речення перетворюється в масив векторів, де кожен елемент відповідає одному вхідному слову. Далі формуються, так звані, вектори-думок, які представляють із себе семантичне визначення кожного вхідного елементу. Тоді, щоб отримати переклад, необхідно знайти найбільш правильний відповідник. Схематично це можна показати наступним чином, рисунок 1.1.

Рисунок 1.1. Схематичний приклад перекладу англійського виразу «the cat likes to eat pizza» в іспанське «el gato le gusta comer Pizza»

Звісно, що цей приклад (рисунок 1.1) є умовним, адже на практиці далеко не всі слова мають однозначний переклад, і вони можуть бути вільно перекладені на різні мови. Окрім цього, існують деякі фундаментальні проблеми машинного перекладу. Одна з них – це багатозначність слів.

Під час виконання перекладу, фактично, виконується задача прогнозування, адже вхідні дані є не однозначними за своєю семантикою. Їх значення залежать від контексту, а отже для їх однозначної інтерпретації необхідно знати

попередні слова в реченні. Це пов’язано із особливістю мови, яка є структурно залежною. Перше слово, визначає наступні, їх зміст та, відповідно, синтаксис.

Другим з найбільш вживаних методів NLP э автокорекція тексту. Схема використання якої дуже подібна до машинного перекладу, адже при виправленні речень необхідно знайти такий відповідник, який найбільше схожий на вхідний елемент.

Більшість з базових методів автозаміни тексту використовують метрику Левенштейна. Ідея якого полягає у підрахунку кількості виправлень, вставок, замін чи видалень, які необхідно виконати щоб перетворити одну строку в іншу. Логічно, що те речення, яке потребуватиме меншої кількості перетворень над вхідним текстом і буде правильним. На практиці метрику Левенштейна використовують наступним чином:

1) Генерація списку кандидатів: система генерує список слів-кандидатів зі словника чи бази даних, який складається зі слів, довжина яких аналогічна вхідному, що має помилку.
2) Розрахунок відстані Левенштейна між словом з помилкою та кожним у списку кандидатів. Що менше відстань Левенштейна, то більша схожість між ними.
3) Ранжування. Слова з найменшими відстанями Левенштейна мають більш високий рейтинг, оскільки вони з більшою ймовірністю будуть передбачуваною корекцією.
4) Презентація користувачу. Система представляє відсортовані пропозиції користувачу, який може вибрати найбільш відповідне виправлення.

Хоча загалом цей інструмент достатньо ефективний, але він має ряд очевидних недоліків. По-перше, не завжди можливо помітити помилку або ж виправлення, яке пропонує система, може бути некоректним. По-друге, як і будь-який NLP метод автокорекція дуже чутлива до мови. Тим не менш, автовиправлення стало не від’ємною частиною будь-яких сучасних текстових редакторів.

Одним з варіантів використання NLP, про який ще не було згадано, є перевірка на плагіат. Ці інструменти аналізують вхідний текст і порівнюють його з великою базою джерел даних, щоб виявити потенційні збіги. Це стосується як безпосереднього копіювання матеріалу, так й його перефразування без відповідного цитування.

Базуються методи визначення плагіату на глибокому аналізі вхідного тексту, який ретельно порівнюється заздалегідь сформованою базою даних, в яку входять веб-ресурси, усі можливі публікації, наукові роботи, тощо. При цьому слова розглядаються і порівнюються як самі по собі, так й у вигляді словосполучень. Знаходяться синонімічні пари. А також береться до уваги структура розглядаємого тексту.

При формуванні баз даних таких сервісів аналізується величезний об’єм тексту. Так, наприклад, систематично переглядаються довірені веб-сайти з подальшим зберіганням їх вмісту. Під час свого функціонування алгоритм перевірки на плагіат ідентифікує та помічає найбільш унікальні речення, фрази, слова чи словосполучення в кожному документі. Звісно, що це стосується лише текстової інформації, усе інше не враховується алгоритмом. А також варто пам’ятати, що при перевірці на плагіат можливо використати лише матеріали з відкритих джерел.

Цікавим, але далеко не популярним, способом використання NLP є автоматичне узагальнення тексту, яке полягає у значному скороченні вхідної інформації шляхом виділення та поєднанні найбільш значущих елементів. Розділяють два принципово різних підходи при узагальнені текстової інформації:

Перший – це екстракція або ж конспектування, коли алгоритм оцінює усі речення вхідного документу і обирає найбільш інформативні з них. Звісно, що у результаті ми можемо отримати не найкращим чином структурований текст, який більше буде нагадувати набір тез, але які здатні передати головну ідею. Базовий алгоритм автоматизованого конспектування складається з наступних кроків:

1) Структурне представлення вхідного документу, де показується взаємозалежність між реченнями або ж словами. Це необхідно для ідентифікації найбільш значущих, для змісту, елементів. Зазвичай для цього достатньо використати частотний аналіз (TF).
2) Паралельно оцінюється з якою вірогідністю той чи інший елемент вхідного елементу з'явиться у вихідному конспекті, адже багато речень, не зважаючи на свою інформаційну цінність, можуть нести один і той самий зміст. У деякому сенсі, бути взаємозамінними.
3) Створення конспекту на основі усіх найбільш інформативних елементів вхідного документу. При деяких реалізаціях на цьому етапі використовується латентний семантичний аналіз (LSA) для виявлення семантично важливих конструкцій.

Другий підхід для узагальнення тексту – це його абстрагування. Результатом цього процесу є вихідний конспект, який може містити в собі абсолютно нові речення, яких не було в початковому документі. Тобто створюється абсолютно новий текст, але який передає головну ідею вхідного.

Загалом, задача абстрагування є набагато складнішою і комплексною ніж екстракція. Як мінімум, абстрактні речення повинні складати не перервний, логічно зв'язаний текст. Хоча на практиці цього до сих пір не вдалося досягти в повній мірі й створені у такий спосіб речення все ще якісно відрізняються від написаних людиною.

Простим прикладом абстрагування – є метод нейронної уваги [6], який широко використовується в структурно орієнтованих моделях послідовностей, де декодер вилучає інформацію з кодеру на основі оцінки уваги на елементах документу. У деякому сенсі це відповідність слів, речень, словосполучень між собою. Тобто наскільки один вираз добре передасть ідею іншого. Для цього будують відповідні матриці уваги. Так на рисунку 1.2. можна побачити візуальний приклад теплової карти з якою вірогідністю одне слово може перетворитися в інше в абстрагованому тексті.

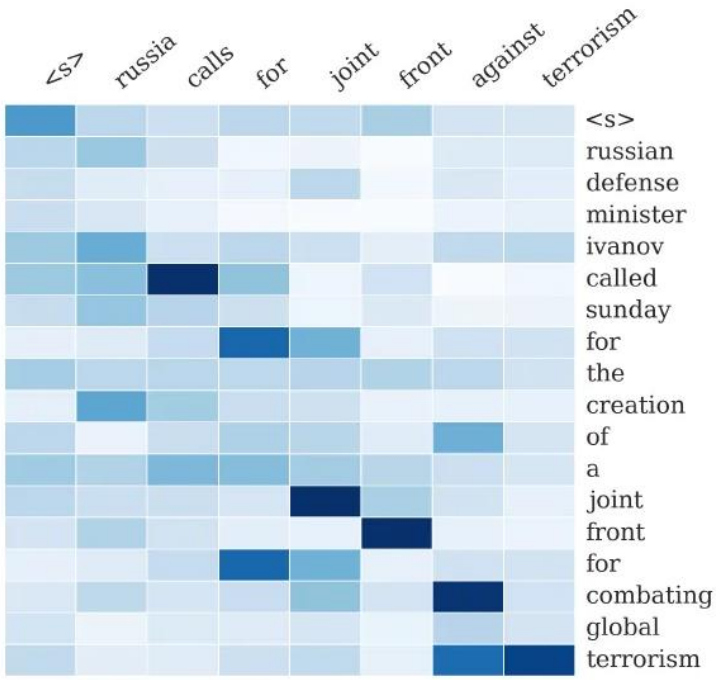

Рисунок 1.2. Приклад абстрагування тексту [6] через метод уваги

Хоч технологія автоматизованого конспектування не настільки поширена, але це неймовірно зручний інструмент, коли потрібно опрацьовувати велику кількість джерел інформації. Не зважаючи на очевидні недоліки, наприклад, відсутність логічної структури у вихідному тексті або ж втрата деяких змістовно важливих частин вхідного документу. Проте все рівно цей інструмент здатний заощадити значну кількість часу при опрацюванні великої кількості подібної між собою інформації., шляхом виокремлення головних ідей.

Окремо варто згадати розпізнавання голосу. Це достатньо специфічна сфера використання NLP, насамперед, із-за того, що в ній вхідні дані отримуються аналоговим шляхом через аналого-цифрові перетворювачі. Робиться це наступним чином:

1) Голос, у вигляді звукової хвилі, записується та конвертується в математичний об’єкт, ознаками якого є амплітуда, яка вимірюються в децебелах (дБ).
2) Неперервний запис розбивається на блоки фіксованого розміру, які містять в собі амплітуди початкового сигналу. Кожен з блоків індексується, що дозволяє конвертувати аналоговий звук в цифровий.

3) Останнім кроком виокремлюється додаткова інформація для кожного з отриманих блоків. Зазвичай цікавить: частота, інтенсивність і час звуку. На практиці, це можна зробити за допомогою швидкого перетворення Фурье, результатом якого є спектрограма звукового сигналу (рис.1.3)

Приклад спектрограми можна побачити на рисунку 1.3. По вертикалі знаходиться частота сигналу, а по горизонталі - час. Колір відповідає за інтенсивність, яка була витрачена на продукування звуку.

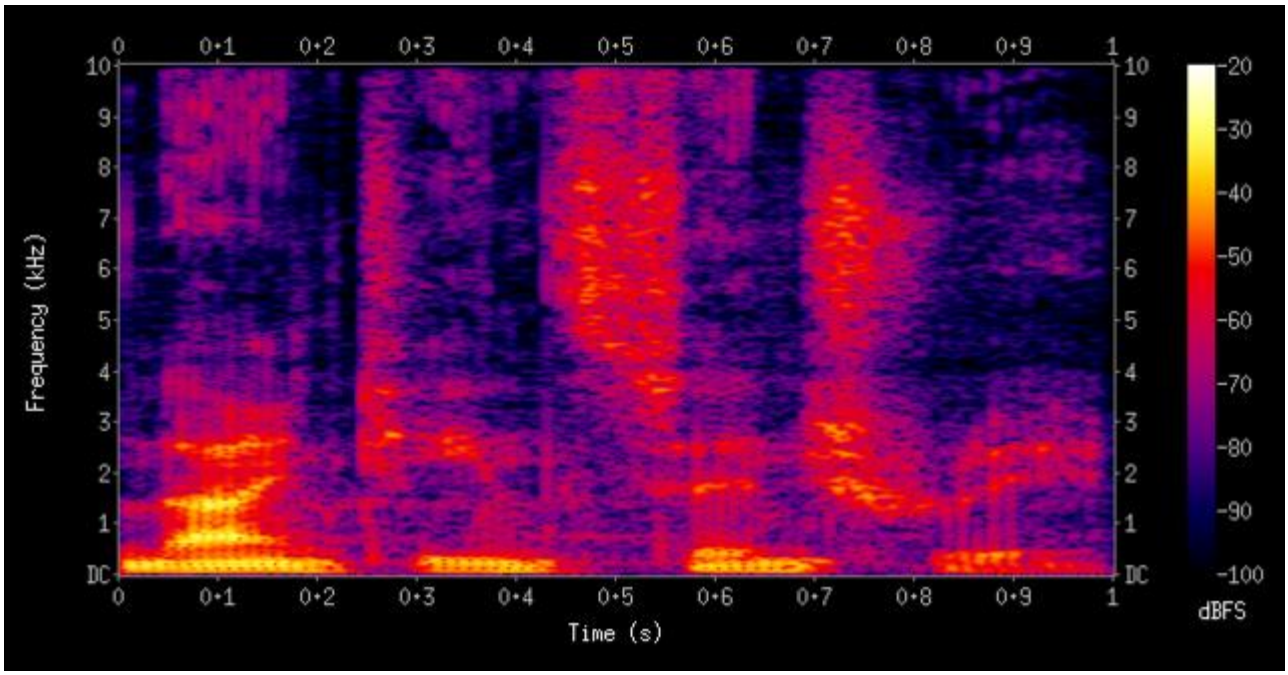

Рисунок 1.3. Приклад спектрограми голосу людини

Для розпізнавання голосу використовується метаінформація про мову, наприклад, фонеми, які представляють із себе окремі звукові елементи слова. У інформаційному еквіваленті, фонеми – це ознаки слів, які використовує алгоритм для їх ідентифікації. Хоча, на практиці, все не так однозначно, адже люди в залежності від акценту, віку чи емоційного стану можуть видозмінювати свою вимову, а отже і фонеми. Лінгвісти називають такі варіаційні ряди аллофонами.

Базовим методом розпізнавання голосу вважається прихована марковська модель (ПММ). Це багатошарова структура, яка розташовує ідентифіковані фонеми у правильному порядку, який враховує їх статистичні вірогідності. Для

цього в моделі передбачено три шари. На першому перевіряється акустичний рівень сигналу, щоб підібрати найбільш відповідні варіації фонем. Другий рівень співставляє ідентифіковані на попередньому кроці фонеми з подальшою перевіркою можливості їх поєднання за семантичними правилами конкретної мови. Адже деякі звуки фізично не можуть поєднуватися, як мінімум, із-за того, що обрана мова не містить слів з такими комбінаціями фонем. Третій рівень відповідає за структурну правильність вихідного речення, шляхом оцінки змістовності отриманих слів. Наприклад, через перевірку кількості дієслів, іменників і т.д. Тобто наскільки збалансованим вийшов текст.

У такому ітеративному процесі система постійно перевіряє та пере провіряє усі можливі вірогідності складу вихідного речення, щоб знайти найбільш відповідний варіант вимовленого людиною тексту. Це дуже помітно, коли використовуєш голосовий друк. Поки диктуєш речення система може змінити вже надруковані слова.

Найголовніша проблема при розпізнаванні звуку – це неоднорідність фонем, на вимову яких впливає індивідуальний фактор кожної окремої людини. Це усугубляється тим фактом, що додавання нових варіацій вимови збільшує вірогідність можливих довільних комбінацій, які на практиці складно заздалегідь врахувати. Тим не менш, ця технологія вже давно активно використовується і з кожним роком розвивається.

Скоріш за все, найбільш показовим і простим варіантом поєднання машинного навчання (ML) з NLP є класифікація тексту. По своїй суті, вона подібна до аналогічних алгоритмів ML, що означає можливість використовувати загально доступні та попередньо реалізовані, у відповідних фреймворках, алгоритми інтелектуального аналізу. У спрощеному вигляді це виглядає наступним чином: вхідний математичний опис документу ідентифікується та відноситься до одного з наявних класів розпізнавання.

Текст документу представляється у векторизованому вигляді, кожен елемент якого зберігає інформацію про слово, його частоту і положення в документі.

Головна мета – це знайти найбільш відповідний клас у просторі ознак, користуючись заданими метриками відстані, рисунок 1.4.

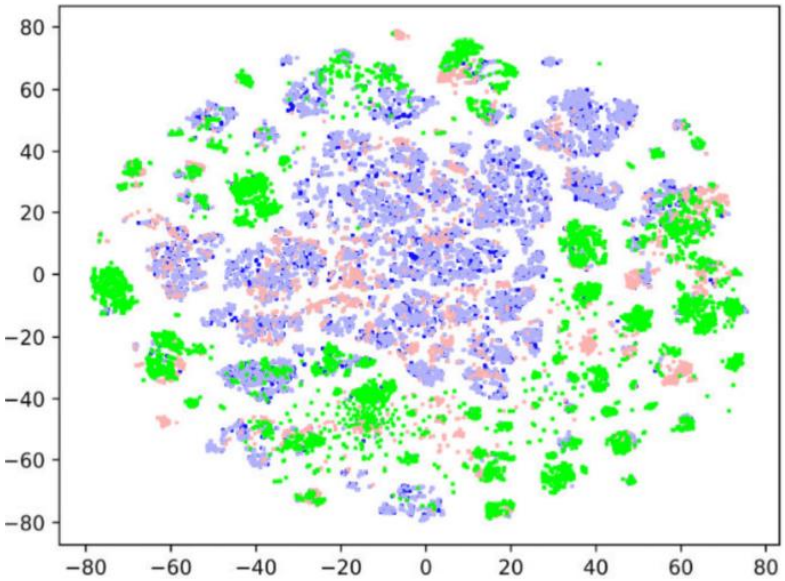

Рисунок 1.4. Приклад візуалізації простору ознак для текстового документу

Класифікація текстових масивів – це один із варіантів використання NLP, який не буде застосовувати пересічний користувач, але це дуже корисно менеджерам структур, де потрібно категоріювати великі об’єми текстових даних. Наприклад, швидко розподілити статті між категоріями або ж верифікувати їх на відповідність певній тематиці.

Окремо варто згадати методи розпізнавання та виявлення сутностей в тексті (NER). Оскільки ця група методів дуже часто є складовою більш комплексних систем, які вже було розглянуто в цій роботі. NER дозволяє виокремити частини початкового документу, які характерні сутностям конкретного класу, наприклад: імена людей, міста, організації і т.д. Це корисно при пошуку інформації, коли потрібно знайти абстрактні ключи в різних документах.

Навчання NER виконується на попередньо розміченому тексті, щоб система змогла адаптуватися під конкретний контекст мови. Наприклад, коли і як

вживаються назви міст. Такий підхід називається: граф знань. І полягає він у формуванні семантичної структури між сутностями у реченні, та їх впливу на зміст тексту, рисунок 1.5.

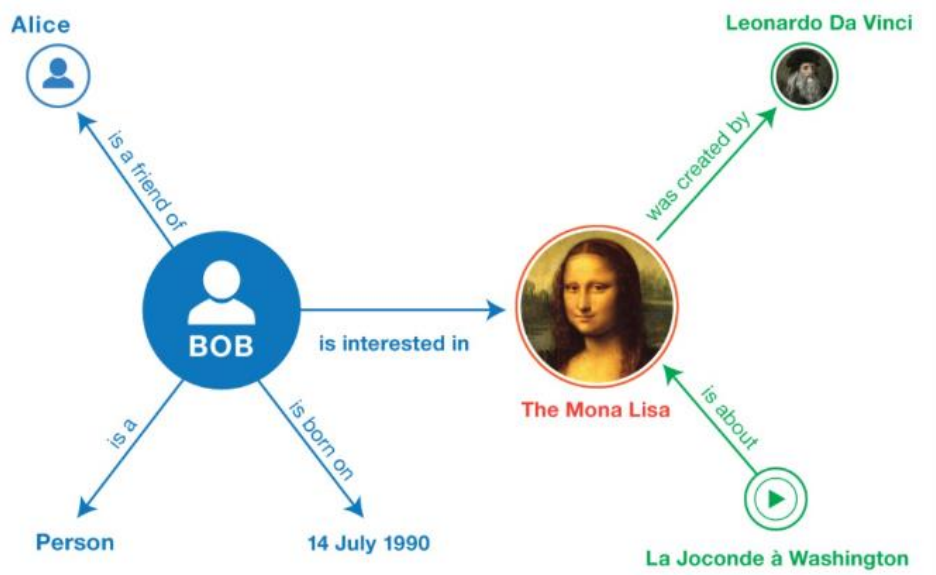

Рисунок 1.5. Приклад семантичного графу знань

Хоч самостійно NER майже не використовується, але при цьому ці алгоритми є невід'ємною частиною більш комплексних NLP систем. Насамперед, як інструмент структуризації та, у деякому сенсі, нормалізації вхідного тексту. Тим не менш, найбільш помітним прикладом застосування NER є пошукові системи, які здатні знайти потрібний об'єкт лише по нечіткому його описі.

Отже, сучасні методи обробки природньої мови – це максимально комплексні системи, які створюються при використанні статистичного аналізу, машинного навчання та метаінформації про особливості конкретної мови. У свою чергу, головною технологічною перешкодою при реалізації NLP застосунків є не однорідність тексту. Оскільки мова, як кажуть лінгвісти – це живий інструмент. Людина наділяє свої слова унікальним змістом, який не завжди можливо однозначно інтерпретувати. Тим не менш, методи NLP є надзвичайно розповсюдженими і їх можна зустріти майже в кожному програмному застосунку або девайсі.

1.3 Постановка задачі

На базі представлених джерел інформації [1-23] та аналізу існуючих рішень в задачах розпізнання семантичних конструкцій в різних мовах та різноманітних застосуваннях сформовано наступну постановку задачі.

В кваліфікаційній роботі розробити систему автоматизованої класифікації масиву речень української мови. Для вирішення поставленої задачі пропонується адаптувати метод кластерного аналізу під вимоги NLP. Це дозволить наділити систему додатковою гнучкістю та, як результат, адаптивністю під довільні умови вхідного математичного опису.

Для виконання прийнятої задачі необхідно виконати наступні завдання:

- ✓ Сформувати датасету тексту, який відповідає вимогам задачі;
- ✓ Дослідити вплив нормалізації вхідного математичного опису, для забезпечення повної ймовірності прийняття правильних класифікаційних рішень під час функціонування системи;
- ✓ Адаптувати метод кластерного аналізу до задач NLP;
- ✓ Розробити перелік евристичних правил нормалізації, які враховують специфіку вхідного навчального тексту.

2 ОПИС МЕТОДУ ДОСЛІДЖЕННЯ

2.1 Опис основних алгоритмів початкової обробки тексту

Хоча методи NLP пристосовані для виконання великої кількості задач, проте існує декілька основних алгоритмів обробки природньої мови, які можна зустріти майже в кожній реалізації NLP. Насамперед – це нормалізація тексту, приведення вхідних даних до деякого уніфікованого вигляду. Як зазначалося раніше, словам притаманна видозмінна, наприклад, за відмінками, часом і т.д. Тобто одне і те саме слово, без зміни свого змісту, може приймати різні форми. Це характерно, не лише для NLP, майже завжди при розробці будь-якої інтелектуальної системи перший крок – це дослідження і, за можливості, нормалізація вхідних даних.

При роботі з природньою мовою існує два підходи нормалізації тексту. Найбільш простий з них – це стемінг. Цей метод скорочує слова шляхом видалення закінчень, префіксів та суфіксів, залишаючи лише корінь слова. Програмно реалізувати примітивний алгоритм стемінгу достатньо легко. Необхідно лише знати базові правила словобудови конкретної мови, щоб сформувати евристичний список правил. Тим не менш, стемінг не завжди вірно скорочує слова з синтаксичної точки зору. Можливі випадки, коли різні за змістом слова зводяться до однієї нормалізованої форми, що може вплинути на результат навчання NLP системи. Тому використання стеммінгу обмежене задачами, де не потрібний детальний морфологічний зміст тексту. Наприклад, задачі класифікації документів.

Другий спосіб нормалізації речень – це лематизація. Цей підхід потребує попередньо навченних інтелектуальних компонент, які здатні повертати будь-які вхідні слова в їх початкову форму. Це набагато комплексний процес, який займає чи малу кількість часу. На практиці лематизація використовується, коли необхідно наділити майбутню NLP систему розумінням морфології тексту. Наприклад, в чат-ботах, де алгоритм повинен правильно зрозуміти питання користувача.

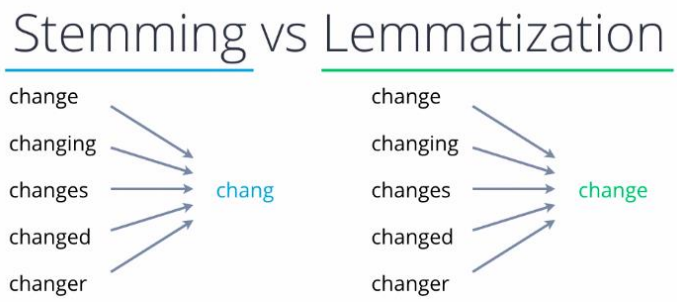

Рисунок 2.1. Візуальне порівняння стеммінгу та лемматизації

Від правильно проведеної нормалізації залежить якість кінцевої моделі, проте у рамках NLP ця задача ускладнюється деякими факторами. Головний з яких – це специфічна структура мови. Різні мовні групи відрізняються морфологічними, семантичними та синтаксичними правилами. При роботі з європейськими мовами, зазвичай, не виникає ніяких проблем, адже вони всі належать до однієї мовної групи. Тим не менш, до сих пір не існує ефективного методу нормалізації арабського тексту. Із-за наявності в ньому аглютинативної морфології, орфографічних варіацій та лексичної двозначності.

Другою невід'ємною частиною NLP алгоритмів є токенізація, де токен – це найменша частина тексту, його ознака, у рамках NLP. Для деяких алгоритмів токеном виступають слова, для інших словосполучення або цілі речення. У деякому сенсі, токенізація є нормалізацією структури речень. Шляхом видалення, так званих, стоп-слів з подальшим розбиттям тексту на найменші, неділимі елементи.

Останнім, але не менш відповідальним етапом, є векторизація вхідного тексту. Річ у тім, що будь-який алгоритм не здатний працювати з синтаксисом на пряму. Необхідно перетворити слова у набір числових векторів. Одним з найпопулярніших способів це зробити є метод мішка слів (МС) [11], який полягає у представленні тексту через відповідність токенів частоті їх появи у документі. Ідея в тому, що будь-яка інформація про порядок або структуру слів не враховується. Тим не менш, наприклад, при порівнянні текстів, це і не потрібно, адже два документи схожі між собою, коли у них подібний зміст. Грубо кажучи, коли в них зустрічаються однакові слова. Однак якщо потрібен точний морфологічний зміст,

то тоді необхідно змінити класичний алгоритм МС або ж використати інші, більш досконалі методи.

У будь-якому разі, при векторизації буде застосована семантична статистика документу. І на цьому етапі варто розуміти, що в людській мові можуть домінувати деякі слова, які не впливають на зміст речення. Для врахування цього аспекту використовують статистичну метрику TF-IDF [10], де TF – це частота токену в конкретному документі, а IDF – це частота токену в усх інших документах. У результаті отримуємо зважену оцінку, яка враховує інформативність, а отже і цінність кожного токену, адже показник IDF для унікальних слів в документів буде високим, а часто вживаних - низьким.

У загальному випадку, токенізація, нормалізація і векторизація – це ті етапи роботи з текстом, які є спільними для будь-якої NLP системи. Проте варто враховувати особливості мови, з якою буде працювати кінцевий алгоритм. Адже в залежності від її вибору можуть бути додані додаткові квантори обробки природньої мови, які базуватимуться на її специфіці.

2.2 Огляд методів візуального аналізу тексту

Найбільш ефективний спосіб дослідження масивів даних, в тому числі й текстових, є їх візуальний аналіз. У багатьох випадках такий підхід дозволяє сформувати або скорегувати вже існуючу стратегію обробки та подальшої роботи з наявними даними.

Відносно простим способом аналізу є візуалізація статистичної інформації про текст. Наприклад, за допомогою хмари слів, де найбільш вживані слова займають більше місця у заданому просторі. У деяких випадках, це дозволяє виявити не очевидні стоп-слова, які часто зустрічаються в тексті, але при цьому не несуть корисну інформацію. Наприклад, слова-паразити, прислівники і т.д., які характерні для конкретного тексту. На рисунку 2.2 показаний більш класичний графік частоти вживання окремих слів, що дозволяє візуально оцінити вміст до-

кументів. У даному випадку, перед векторизацією цього тексту, потрібно провести додаткову нормалізацію структури, щоб видалити сполучники та прийменники, які на практиці не впливають на зміст, а лише прикрашають речення.

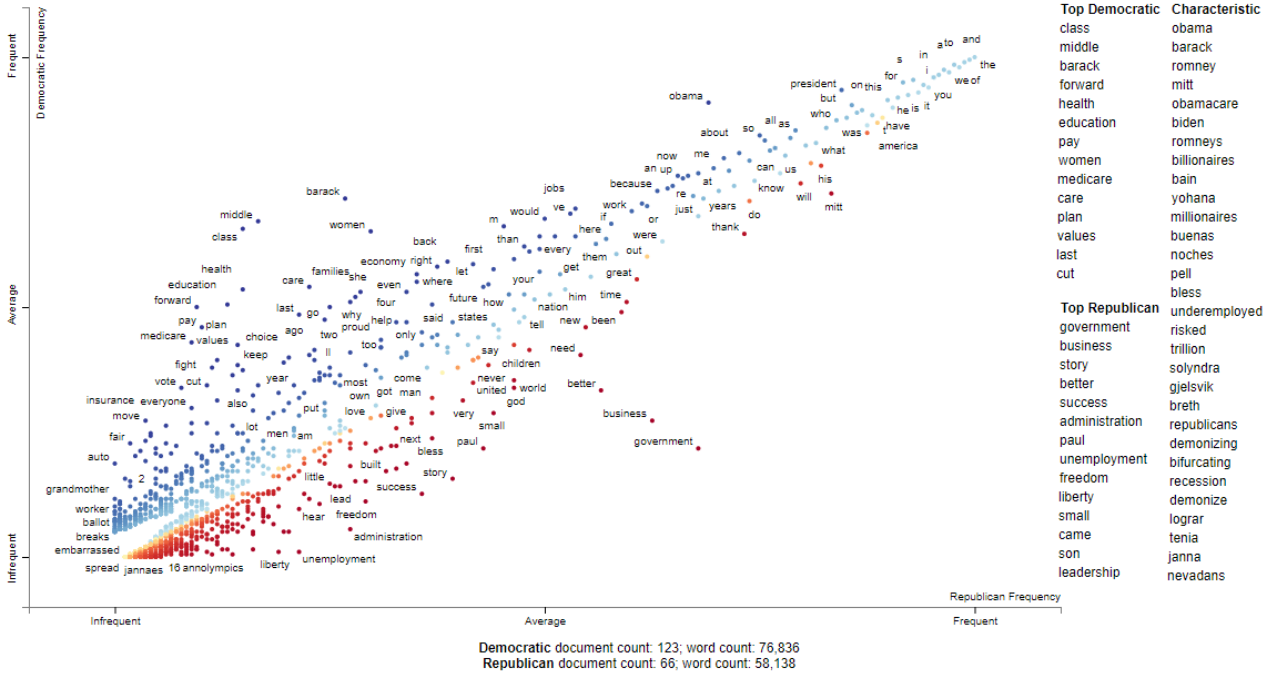

Рисунок 2.2. Приклад візуальної синтаксичної статистики

Також варто згадати про графи знань, які візуально демонструють взаємозалежність між словами в реченні та їх вплив на зміст тексту. У такий спосіб можливо виявити найбільш інформативні синтаксичні конструкції, які варто виділити для подальшого навчання NLP системи. Проаналізувати як вони впливають на зміст, шляхом їх видалення або видозміни. Окремо можна дослідити вплив речень на загальний зміст тексту.

Додаткової уваги заслуговують способи візуалізації векторизованих речень у просторі ознак. Адже це дозволяє візуально оцінити і припустити кількість класів, їх схожість між собою та найбільш інформативні ознаки. У свою чергу, цю інформацію можна використати при проектуванні математичної моделі, для досягнення найкращих результатів. А також візуально дослідити різницю між методами початкової обробки тексту.

Отже, хоч візуальний аналіз не є безпосередньою частиною машинного навчання, тим не менш, він відіграє головну роль при формуванні стратегії розробки математичної моделі, що на пряму впливає на ефективність кінцевої системи.

2.3 Огляд алгоритму кластерного аналізу для задач NLP

Методи кластеризації необхідні для групування великих об’ємів даних на задану кількість класів. При цьому, не потрібно формувати попередню навчальну вибірку, алгоритм самостійно визначає закономірності між даними та виокремлює їх в окремі каталоги шляхом створення вирішальних правил. У межах NLP, речення або документи об’єднуються за принципом подібності тем в них.

Для початку варто визначитися з рівнем кластеризації, який на пряму залежить від вхідних даних. На практиці, кластерний аналіз NLP розділяється на наступні рівні абстрагування:

- **Рівень документів**. Робота з усім вхідним текстом і класифікація його відповідно до загальних тем. Наприклад, визначення змісту статті або листа.
- **Рівень речень**. Розбиття початкового тексту на окремі речення і їх ідентифікація. Наприклад, визначення настрою людини по кожному її повідомленню.
- **Рівень слів.** Групування схожих слів. Використовується для знаходження синонімів в текстів, які мають однаковий зміст.

У загальному випадку для кластеризації будь-якого рівня спочатку потрібно виконати попередню обробку вхідного тексту, що включає в себе його нормалізацію, як окремих слів, так й структури речень шляхом видалення стоп-слів. Наступним кроком йде векторизація отриманого тексту з виокремленням інформативних ознак, наприклад, за допомогою методу TF-IDF.

Безпосередній алгоритм кластерного аналізу використовує дистанційну метрику між векторами-ознаками для знаходження найбільших конгломератів у відповідному просторі ознак. Логічно припустити, що речення, які потрапляють

у такі скупчення, будуть належати одному класу. Тобто об’єднані однією спільною темою.

Слід виділити основні науково-методологічні проблеми кластеризації тексту:

- **Висока розмірність.** Текстові дані, зазвичай, мають великі розміри, що призводить до створення векторів-ознак великої розмірності. У свою чергу, це впливає на розрахунок відстані між реченнями та ускладнює пошук кластерів;
- **Гетерогенність.** Природня мова може бути не нормованою і складатися з не академічних слів, наприклад, сленгу. Із-за цього такі слова можуть помилково вважатися системою інформативними, адже, у деякому сенсі, вони є унікальними.
- **Масштабування.** При збільшенні об’єму даних, збільшується і простір ознак, що значно ускладнює розрахунок кластерів. При цьому, мінімізувати або позбутися масштабування не можливо, адже природня мова не детермінований об’єкт. Для подолання цієї проблеми пропонується використовувати методи паралельних або розподілених обчислень.
- **Наявність зашумлених даних.** Текст може мати орфографічні чи граматичні помилки або зберігати не актуальну інформацію про тему, що вплине на результат кластеризації. У таких випадках, необхідно створювати додатковий квантор попередньої обробки тексту.

У поточному дослідженні пропонується використати ітеративний алгоритм кластеризації K-середніх (рис.2.3), який намагається згрупувати вектори-ознак в заздалегідь визначену кількість k класів. Цей метод використовує векторне квантування для мінімізації відстані між кожним вектором і центром майбутнього кластеру. Таким чином, у ітеративному процесі, відбувається пошук найкращого набору центроїдів в багатомірному просторі ознак, де сума квадратів відстаней між векторами сусідами буде мінімальна.

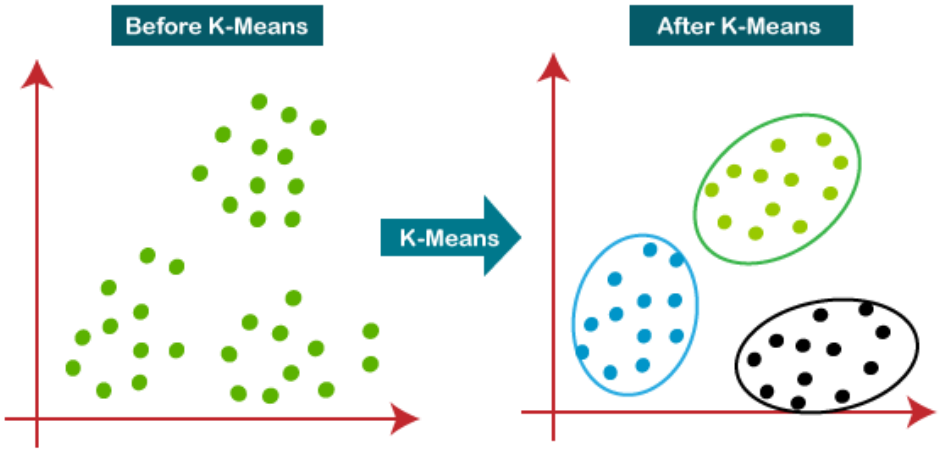

Рисунок 2.3. Візуалізація кластеризації за допомогою K-means

Найкращий спосіб використання методів кластерного аналізу в NLP задачах – це кластеризація тексту отриманого з одного джерела, коли вже попередньо відомо, що ці речення у будь-якому разі повинні розбиватися на n кількість категорій. Наприклад, відгуки про товар. Очевидно, що вони можуть бути негативні, позитивні або нейтральні. У такому разі, замість мануального перегляду усього об’єму даних достатньо використати кластер аналіз з пошуком трьох класів. Це значно пришвидшить та оптимізує роботу з текстом.

3 ПРОГРАМНА РЕАЛІЗАЦІЯ

3.1 Дослідження вхідних даних

Під час поточного дослідження розглядався датасет, який був сформований з сайту новин. Таким чином були обрані статті, які знаходилися в розділах: культура, технології, спорт. Припускається, що їх текст, хоч і буде подібним, проте в просторі ознак вдасться автоматично сформувати чіткі n-мірні кластери. Звісно, що під час машинного навчання система не буде заздалегідь знати, які документи до якої з заданих тем відносяться. Вона повинна самостійно знайти відмінності й виокремити три класи.

Для початку проаналізуємо створений датасет і сформуємо набір неінформативних стоп-слів. Для цього виведемо синтаксичну статистику документів (рис.3.1), щоб подивитися які слова є найбільш вживаними. На практиці, це повинні бути сполучники і прийменники.

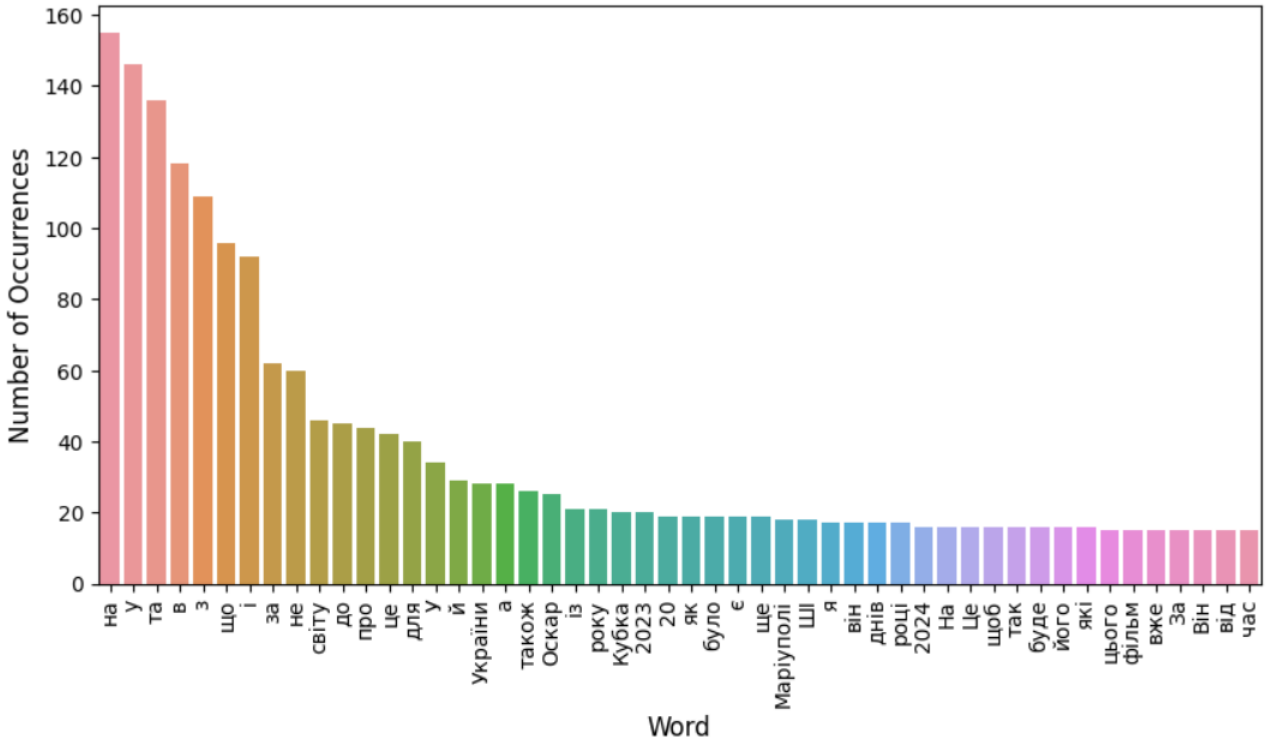

Рисунок 3.1. Семантична статистика навчального датасету

Згідно отриманого на рисунку 3.1 результату до списку найбільш вживаних неінформативних слів входять: на, у, та, в, з, що, і, за, не, до, про, це, у, й, а, також,

із, як, було, є, ще, він, щоб, так, буде, які, цього, вже, від. Ці слова варто буде видалити на етапі нормалізації речень, адже вони можуть погіршити результат машинного навчання. Також це допоможе зменшити розмірність кінцевого простору ознак.

Під час попереднього аналізу датасету було виявлено, що в статтях дуже велика кількість власних назв, які хоч і можуть бути унікальними для кожної категорії, але не характеризувати її. Більше того, без знання контексту, власні назви та імена людей можуть погіршити кінцевий результат машинного навчання. Адже система буде, наприклад, співвідносити імена конкретних людей з однією з категорій «Культура», «Наука» чи «Спорт». Таким чином, пропонується видалити власні назви.

На при великий жаль, поки що не існує окремого NER фреймворку для української мови, який би був здатний видалити власні назви та імена людей з тексту. Для вирішення цієї проблеми можна використати просте правило, якому слідує більшість алгоритмів машинного перекладу: «Власні імена не переводяться дослівно». Тому достатньо перевести датасет на англійську мову, і вже в ньому виокремити власні імена з подальшим їх видаленням. На рисунку 3.2, за допомогою візуального методу хмари слів, показані усі слова, які вважаються алгоритмом назвами компаній, заходів чи іменами людей.

Рисунок 3.2. Візуальне представлення власних назв в датасеті

Хоч отриманий результат, рисунок 3.2., не є ідеальним, бо до списку власних імен потрапили й інші іменники. Проте, видалення цих слів принесе більше

користі для подальшої інтелектуальної компоненти, але це тільки із-за специфіки завдання, де потрібно розділити текст на абстрактні теми. Якщо необхідний був детальний змістовний аналіз з виокремленням ідеї кожного окремого речення, то видалення власних імен стало б грубою помилкою.

Останнім кроком при дослідженні датасету є візуальна перевірка того, наскільки добре дані розділені в просторі ознак, рисунок 3.3.

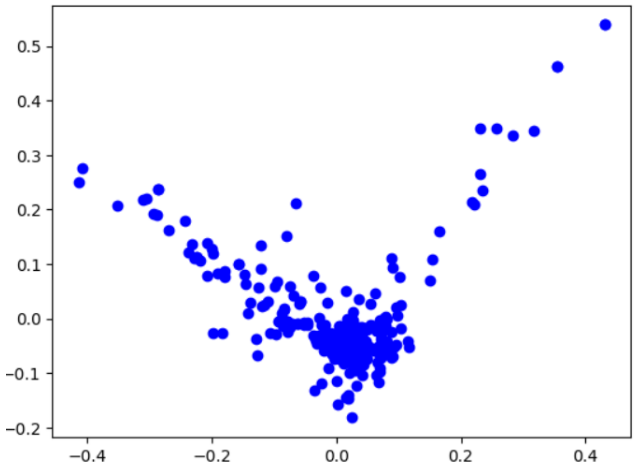

Рисунок 3.3. Простір ознак вхідних даних

Перед аналізом рисунку 3.3. варто зазначити, що простір ознак в NLP задачах є n-мірним, де n – це кількість унікальних слів чи словосполучень. Тим не менш, для простоти візуального сприйняття можна використати методи ортогонального перетворення, наприклад головних компонент (PCA), щоб зменшити розмірність векторів-ознак і розмістити їх на звичайному графіку.

Таким чином, на рисунку 3.3. можна розгледіти, як мінімум, два чітко виражених класи і третій, який вочевидь знаходиться на їх перетині. Необхідно зазначити, що це лише двовимірна проекція n-мірного простору ознак, тому в повній мірі оцінити розміщення майбутніх кластерів на рисунку 3.3. не можливо. Однак того факту, що можливо візуально розділити дані на три абстрактні групи цілком достатньо для поточного етапу, адже це означає, що процес нормалізації структури речень, шляхом видалення не інформативних конструкцій, дав позитивний результат.

Дослідження вхідних даних є невід'ємною частиною розробки будь-якої інтелектуальної компоненти, адже на цьому етапі формується майбутня стратегія роботи, яка повинна включати в себе унікальні особливості конкретного об'єкту інтересу. У нашому випадку, це знаходження специфічних стоп-слів датасету. Видалення яких дозволило зменшити масштаб простору ознак і розмірність відповідних векторів, що повинно покращити результат машинного навчання.

3.2 Реалізація алгоритму кластеризації для задачі NLP

Вхідний датасет складається з 54 870 слів, які утворюють речення, що описують одну з заданих тем: «Культура», «Наука» і «Спорт». На поточному етапі було зменшено розмірність початкового простору ознак шляхом видалення не інформативних елементів. Тепер необхідно нормалізувати вхідний математичний опис та перетворити його в вектори-ознак для подальшого навчання інтелектуальної компоненти.

Як метод нормалізації тексту був обраний алгоритм стемінгу описаний в роботі [7], де автори дослідили морфологічну структуру українських слів та правила їх видозміни. Таким чином вдалося привести усі вхідні слова до одного структурного вигляду.

Як метод кластерного аналізу був обраний алгоритм k-means. Візуальний результат кластеризації показаний на рисунку 3.4.

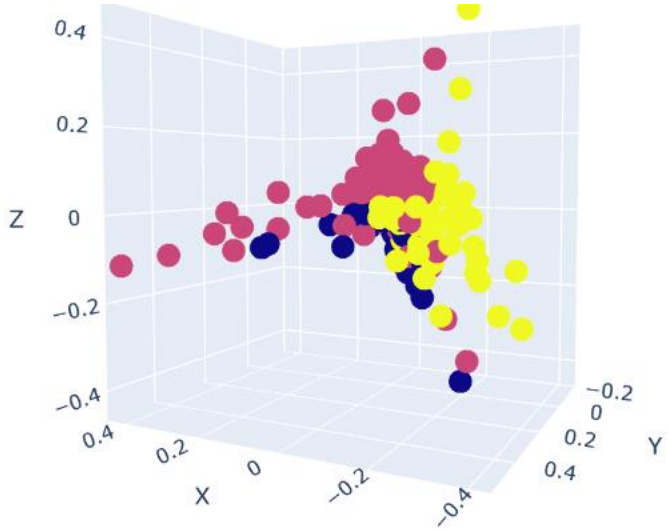

Рисунок 3.4. Візуалізація результату кластеризація

Оскільки на етапі формування датасету речення були розміченні, то зараз можливо перевірити точність результату кластеризації шляхом співставлення отриманих класів з істинними. Таким чином, була отримана точність в 87.3%, чого цілком достатньо для більшості прикладних задач. Цікавим є те, що найбільше помилок система припускала у реченнях, які не можливо однозначно віднести до жодного з наявних класів. Оскільки вони не містили ні якої специфічної інформації про клас розпізнавання. Пропонується у подальших реалізаціях додати, так звану, проміжну категорію, яка б складалася з беззмістовних речень загального характеру.

Згідно рисунку 3.4. класи вхідного датасету формують перетин, який виглядає на графіку як скупчення речень. При аналізі яких було виявлено, що в цей перетин потрапляють речення, які містять мінімальну кількість ознак кожного з класів розпізнавання. Для подолання цієї проблеми можна спробувати імплементувати додатковий квантор перевірки інформативності речень, який би відсіював текст, який не можливо точно віднести до жодної з наявних категорій.

3.3 Короткий опис програмної реалізації

Для реалізації програмного застосунку було використано мову програмування Python. Вибір обумовлений тим фактом, що на ній реалізовано велика кількість фреймворків пов’язаних з машинним навчанням та алгоритмів обробки природньої мови.

При розробці застосунку було використано декілька програмних особливостей пов’язаних з морфологією та семантикою. Зокрема, для нормалізації тексту було сформовано припущення про не інформативність власних назв у тексті, адже вони не вказують на приналежність документу до одного з класів розпізнавання. Для їх видалення було використано бібліотеку обробки природньої мови spaCy.

```
for i, sentence in enumerate(texts):
```

```
    doc = nlp(sentence)
    for token in doc:
        tokenList.append([i, token.text, token.lemma_, token.pos_,
token.tag_, token.dep_])
tokenDF = pd.DataFrame(tokenList, columns=["i", "text", "lemma",
"POS", "tag", "dep"]).set_index("i")
```

Нажаль в цьому фреймворці не реалізовано підтримку української мови, але він був використаний лише для нормалізації вхідного тексту шляхом видалення з нього власних назв. Для цього початковий текст було автоматично перекладено на англійську мову за допомогою python бібліотеки deep-translator. Оскільки власні назви не перекладаються, то за допомогою вище зазначеного коду вдалося виокремити і видалити їх. Спочатку в перекладеному тексті, а потім в початковому. Тим самим, було проведено один з етапів нормалізації вхідних даних.

Наступні кроки попередньої обробки тексту проводилися за допомогою фреймворку Natural Language Toolkit (NLTK) функціонал якого було розширено за допомогою користувацьких функцій, щоб виконати задачу стемінгу для українського тексту.

```
def makeStemmed(texts):
    stemmed_texts_list = []
    for text in texts:
        tokens = word_tokenize(text)
        stemmed_tokens = [stemmer.stem(token) for token in tokens
if token not in stopwords]
        text = " ".join(stemmed_tokens)
        stemmed_texts_list.append(text)
    return stemmed_texts_list
```

Бібліотека sklearn була використана для реалізації машинного навчання для задачі NLP. Зокрема, за допомогою feature_extraction.text та TfidfVectorizer було виконане, по-перше, масштабування простору ознак за допомогою методу головних компонент (PCA). По-друге, векторизація вхідного тексту, для перетворення лінгвістичних даних у математичний об’єкт. Також через sklearn було реалізовано кластеризацію тексту, методом k-means.

Для візуалізації даних застосовувалися наступні бібліотеки: plotly, matplotlib, seaborn.

ВИСНОВКИ

Дослідження, що проведені в представленій кваліфікаційній бакалаврській роботі відповідають поставленій задачі. Результатами виконання є наступне.

1. Під час дослідження вдалося адаптувати метод кластерного аналізу до задач NLP. При цьому було розроблено спеціальний перелік евристичних правил нормалізації, які враховують специфіку вхідного навчального тексту. Цей етап дозволив покращити точність машинного навчання шляхом зменшення розмірності простору ознак та максимізації показника інформативності слів в навчальній вибірці.
2. Розглянуто підходи прикладного семантичного аналізу, який є невід'ємною частиною розробки інтелектуальних NLP компонент. На практиці було перевірено ефективність деяких з цих методів.
3. Були виявлені основні науково-методологічні проблеми розробки інтелектуальних компонент при роботі з українською мовою. Під час дослідження було запропоновано і реалізовано варіанти вирішення проблеми високої розмірності простору ознак NLP систем.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Ghosh, S., Gunning D. (2019) Natural Language Processing Fundamentals: Build intelligent applications that can interpret the human language to deliver impactful results 1st Edition, Packt Publishing, 2019. – 376 p.
2. Scitovski, R., Sabo K., Martínez-Álvarez F., Ungar S. (2021) Cluster Analysis and Applications, Springer Cham, 2021. – 271 p.
3. Hennig, C., Meila, M., Murtagh, F., Roberto Rocci, R. (2020) Handbook of Cluster Analysis Chapman and Hall/CRC; 1st edition, 2020 – 753 p.
4. Ihnatchyck, Y., (2023) What are the benefits of using Natural Language Processing (NLP) in Business? [Електронний ресурс] – Режим доступу до ресурсу: https://www.datasciencecentral.com/what-are-the-benefits-of-using-natural-language-processing-nlp-in-business/
5. Monga, J. (2023)Natural Language Processing: Empowering Machines to Understand Human Language [Електронний ресурс] – Режим доступу до ресурсу: https://medium.com/@jeremiahmonga/natural-language-processing-empowering-machines-to-understand-human-language-7e7d9e008fc5
6. Lee, J., Yang, S., Holland-Hall, C., Sezgin, E., Gill, M., Linwood, S., Huang, Y., Hoffman, J. (2022) Prevalence of Sensitive Terms in Clinical Notes Using Natural Language Processing Techniques: Observational Study, JMIR Med Inform;10(6), doi: 10.2196/38482.
7. Turchin, A., Florez Builes, L., F. (2022) Using Natural Language Processing to Measure and Improve Quality of Diabetes Care: A Systematic Review, Journal of Diabetes Science and Technology, 15(3) p. 553–560, doi: 10.1177/19322968211000831.
8. Кряжич, О. О., Трофимчук, О. М. (2022) Метод обробки неструктурованої інформації на веб-ресурсах, International Scientific Technical Journal "Problems of Control and Informatics", 67(4), с. 106–115. doi:10.34229/2786-6505-2022-4-8.

9. Карпов, І. А., Антоненко, С. В. (2020) Огляд методів інтелектуального аналізу тексту, Актуальні проблеми автоматизації та інформаційних технологій, Том 24, с. 40-46.
10. Погорілий, С.Д., Крамов, А.А., Білецький, П.В., (2019) Метод оцінки когерентності україномовних текстів з використанням згорткової нейронної мережі, Збірник наукових праць Військового інституту Київського національного університету імені Тараса Шевченка, с. 64-71, doi: https://doi.org/10.17721/2519-481X/2019/65-08
11. СЛОВНИК УКРАЇНСЬКОЇ МОВИ ONLINE. ТОМИ 1-11 [Електронний ресурс]– Режим доступу до ресурсу:https://services.ulif.org.ua/expl/Entry/index.
12. Sarangam A. Data Mining vs Data Analysis – An Easy Guide In Just 3 Points [Електронний ресурс] / Ajay Sarangam // Jigsaw. – 2021. – Режим доступу до ресурсу: https://www.jigsawacademy.com/blogs/data-science/datamining-vs-data-analysis/#Difference-between-data-mining-and-data-analysis.
13. Calzon B. Your Modern Business Guide To Data Analysis Methods And Techniques [Електронний ресурс] / Bernardita Calzon // DataPine. – 2021. – Режим доступу до ресурсу: https://www.datapine.com/blog/data-analysismethods-and-techniques/.
14. Remanan S. Association Rule Mining [Електронний ресурс] / Surya Remanan // Towards. Data Science. – 2018. – Режим доступу до ресурсу: https://towardsdatascience.com/association-rule-mining-be4122fc1793.
15. Биков, М. М. Б60 Основи інтелектуальних технологій. Частина 1. Технології розпізнавання : електронний навчальний посібник комбінованого (локального та мережного) використання [Електронний ресурс] / Биков М. М., Ковтун В. В., Гаврилюк В. О. – Вінниця : ВНТУ, 2023. – 229 с.
16. Кряжич, О. О., Трофимчук, О. М. (2022) Метод обробки неструктурованої інформації на веб-ресурсах, International Scientific Technical Journal "Problems of Control and Informatics", 67(4), с. 106–115. doi: 0.34229/2786-6505-2022-4-8.

17. Pathak, A. (2023) Natural Language Processing in Everyday Life: How AI is Transforming Communication. [Електронний ресурс] / Dr. D. Y. Patil School of Science & TechnologyРежим доступу до ресурсу: https://dypsst.dpu.edu.in/blogs/natural-language-processing-in-everyday-life-how-ai-is-transforming-communication
18. Висоцька, В. А. (2023) Аналіз та синтез комп'ютерних лінгвістичних систем опрацювання україномовного текстового контенту. Дисертаційна робота на здобуття наукового ступеня доктора технічних наук. Національний університет «Львівська політехніка», Львів, 2023. – 480 с.
19. How machine learning removes spam from your inbox [Електронний ресурс] – Режим доступу до ресурсу: https://bdtechtalks.com/2020/11/30/machine-learning-spam-detection/
20. The ultimate guide to sentiment and emotion analysis. [Електронний ресурс] – Режим доступу до ресурсу: https://callminer.com/blog/guide-to-sentiment-and-emotion-analysis
21. What Makes Your Smart Speaker Smart? [Електронний ресурс] – Режим доступу до: httpstoppandigital.com/machine-translation-email/what-makes-your-smart-speaker-smart/
22. What is text analytics and how does it work? [Електронний ресурс] – Режим доступу до ресурсу: https://callminer.com/blog/what-is-text-analytics-and-how-does-it-work
23. How natural language processing empowers consumers. [Електронний ресурс] – Режим доступу до ресурсу: https://www.cio.com/article/236850/how-natural-language-processing-serves-as-a-consumer-empowerment-strategy.html

ДОДАТОК

```
!pip install deep-translator

from deep_translator import GoogleTranslator
import nltk
import requests
import pandas as pd

url = 'https://raw.githubusercontent.com/olegdubetcky/Ukrainian-
Stopwords/main/ukrainian'
r = requests.get(url)
with open('nltk_data\\corpora\\stopwords\\ukrainian', 'wb') as f:
    f.write(r.content)

import codecs

file = codecs.open("dataset.txt", "r", "utf_8_sig")
texts = file.read().splitlines()
file.close()

texts
translated = []
for txt in texts:
    translated.append(GoogleTranslator(source='auto',
target='english').translate(txt))

with open("translatedDataset.txt", "w", encoding="utf-8") as file:
    for  line in translated:
        file.write(line + '\n')

classes = pd.DataFrame(translated, columns =['original'])

nltk.download('stopwords')
from nltk.corpus import stopwords

stopwords = stopwords.words("ukrainian")
stopwords.extend(['…', '«', '»', '...', 'т.д.', 'т', 'д'])

from nltk.tokenize import word_tokenize

import string
def remove_punctuation(text):
    return "".join([ch if ch not in string.punctuation else ' '
for ch in text])

def remove_numbers(text):
    return ''.join([i if not i.isdigit() else ' ' for i in text])

import re
def remove_multiple_spaces(text):
```

```
     return re.sub(r'\s+', ' ', text, flags=re.I)

from nltk.stem import *
from nltk.corpus import stopwords
from pymystem3 import Mystem
from string import punctuation

preperation =
[remove_multiple_spaces(remove_numbers(remove_punctuation(text.low
er()))) for text in classes["original"]]

classes['preparation'] = preperation

from nltk.stem.snowball import SnowballStemmer
stemmer = SnowballStemmer("ukrainian")

def makeStemmed(texts):
    stemmed_texts_list = []
    for text in texts:
        tokens = word_tokenize(text)
        stemmed_tokens = [stemmer.stem(token) for token in tokens
if token not in stopwords]
        text = " ".join(stemmed_tokens)
        stemmed_texts_list.append(text)
    return stemmed_texts_list

classes['stem'] = makeStemmed(classes['preparation'])

from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.feature_extraction.text import CountVectorizer,
TfidfTransformer
from sklearn.pipeline import Pipeline
from sklearn.decomposition import PCA
import plotly.express as px
import plotly.offline as pyo
import plotly.graph_objects as go

pyo.init_notebook_mode()

pipeline = Pipeline([
  ('vect', CountVectorizer()),
  ('tfidf', TfidfTransformer())])

from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.cluster import KMeans
from sklearn.cluster import SpectralClustering
from sklearn.cluster import MiniBatchKMeans
from sklearn.cluster import Birch
from sklearn.metrics import silhouette_score

import matplotlib.pyplot as plt
```

```
import seaborn as sns
import pickle
%matplotlib inline

import warnings
warnings.filterwarnings("ignore")

k = 3
classes = classes.drop(columns = ['clusters'],axis = 1)

# Set the number of clusters

# Vectorize the text
vectorizer = TfidfVectorizer(ngram_range=(1,1))
vectorizer.fit(classes['original'].values)
X = vectorizer.transform(classes['original'].values)

# Fit our Model
model = KMeans(init="k-means++", n_clusters=3, algorithm =
'elkan')
model.fit(X)

# Get the cluster labels
clust_labels = model.predict(X)
cent = model.cluster_centers_

labels = pd.DataFrame(clust_labels)
classes.insert((classes.shape[1]),'clusters',labels)

classes.groupby(['clusters']).count()

import textwrap

pca = PCA(n_components=3)

data3D = pca.fit_transform(X.toarray())

d = {'X': data3D[:,0], 'Y': data3D[:,1], 'Z':data3D[:,2], 'text':
classes.original, 'mark' : classes.clusters}
df = pd.DataFrame(data=d)
df["text"] = df["text"].apply(
    lambda t: "<br>".join(textwrap.wrap(t))
)

fig = px.scatter_3d(df, x="X", y="Y", z='Z', color = 'mark',
hover_name="text", log_x=False, log_y=False, log_z=False)

fig.show()

vectorizer = TfidfVectorizer(ngram_range=(1,1))
vectorizer.fit(classes['original'].values)
```

```
X = vectorizer.transform(classes['original'].values)

pca = PCA(n_components=2)

data3D = pca.fit_transform(X.toarray())

d = {'X': data3D[:,0], 'Y': data3D[:,1], 'text': classes.original,
'mark' : classes.clusters}
df = pd.DataFrame(data=d)
df["text"] = df["text"].apply(
    lambda t: "<br>".join(textwrap.wrap(t))
)

plt.plot(data3D[:,0], data3D[:,1], 'ob')

counts =
classes["original"].str.findall(r"(\w+)").explode().value_counts()
top_10 = counts.nlargest(50)

fig, ax = plt.subplots(figsize=(10,5))
sns.barplot(x=top_10.index, y=top_10.values, ax=ax)
ax.set_ylabel('Number of Occurrences', fontsize=12)
ax.set_xlabel('Word', fontsize=12)
ax.xaxis.set_tick_params(rotation=90)

import spacy
nlp = spacy.load("en_core_web_sm")

tokenList = []

for i, sentence in enumerate(texts):
    doc = nlp(sentence)
    for token in doc:
        tokenList.append([i, token.text, token.lemma_, token.pos_,
token.tag_, token.dep_])
tokenDF = pd.DataFrame(tokenList, columns=["i", "text", "lemma",
"POS", "tag", "dep"]).set_index("i")

comment_words = ''

for val in tokenDF[((tokenDF.POS == 'PROPN'))].text:
    val = str(val)
    tokens = val.split()
    for i in range(len(tokens)):
        tokens[i] = tokens[i].lower()
    comment_words += " ".join(tokens)+" "

from wordcloud import WordCloud, STOPWORDS

wordcloud = WordCloud(width = 1000, height = 300,
                background_color ='white',
```

```
                min_font_size = 10).generate(comment_words)

plt.figure(figsize = (10, 10), facecolor = 'white',
edgecolor='blue')
plt.imshow(wordcloud)
plt.axis("off")
plt.tight_layout(pad = 0)

plt.show()
```