Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
(повна назва)

Кафедра Інформаційних управляючих систем
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

Дослідження методів нечіткого пошуку текстової інформації
та їх використання в ІТ-проєктах електронної комерції
(тема)

Виконав:
студент 2 курсу, групи УПГІТм-22-2
Семенець Катерина Євгенівна
(прізвище, ім'я, по батькові)
Спеціальність 122 Комп'ютерні науки
(код і повна назва спеціальності)
Тип програми освітньо-наукова
(освітньо-професійна або освітньо-наукова)
Освітня програма Управління проєктами
в галузі інформаційних технологій
(повна назва освітньої програми)

Керівник зав. каф. ІУС Костянтин ПЕТРОВ
(посада, власне ім'я, прізвище)

Допускається до захисту

Зав. кафедри ІУС _____ Костянтин ПЕТРОВ
(підпис) (власне ім'я, прізвище)

2024 р.

Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук

Кафедра Інформаційних управляючих систем

Рівень вищої освіти другий (магістерський)

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

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

Освітня програма Управління проєктами в галузі інформаційних технологій
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
« 01 » квітня 2024 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Семенець Катерині Євгенівні
(прізвище, ім'я, по батькові)

1. Тема роботи Дослідження методів нечіткого пошуку текстової інформації та їх використання в ІТ-проєктах електронної комерції

затверджена наказом університету від 01 квітня 2024 р. № 258Cm
2. Термін подання студентом роботи до екзаменаційної комісії 05 червня 2024 р.
3. Вихідні дані до роботи науково-технічна література, публікації та інтернет-ресурси, що стосуються теми кваліфікаційної роботи; матеріали звіту з науково-дослідної практики

4. Перелік питань, що потрібно опрацювати в роботі огляд існуючих методів нечіткого пошуку текстової інформації та виявлення їхніх переваг та недоліків; створення комбінованого методу пошуку для використання в проєктах е-комерції; проведення експериментальної перевірки розробленого методу; аналіз отриманих результатів.

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Отримання завдання на кваліфікаційну роботу	01.04.2024	виконано
2	Пошук наукових публікацій, літературних джерел та інформаційних ресурсів у наукових виданнях і Інтернеті	02.04.2024 – 05.04.2024	виконано
3	Аналіз сучасного стану об’єкта дослідження	06.04.2024 – 10.04.2024	виконано
4	Формування проблеми щодо вдосконалення з урахуванням її актуальності	11.04.2024	виконано
5	Огляд існуючих підходів, моделей, методів та технологій управління змінами при реалізації ІТ-проєктів	12.04.2024 – 15.04.2024	виконано
6	Постановка задачі дослідження	16.04.2024	виконано
7	Розробка комбінованого методу пошуку тексту та аналіз його застосування в проєктах е-комерції	17.04.2024 – 22.04.2024	виконано
8	Практична апробація розробленого метода	23.04.2024 – 05.05.2024	виконано
9	Аналіз отриманих теоретичних та практичних результатів використання розробленої моделі	06.05.2024 – 12.05.2024	виконано
10	Оформлення пояснювальної записки	13.05.2024 – 25.05.2024	виконано
11	Підготовка презентації	25.05.2024 – 31.05.2024	виконано
12	Надання роботи для перевірки на плагіат	02.06.2024	виконано
13	Надання роботи на підпис науковому керівникові	03.06.2024	виконано
14	Надання роботи на рецензування	04.06.2024	виконано
15	Надання підписаної завідувачем кафедрою роботи в ЕК	05.06.2024	виконано
16	Захист кваліфікаційної роботи в екзаменаційній комісії	06.06.2024	виконано

Дата видачі завдання 01 квітня 2024 р.

Студент _____

(підпис)

Керівник роботи _____ зав. каф. ІУС Костянтин ПЕТРОВ

(підпис) (посада, власне ім’я, прізвище)

РЕФЕРАТ

Пояснювальна записка до магістерської кваліфікаційної роботи містить: 79 с., 4 розділи, 15 рисунків, 4 таблиці, 21 джерело, 1 додаток.

БАЗА ДАНИХ, ВІДСТАНЬ ЛЕВЕНШТЕЙНА, Е-КОМЕРЦІЯ, ІНДЕКСАЦІЯ, МЕТОДИ ОБРОБКИ ТЕКСТУ, МЕТОДИ РАНЖУВАННЯ, НЕЧІТКИЙ ПОШУК, РЕЛЕВАНТНІСТЬ РЕЗУЛЬТАТІВ, СТЕМІНГ, ТОКЕНІЗАЦІЯ, ФОНЕТИЧНІ АЛГОРИТМИ.

Об’єктом дослідження є процес нечіткого пошуку текстової інформації.

Предметом дослідження є методи нечіткого пошуку текстової інформації та їх використання в інформаційних системах електронної комерції.

Метою кваліфікаційної роботи є дослідження методів нечіткого пошуку текстової інформації та можливостей їх використання при створенні інформаційних систем в сфері е-комерції, а також розробка удосконаленого методу пошуку на основі комбінації найбільш ефективних з них для поліпшення користувацького досвіду, підвищення рівня релевантності результатів пошуку та рівня конкурентоспроможності електронних комерційних платформ.

Наукова новизна теми дослідження полягає в аналізі, розробці та виявленні нових теоретичних та практичних підходів до нечіткого пошуку текстової інформації та їх використання в ІТ-проєктах електронної комерції. Дослідження охоплює впровадження комбінованих методів пошуку, таких як алгоритм Левенштейна, та їх оптимізацію для поліпшення продуктивності та точності пошукових систем. Це дослідження дозволить вирішити проблеми, пов'язані з пошуком тексту з орфографічними помилками, багатомовністю та використанню неправильної розкладки у запитах користувачів, забезпечуючи тим самим більш ефективний та зручний пошук товарів в електронних комерційних платформах.

ABSRTACT

The explanatory note to the master's qualification thesis contains: 79 pages, 4 chapters, 15 figures, 4 tables, 21 sources, 1 application.

DATABASE, LEWENSTEIN DISTANCE, E-COMMERCE, INDEXING, TEXT PROCESSING METHODS, RANKING METHODS, Fuzzy Search, RELEVANCE OF RESULTS, STEMING, TOKENIZATION, PHONETIC ALGORITHMS.

The object of research is the process of fuzzy search for textual information.

The subject of research is the methods of fuzzy search of textual information and their use in electronic commerce information systems.

The purpose of the qualification work is to research the methods of fuzzy search for text information and the possibilities of their use in the creation of information systems in the field of e-commerce, as well as the development of an improved search method based on a combination of the most effective of them to improve the user experience, increase the level of relevance of search results and the level of competitiveness electronic commercial platforms.

The scientific novelty of the research topic consists in the analysis, development and discovery of new theoretical and practical approaches to fuzzy search of textual information and their use in e-commerce IT projects. The research covers the implementation of combined search methods, such as the Lowenstein algorithm, and their optimization to improve the performance and accuracy of search engines. This research will solve the problems associated with searching for text with spelling errors, multilingualism and the use of incorrect layout in user queries, thus providing more efficient and convenient product search in electronic commerce platforms.

ЗМІСТ

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАКИ

БД – база даних

ЕК – електронна комерція

ІТ – інформаційні технології

КМ – комбінований метод

МНП – методи нечіткого пошуку

ПС – пошукова система

API – Application Programming Interface – прикладний програмний інтерфейс

CA – Contextual analysis – метод контекстуального аналізу

LD – Levenshtein distance – метод відстані Левенштейна

NLP – Natural Language Processing – методи обробки природної мови

SE – SoundEx

ВСТУП

Сучасний стан електронної комерції відзначається високим рівнем розвитку та інтеграції в повсякденне життя споживачів. Інтернет-магазини пропонують безліч товарів і послуг, доступних лише за кілька кліків. Однак, разом із зростанням кількості пропозицій виникає проблема ефективного пошуку конкретних товарів, що стає все більш актуальною. Користувачі, які не можуть швидко знайти потрібний товар, схильні переходити на інші платформи, що знижує конкурентоспроможність та прибутковість інтернет-магазинів. Тому вдосконалення пошукових систем є критично важливим завданням для розробників електронних комерційних платформ.

Світові тенденції в цій галузі свідчать про активне впровадження нових технологій та методів обробки текстової інформації для покращення пошукових можливостей. Одним з перспективних напрямків є використання нечітких методів пошуку, які дозволяють зменшити вплив орфографічних помилок, різних мовних особливостей та неправильної розкладки клавіатури на результати пошуку. Комбінація різних методів пошуку та їх оптимізація стають основою для створення ефективних і точних пошукових систем, що забезпечують високу релевантність результатів.

Актуальність даного дослідження зумовлена необхідністю покращення користувацького досвіду та підвищення конкурентоспроможності електронних комерційних платформ. Використання удосконалених методів пошуку дозволить значно покращити точність та швидкість знаходження товарів, що, в свою чергу, підвищить задоволеність користувачів і сприятиме зростанню доходів інтернет-магазинів.

Об’єктом дослідження є процес нечіткого пошуку текстової інформації.

Предметом дослідження є методи нечіткого пошуку текстової інформації та їх використання в інформаційних системах електронної комерції.

Метою кваліфікаційної роботи є дослідження методів нечіткого пошуку

текстової інформації та можливостей їх використання при створенні інформаційних систем в сфері електронної комерції, а також розробка удосконаленого методу пошуку на основі комбінації найбільш ефективних з них для поліпшення користувацького досвіду, підвищення рівня релевантності результатів пошуку та рівня конкурентоспроможності електронних комерційних платформ.

Структура і логіка дослідження підпорядковані вирішенню поставлених завдань. При вирішенні конкретних завдань використовуються методи логічного, функціонального та системного аналізу, а також статистичні методи. Застосовується аналіз алгоритмів нечіткого пошуку за критеріями ефективності, точності, швидкості обробки та зручності інтеграції в IT-системи [1-2]. Особлива увага приділяється порівнянню різних алгоритмів і методів з метою визначення найбільш оптимальних для використання в електронній комерції.

Наукова новизна теми дослідження полягає в аналізі, розробці та виявленні нових теоретичних та практичних підходів до нечіткого пошуку текстової інформації та їх використання в IT-проєктах електронної комерції. Дослідження охоплює впровадження комбінованих методів пошуку, таких як алгоритм Левенштейна, N-грам, та їх оптимізацію для поліпшення продуктивності та точності пошукових систем. Це дослідження дозволить вирішити проблеми, пов'язані з пошуком тексту з орфографічними помилками, багатомовністю та використанню неправильної розкладки у запитах користувачів, забезпечуючи тим самим більш ефективний та зручний пошук товарів в електронних комерційних платформах.

Результати виконання дослідження включають розробку і впровадження удосконаленого методу нечіткого пошуку, який продемонстрував підвищену релевантність і точність результатів у порівнянні з існуючими методами. Це дозволить досягти значних покращень у функціонуванні пошукових систем інтернет-магазинів, забезпечуючи більш ефективний та зручний процес знаходження товарів для користувачів.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАДАЧІ ДОСЛІДЖЕННЯ

1.1 Використання нечіткого пошуку в системах е-комерції

Електронна комерція сьогодні стала невід'ємною складовою підприємницького середовища, пропонуючи споживачам широкий асортимент товарів та послуг, доступних за кілька кліків. Однак, разом із зростанням обсягів товарів та послуг, що пропонуються в інтернет-магазинах, зростає і складність пошуку конкретного товару для користувачів. Якщо користувач швидко не знаходить цікавий йому товар на одному сайті, він, швидше за все, перейде до іншого ресурсу. У зв'язку з цим, розробники повинні особливу увагу приділяти вдосконаленню процесу пошуку шляхом використання різних технологій [3].

Розглянемо більш детально сутність пошукових систем, які відіграють ключову роль у забезпеченні ефективного пошуку товарів та послуг в ЕК (електронній комерції). Пошукові системи – це складні програмні продукти, які здійснюють збір, індексацію та ранжування величезних обсягів інформації в Інтернеті. Основним завданням пошукових систем є забезпечення користувачам швидкого та точного доступу до релевантної інформації відповідно до їхніх запитів [4].

Розробники, що починають з нуля розробляти систему пошуку неодмінно стикнуться з тим, що “чіткий” (слово в слово як записано у базі даних) пошук може бути обмеженим у роботі з запитами користувачів у зв'язку з можливістю виникнення помилок у написанні запитів або використанням синонімічних слів. При введенні запиту точно відповідного до того, що користувач має на увазі, “чіткий” пошук може ефективно забезпечити релевантні результати. Однак, у реальному житті користувачі часто роблять помилки у написанні запитів або використовують інші слова з схожим значенням, пишуть не по порядку слова у словосполученнях назв товарів, а також випадково використовують іншу розкладку клавіатури, або навпаки, навмисне пишуть імена іншою, більш

звичною для них мовою.

Також, стосовно реалізації пошуку по базі даних інтернет-магазину, слід звернути увагу на те, що товари мають різноманітні характеристики, які можуть бути важливими для користувачів під час пошуку. Наприклад, в запитах щодо купівлі електроніки, споживачі можуть враховувати бренд, модель, технічні характеристики, розміри, вартість та інші параметри. У випадку пошуку одягу, користувачі можуть враховувати розмір, кольори, матеріали, стиль, типи одягу та інші фактори. Додатково, важливо розробляти функції фільтрації, які дозволяють користувачам обирати бажані параметри товарів, щоб забезпечити їм зручний та ефективний пошук [5].

Таким чином, необхідно вийти за межі дослівного пошуку задля забезпечення релевантності відображення товарів та послуг. Для цього і використовуються різноманітні МНП (методи нечіткого пошуку).

МНП замість того, щоб дати точну відповідь – дають наближену вірогідність відношення до деякого паттерну, що є більш гнучким і дієвим вирішенням проблеми. Ці методи використовують алгоритми, які дозволяють знаходити результати, що містять аналогічні або схожі слова, фрази або концепції, навіть якщо вони не точно співпадають зі словами запиту користувача [6].

Однак МНП існує дуже багато, і усі вони відрізняються один від одного варіантами вирішення однієї і тієї ж проблеми. Деякі з них використовують перестановки букв у слові, деякі базуються на правилах написання слів конкретною мовою, інші складають словники синонімів. Необхідно розглянути та проаналізувати як кожен з них допомагає в вирішенні проблеми релевантного пошуку.

1.2 Аналіз існуючих методів нечіткого пошуку

1.2.1 Методи, що базуються на використанні метрик Левенштейна та Дамерау-Левенштейна

Метрика Левенштейна (Levenshtein distance – LD) , також відома як відстань редагування, є одним з ключових методів в області нечіткого пошуку. Цей метод вимірює відстань між двома рядками символів, що вказує на кількість операцій (вставок, вилучень, замін), які потрібно виконати для перетворення одного рядку на інший.

При використанні LD спочатку обчислюється матриця, де кожен елемент відповідає відстані редагування між підрядками символів вихідного та цільового рядків. Потім шляхом динамічного програмування визначається мінімальна кількість операцій редагування, які необхідні для перетворення одного рядку на інший. На рис. 1.1 можна побачити принцип роботи алгоритму.

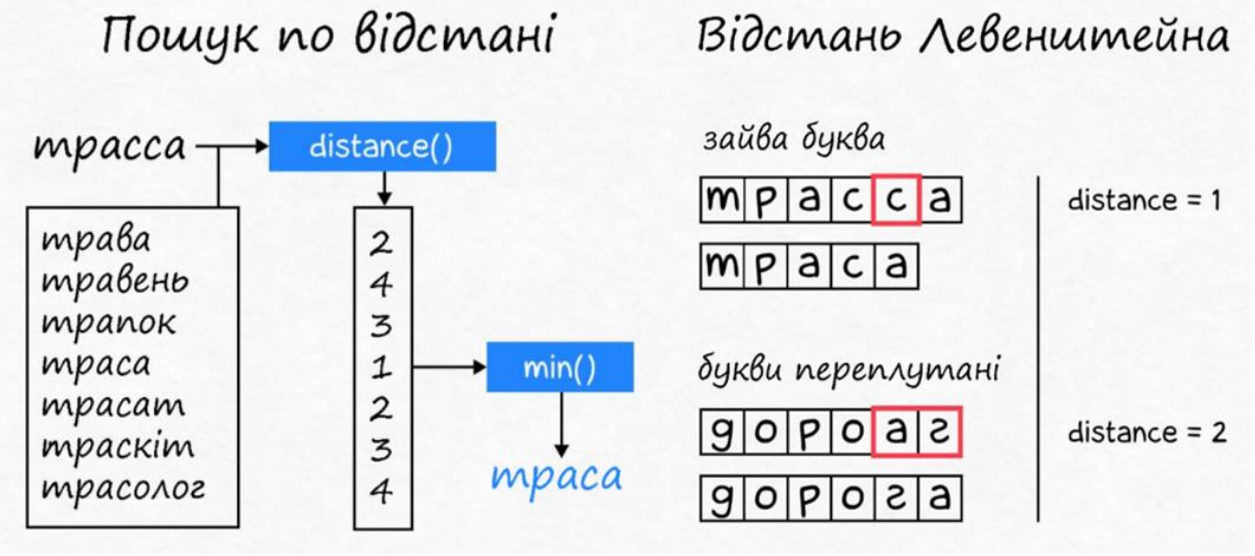

Рисунок 1.1 – Графічне представлення алгоритму пошуку по відстані

Алгоритм Дамерау-Левенштейна подібний до класичного алгоритму Левенштейна, але включає додатковий шаг для перевірки можливості

транспозиції. Це означає, що алгоритм розглядає не лише вставки, вилучення та заміни символів, але й можливість обміну місцями двох сусідніх символів для досягнення більшої схожості між рядками [7].

Перевагами методів, що базуються на використанні LD та його модифікацій є їх простота та універсальність. Вони можуть бути застосовані для порівняння будь-яких текстових даних, включаючи слова, фрази, або навіть документи. Також ці методи добре працюють з великими обсягами даних та враховують найменші різниці між рядками.

Проте, метод пошуку по відстані має деякі недоліки. Зокрема, він може бути вимогливим до обчислювальних ресурсів, особливо при порівнянні великих текстів. Крім того, він може бути не так ефективним для деяких специфічних видів даних або мов програмування, де важлива інша метрика схожості.

1.2.2 Метод N-грам

Метод N-грам є одним з популярних МНП, який базується на аналізі послідовностей символів у словах або фразах. У цьому методі, текст розбивається на невеликі фрагменти, відомі як N-грами, де "N" позначає кількість символів у фрагменті. Наприклад, у біграмах "яблуко" розбивається на "яб", "бл", "лу", "ук", "ко".

Основна ідея методу N-грам полягає у тому, щоб знайти спільні N-грами між запитом користувача і текстом у базі даних. Чим більше спільних N-грам між запитом та текстом, тим більше ймовірність, що текст відповідає на запит користувача [8].

Найбільш часто використовуваними на практиці є триграмми – підрядки довжини 3. Вибір більшого значення N веде до обмеження на мінімальну довжину слова, при якому вже можливо виявити помилку [9]. На рис. 1.2 зображена візуалізація пошуку слова “КАРТОПЛЯ” в базі даних товарів

магазину з використанням триграм. Триграми слова зустрічаються в деяких інших словах в базі даних, але тільки в слові “КАРТОПЛЯ” є усі вони, а тому 6 з 6 співпадінь буде найпріоритетнішим результатом пошуку.

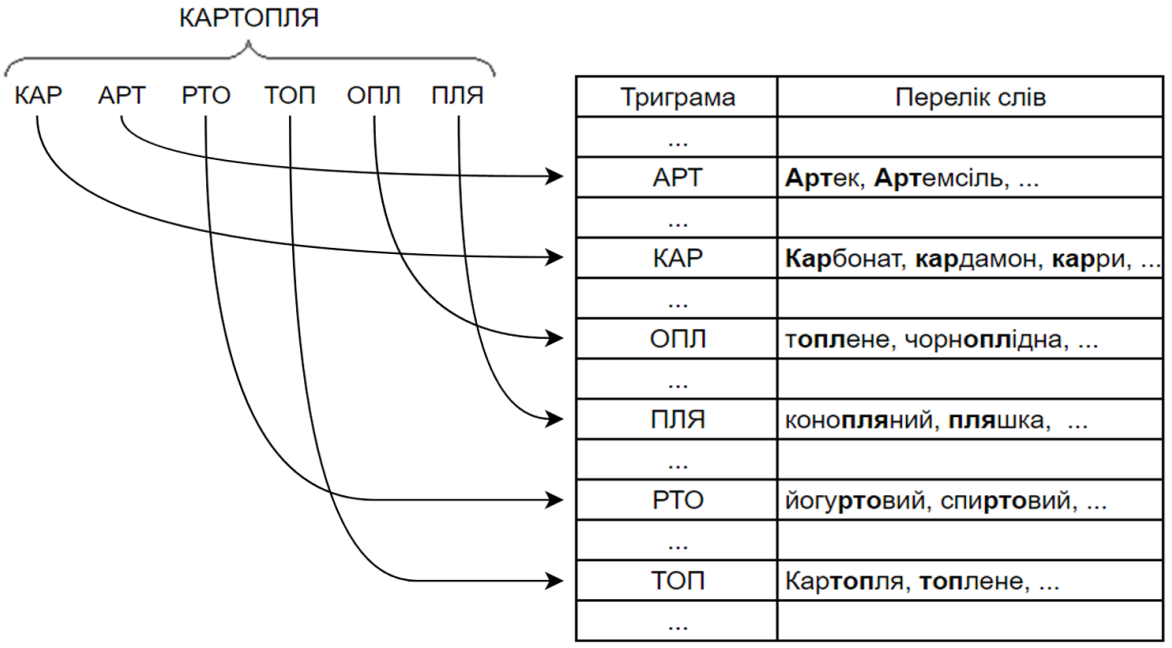

Рисунок 1.2 – Триграми слова “КАРТОПЛЯ” та їх співпадіння з іншими словами в БД

Перевагами методу N-грам є простота реалізації та ефективність у виявленні схожих фрагментів тексту, навіть якщо вони містять деякі помилки або відмінності. Також цей метод може бути ефективним для пошуку великих обсягів тексту, оскільки він оперує лише невеликими фрагментами.

Проте метод N-грам також має свої обмеження. Він може бути менш ефективним у виявленні схожих текстів, якщо вони містять багато варіантів внутрішньої структури або схожих фрагментів, які відрізняються тільки місцем розташування. Крім того, для більш точного порівняння текстів потрібно враховувати не тільки окремі N-грами, але й їх контекст у межах тексту.

1.2.3 Метод SoundEx

Метод SoundEx (SE) є алгоритмом фонетичного кодування, призначеним для перетворення слова на короткий код, який представляє його фонетичну звучність. Цей метод був розроблений для пошуку слів, які звучать подібно, але можуть бути написані по-різному. Основна ідея полягає в тому, щоб призначити код кожному слову, що відображає його звучання, з врахуванням фонетичних характеристик [10].

Метод SE може бути корисним у випадках, коли користувачі використовують різні варіанти транслітерації при складанні своїх запитів для пошуку. Наприклад, якщо користувач шукає товар або послугу, назва якого може мати деяку транслітераційну невизначеність (наприклад, між кирилицею та латиницею), метод SE допоможе знаходити схожі фонетично слова незалежно від їхнього написання. Таким чином, навіть якщо запит користувача містить різні варіації транслітерації, система може використати SE для знаходження релевантних результатів пошуку, що сприяє покращенню точності і повноти пошуку.

Проте, метод SE має деякі недоліки, зокрема, він може бути нечутливим до деяких видів орфографічних варіантів та не завжди точно відображати фонетичні властивості слів. Крім того, він може бути менш ефективним у випадках, коли слова мають значну фонетичну варіативність або складно розрізнити їх вимовляння.

1.2.4 Методи токенізації та стемінгу

Методи токенізації та стемінгу є ключовими для обробки тексту в пошукових системах. Вони допомагають перетворити тексти на множину слів

або токенів, що полегшує пошук та аналіз. Роз’яснимо кожен термін окремо та як вони можуть бути поєднані для досягнення однієї цілі.

Токенізація – це процес розбиття тексту на окремі компоненти, які можна аналізувати окремо, такі як слова, фрази або символи. Зазвичай слова виокремлюються з тексту на основі пробілів, пунктуації та інших роздільників. Токени можуть також бути створені шляхом видалення зайвих символів, нормалізації тексту або врахування специфічних правил для конкретного типу документів [11].

Стемінг, з іншого боку, є процесом видалення афіксів та закінчень зі слів, залишаючи їх кореневу форму або стему. Наприклад, слово "картопляний" після стемінгу стане "картопл". Це допомагає зменшити обсяг даних та забезпечити консистентність у виразі однакових понять.

Токенізація та стемінг можуть бути поєднані для оптимального аналізу тексту в пошукових системах. Під час токенізації текст розбивається на окремі слова або фрази, після чого застосовується стемінг, який зводить слова до їхніх кореневих форм, що допомагає у виявленні схожих слів та покращує релевантність пошукових результатів. Вони дозволяють зменшити обсяг даних, спрощуючи процес пошуку та аналізу. Також вони допомагають групувати слова з однаковими коренями разом, що полегшує розуміння тексту та підвищує релевантність результатів пошуку [12].

Однак, важливо розуміти, що стемінг може мати свої обмеження. Наприклад, він може призводити до втрати деякої частини значення слів або до неправильного визначення стем у випадку слів з різними значеннями. Також, не всі мови мають однакові правила стемінгу, тому може бути потрібно використовувати різні підходи для різних мов або текстових корпусів.

1.2.5 Метод тезауруса

Метод тезауруса – це техніка пошуку інформації, що ґрунтується на використанні схожих або семантично пов'язаних термінів для знаходження релевантної інформації. Тезаурус – це структурований словник, який містить синоніми, антоніми та інші взаємовідношення між словами.

У методі тезауруса слова групуються в синонімічні групи або тематичні категорії, що дозволяє покращити результати пошуку шляхом включення до пошукового запиту синонімів або асоційованих термінів. Наприклад, якщо користувач шукає "смартфон", пошукова система може також врахувати слова "мобільний телефон" або "гаджет", які є семантично пов'язаними з цим поняттям. У разі пошуку кілець для весілля, тезаурус зробить асоціацію з поняттями "каблучки" чи "обручки". А у випадку пошуку "косметики", тезаурус може також включати "засоби для догляду за шкірою", що розширює можливості знаходження користувачем необхідних засобів для догляду за собою [13].

Однією з переваг методу тезауруса є можливість покращити точність та повноту пошуку шляхом врахування семантичних взаємозв'язків між термінами. Він також допомагає у вирішенні проблеми синонімів, коли користувач може використовувати різні терміни для позначення одного й того ж поняття.

Проте, метод тезауруса може мати деякі обмеження, такі як обмежений обсяг та актуальність тезаурусу. Також побудова та підтримка актуального тезаурусу може бути трудомісткою задачею.

1.2.6 Методи контекстного аналізу

Методи контекстного аналізу (Contextual analysis – CA) включають в себе різноманітні підходи до аналізу та розуміння контексту, в якому знаходиться

певна інформація чи подія. Основна мета цих методів – розкрити смислові зв'язки та взаємовплив між об'єктами або явищами, що дозволяє краще розуміти та інтерпретувати інформацію.

Одним з методів СА є аналіз семантики тексту. Цей підхід передбачає виявлення смислових зв'язків між словами, реченнями та текстовими документами. Зазвичай використовуються методи обробки природної мови (Natural Language Processing – NLP), які дозволяють автоматично розпізнавати та аналізувати семантичну структуру тексту для виявлення ключових тем, ідей та взаємозв'язків [14].

Інший метод – контентний аналіз, який полягає в систематичному описі та класифікації контенту на основі певних параметрів чи категорій. Цей підхід широко використовується в дослідженнях соціальних мереж, медіа та інших джерел інформації для виявлення тенденцій, ставлень або тематик, які є актуальними для досліджуваної групи чи спільноти.

Також методи СА можуть включати в себе аналіз графів та мереж. Цей підхід полягає у вивченні зв'язків між об'єктами у вигляді вузлів та зв'язків між ними. Аналіз графів може допомогти виявити ключові вузли, групи або взаємозв'язки, які можуть впливати на динаміку та структуру контексту.

Усі ці методи використовуються для підвищення розуміння контексту та забезпечення більш глибокого та інформативного аналізу інформації. Вони допомагають виявляти ключові аспекти та тенденції, що важливі для прийняття рішень, розробки стратегій чи розвитку досліджень.

Одним з основних недоліків є складність реалізації. Розробка та впровадження алгоритмів контекстного аналізу може бути складною та вимагати значних зусиль та ресурсів, особливо для невеликих інтернет-магазинів або компаній з обмеженими технічними знаннями.

Ще одним недоліком є потреба великого обсягу даних. Багато методів СА, таких як аналіз тексту або графів, потребують великого обсягу даних для ефективної роботи. Для малих бізнесів або стартапів, які мають обмежений обсяг даних, це може стати перешкодою у використанні таких методів.

Також важливо враховувати складність валідації результатів. При використанні методів СА необхідно враховувати, що отримані результати можуть бути непередбачуваними або неправильно інтерпретованими. Це може виникнути через складність алгоритмів або недостатню чіткість вихідних даних.

Таким чином, не зважаючи на потенційні переваги, методи контекстного аналізу мають свої обмеження і вимоги, які слід ураховувати при їхньому використанні в практичних застосунках.

1.3 Постановка задачі дослідження

В ході проведеного аналізу МНП текстової інформації для використання в ІТ-проєктах електронної комерції було виявлено декілька потенційних недоліків кожного з них:

- вимогливість до ресурсів у випадках великої кількості даних;
- мала ефективність у випадках, коли користувач використовує дуже специфічні терміни;
- хибність результатів, коли слова мають схожу вимову, але різний контекст;
- неможливість адаптуватись через те, що різні мови мають різні правила побудови речення та семантику;
- залежність від великого обсягу даних для ефективної роботи деяких методів [15].

Таким чином, враховуючи вищезазначені недоліки, можна визначити мету, об'єкт та предмет дослідження.

Метою кваліфікаційної роботи є дослідження методів нечіткого пошуку текстової інформації та можливостей їх використання при створенні інформаційних систем в сфері ЕК, а також розробка удосконаленого методу пошуку на основі комбінації найбільш ефективних з них для поліпшення

користувацького досвіду, підвищення рівня релевантності результатів пошуку та рівня конкурентоспроможності ЕК платформ.

Об’єктом дослідження є процес нечіткого пошуку текстової інформації.

Предметом дослідження є методи нечіткого пошуку текстової інформації та їх використання в інформаційних системах ЕК.

Для досягнення сформульованої вище мети потрібно вирішити наступні основні завдання:

- дослідити складові частини процесу використання методів нечіткого пошуку у контексті реалізації ІТ-проєктів в ЕК;
- проаналізувати можливість використання комбінації різних методів нечіткого пошуку текстової інформації для поліпшення релевантності пошукової системи;
- вдосконалити або розробити комбінований метод базуючись на перевагах методів нечіткого пошуку в аспекті ІТ-проєктів ЕК;
- перевірити ефективність розробленого методу на практиці;
- проаналізувати отримані результати.

2 ДОСЛІДЖЕННЯ МЕТОДІВ НЕЧІТКОГО ПОШУКУ У КОНТЕКСТІ РЕАЛІЗАЦІЇ ІТ-ПРОЄКТІВ В СФЕРІ ЕЛЕКТРОННОЇ КОМЕРЦІЇ

2.1 Аналіз процесу пошуку в контексті електронної комерції.

Для кращого розуміння того як можна покращити ПС (пошукові системи) в контексті електронної комерції, необхідно розглянути основні дії, через які проходить система та користувач в процесі вибору товарів в інтернет магазині. Процес пошуку в ЕК включає в себе наступні етапи:

1) введення запиту користувачем. Користувач вводить ключові слова або фрази, що характеризують товар або послугу, яку він бажає знайти;

2) обробка запиту. ПС отримує запит від користувача та проводить обробку цього запиту для пошуку відповідних товарів або послуг у базі даних;

3) пошук товарів. Здійснюється пошук товарів або послуг, які відповідають запиту користувача. Цей процес може включати пошук за ключовими словами, фільтрацію за параметрами (наприклад, ціною, брендом, розміром тощо), а також врахування персоналізованих рекомендацій;

4) відображення результатів. Знайдені товари або послуги відображаються користувачу у вигляді списку чи каталогу, що містить інформацію про кожен з них, таку як назва, зображення, опис, ціна тощо;

5) вибір та купівля товару. Користувач переглядає результати пошуку, обирає підходящий товар або послугу, та здійснює покупку [16].

Кожен з цих етапів є важливим в процесі реалізації товарів в інтернеті. Насамперед тому, що від запиту попереднього пункту залежить результати роботи наступного, а в фіналі – купівля товару або перехід користувача на інший інтернет ресурс. Розглянемо які складові відіграють ключову роль в процесі надання якісних результатів пошукового запиту:

1) точність та релевантність результатів пошуку значно залежать від якості самого запиту, який вводить користувач. Наявність в запиті помилок, слів в скороченій формі чи на мові сленгу погіршують спроможность системи видати

якісний результат. Чим чіткіше та специфікованіше сформульований запит, тим точніші і релевантніші будуть результати. Наприклад, якщо користувач шукає "чорні кросівки Adidas" в інтернет-магазині, запит більш специфікований, ніж просто "кросівки". В результаті ПС може забезпечити видачу лише чорних кросівок торгової марки Adidas, що відповідає бажанню користувача;

2) грамотна структура та якісне наповнення бази даних (БД) є важливими для забезпечення ефективного та точного пошуку в електронній комерції. Ось кілька аспектів, які слід враховувати;

а) структурованість даних: дані про товари та послуги повинні бути організовані у логічні категорії та підкатегорії. Наприклад, в категорії "Електроніка" можуть бути підкатегорії "Смартфони", "Ноутбуки", "Планшети" тощо. Це допомагає користувачам швидше знаходити потрібні їм товари;

б) актуальність даних: інформація в базі даних повинна бути постійно оновлюваною, щоб відображати поточний асортимент товарів та послуг. Наприклад, якщо товар вже не доступний для продажу, він повинен бути видалений з бази даних або позначений як "недоступний";

в) повнота та точність даних: кожен товар або послуга повинні мати достатньо детальний опис, включаючи фотографії, характеристики, ціни та іншу важливу інформацію. Наприклад, у описі смартфона можуть бути зазначені його технічні характеристики, розмір екрану, обсяг пам'яті тощо;

г) стандартизація даних: використання стандартизованих форматів для подання даних дозволяє забезпечити їхню узгодженість та легкість аналізу. Наприклад, усі ціни можуть бути вказані в одній валюті, а розміри – у стандартних одиницях вимірювання;

д) класифікація товарів: класифікація товарів за спеціальними атрибутами, такими як бренд, розмір, категорія тощо, допомагає здійснювати більш точний та специфікований пошук [17] ;

е) додатково, не буде зайвим зробити прихований атрибут, в якому буде міститись альтернативні специфічні варіанти опису товара, на той випадок, якщо користувач не знає точної назви продукту і шукає його за допомогою

фізичних характеристик предмета або ціллю його використання;

3) для ефективної роботи ПС важливо мати належну функціональність, яка забезпечує точні та релевантні результати для користувачів. Ось деякі аспекти функціональності ПС;

а) алгоритми пошуку: ПС повинна мати потужні алгоритми пошуку, які дозволяють ефективно знаходити товари та послуги, що відповідають запитам користувачів. Наприклад, алгоритми, які враховують схожість запиту користувача з текстом опису товару, можуть допомогти забезпечити більш точні результати;

б) ранжування результатів: функціональність ранжування дозволяє відсортувати результати пошуку за різними критеріями, такими як релевантність, популярність, ціна тощо. Найчастіше можна зустріти, що ПС автоматично ранжує товари згідно з їхньою актуальністю або рейтингом користувачів;

в) фільтрація результатів: можливість фільтрувати результати пошуку за різними параметрами дозволяє користувачам точніше вибирати те, що вони шукають. Наприклад, фільтри за ціною, брендом, розміром тощо допомагають звужувати кількість варіантів і знаходити потрібний товар швидше;

г) мультимовна підтримка: функціональність, яка дозволяє користувачам здійснювати пошук та отримувати результати на різних мовах, розширює аудиторію і полегшує здійснення покупок міжнародними користувачам;

ґ) підтримка семантичного пошуку: система може використовувати семантичний пошук для знаходження товарів або послуг, які відповідають значенню запиту користувача, а не просто ключовим словам. Наприклад, система може розпізнати схожі терміни або синоніми та включати їх у результати пошуку;

4. сучасні великі компанії використовують у своїх пошукових системах принцип надання результатів пошуку заснований на конверсії товарів. Основна ідея полягає в тому, щоб відображати в перших результатах пошуку ті товари, які мають найвищу ймовірність конверсії, тобто ті, які користувачі ймовірніше

придбають. Це досягається завдяки алгоритмам машинного навчання, які аналізують дані про попередні покупки, перегляди товарів, рейтинги та відгуки користувачів. Товари, які мають високу конверсію, частіше відображаються на більш видимих позиціях у пошукових результатах, що збільшує їх шанси бути поміченими і придбаними користувачами. Такий підхід допомагає оптимізувати процес пошуку для покупців та збільшує ефективність ЕК для продавців [18].

Отже, аналіз процесу пошуку в контексті ЕК показує, що точність та релевантність результатів пошуку значно залежать від як мінімум чотирьох факторів: якості запиту користувача, якості структури та наповненню бази даних, а також від функціональності ПС. Крім того, принцип надання результатів пошуку, заснований на конверсії товарів, може стати ключовим для покращення якості ПС і буде сприяти підвищенню ефективності ЕК [19].

Незважаючи на те, що точність та релевантність результатів пошуку значно залежать від якості запиту користувача, неможливо повністю контролювати процес його формулювання. Користувачі можуть використовувати різноманітні форми запитів, іноді неправильно або недостатньо конкретно формулюючи їх, що може призвести до неякісних результатів. Тому для поліпшення користувацького досвіду від роботи з ПС необхідно, щоб інтерфейс ПС пропонував користувачам підказки за змістом, автодоповнення або варіанти схожих запитів.

2.2 Аналіз можливостей використання методів нечіткого пошуку у ІТ-проєктах створення інформаційних систем електронної торгівлі.

Розглянемо можливості та доцільність використання різних нечітких методів в ПС е-комерції.

Методи токенізації, стемінгу, лематізації та індексації можна розглядати як логічну послідовність. Розберемося як ці методи утворюють потужний алгоритм

дій з обробки тексту для подальшого його використання в ПС.

Застосування токенізації є невід’ємною частиною при роботі та аналізі тексту, вона дозволяє ефективно розділити текст на слова, усуваючи непотрібні символи та пробіли, і отримати набір токенів, які легко порівняти з базою даних товарів та послуг.

Наприклад з опису товару “Смузі «Крафтяр» з ківі, полуницею та бананом.”, по перше, буде видалено символи, що не є буквами («»,.), по друге, речення буде поділено за пробілами і утворить наступний список: Смузі, Крафтяр, з, ківі, полуницею, та, бананом. На цьому етапі додатково можна замінити всі великі літери в словах на маленькі, що дозволить не залежати ПС від регістру слів бази даних чи запита користувача.

Після розподілу усіх описів товарів на окремі слова, можна помітити, що в різних описах товарів зустрічаються слова, що мають однакове походження, але різні закінчення. Наприклад слова “курка”, “куряче”, “курячі” та “курячий” означають що товар або виготовлений з курки (ковбаски курячі, фарш курячий), або є її частиною (куряче крило). Для того, щоб система могла в певній мірі “розуміти” контекст запиту користувача та відобразити усі необхідні товари, можна використати принципи стемінгу та лематизації слів.

Стемінг використовує правила (відсікання закінчень, ознак множини, подвоєння та інше) та дозволяє зводити слова до їхніх основ, тим самим уніфікуючи слова з різними закінченнями чи формами. Наприклад, слова “куряче”, “курячі” та “курячий” будуть стемінгуватися до спільної основи "куряч", що спрощує пошук усього що зроблено з курки, але слово “курка” стемінгується до “курк”, що вже відрізняється і не буде об'єднуватися з попередніми товарами.

Для вирішення цієї проблеми можна застосовувати лемінг слів. Лемінг (лематизація) – це процес приведення слова до його початкової або базової форми, званої лемою. На відміну від стемінгу, який просто відсікає закінчення слів і може іноді призвести до втрати частини значення слова, лематизація використовує морфологічний аналіз, щоб знайти точну базову форму слова. Для

цього створюється словник, що містить усі слова та їхні леми. А під час пошуку необхідно тільки застосувати лемінг до запиту і знайти від базової форми цього слова усі леми, що зустрічаються в базі даних. Слід зазначити, що для кожної мови створюється окремий словник із лемами, при чому характерними мовами, що використовуються в опиті товарів в українських магазинах є українська та англійська [20].

Під час створення додаткових словників зі стемами та лемами, системі необхідно запам’ятовувати де саме використовувалося те чи інше слово серед бази даних, і для цього використовують індексацію.

Індексація є критично важливим елементом у ПС. Вона дозволяє створювати структури, які набагато швидше надають доступ до інформації, зменшуючи час пошуку та збільшуючи точність результатів. Індексування даних передбачає створення словника, де кожне слово асоціюється зі списком товарів, у яких це слово зустрічається. Це значно прискорює процес пошуку, оскільки ПС не повинна проходити через всю базу даних щоразу при запиті користувача. Замість цього, система звертається до індексу, що містить тільки необхідну інформацію, та швидко знаходить відповідні товари.

Індексація також забезпечує оптимізацію ресурсів, дозволяючи системі ефективніше використовувати пам'ять та обчислювальні потужності, за рахунок попередньої обробки даних перед запуском усієї системи, що є великим плюсом для великих інтернет-магазинів з великим асортиментом товарів.

Поєднавши тільки токенізацію, стемінг, лематизацію та індексацію можна вже побудувати ПС, що буде в певній мірі надавати релевантні результати пошуку, грунтуючись на зведенні слів до їх основ. Але як тільки користувач зробить хоча б одну помилку в слові, система не надасть жодного результату.

Усі описані дії з текстом добре підходять, щоб використовувати їх для попередньої обробки даних у наступних методах порівняння текстів.

Розглянемо можливості використання методів, що базуються на застосуванні відстані Левенштейна.

Для того, щоб знайти релевантні результати пошуку запитів із помилками

необхідно скористатися нечіткими методами пошуку. Одним із найпростіших для реалізації є алгоритм відстані Левенштейна. Замість надання чіткої відповіді, алгоритм надає оцінку подібності до всіх слів БД. Ця оцінка відстані між словами також в певній мірі дозволила б ігнорувати різні закінчення в словах (як альтернатива застосуванню сметінга та лелітизації), достатньо тільки відсортувати результати по найменшій оцінці подібності.

Скоріш за все велика кількість товарів в магазині буде мати однакові назви, а змінною частиною буде тільки колір, модель або інші їхні характеристики. Враховуючи це, перед застосуванням метода LD, який використовує порівняння слів із запита користувача з усіма словами з бази даних, необхідно попередньо обробити базу даних використовуючи методи токенізації та індексації і створити додатковий словник, що буде містити унікальні токени з усієї БД, де кожному слову буде відповідати перелік індексів товарів, в описі якого це слово зустрічається.

Базовий варіант цього алгоритму має часову складність O(m*n) і використовує O(m*n) пам'яті (рис. 2.1), де m та n – довжини порівнюваних рядків. Під час пошуку деякого слова з запросу користувача цей алгоритм застосовується до кожного слова і словник БД, тобто складність збільшується до O(m*n*d), де d — розмір словаря БД магазину.

		в	е	л	о	к	р	е	с	л	о
	0	1	2	3	4	5	6	7	8	9	10
в	1	0	1	2	3	4	5	6	7	8	9
е	2	1	0	1	2	3	4	4	5	6	7
л	3	2	1	0	1	2	3	4	5	5	6
о	4	3	2	1	0	1	2	3	4	5	5
с	5	4	3	2	1	1	2	3	3	4	5
и	6	5	4	3	2	2	2	3	4	5	6
п	7	6	5	4	3	3	3	3	4	5	6
е	8	7	5	5	4	4	4	3	4	5	6
д	9	8	6	6	5	5	5	4	5	5	6

Рисунок 2.1 – Матриця розрахунків відстані Левенштейна

У випадку якщо в запиті користувача не одно, а декілька слів, то для забезпечення повнотекстового пошуку (з урахуванням того, що слова в реченні можуть стояти в різному порядку) необхідно шукати усі слова окремо, тому складність збільшується на кількість слів в запиті O(m*n*d*k), де k – кількість окремих слів в запиті.

Щоб зрозуміти який це обсяг роботи, візьмемо вибірку з бази даних середнього розміру магазину, що містить різні категорії, підкатегорії та детальну інформацію про товари, наприклад, з сайту магазину Сільпо.

Категорія “Фрукти та овочі” має близько 500 унікальних слів в назвах товарів. При врахуванні того, що середня довжина слова в словнику складає 9 літер, складність пошуку словосполучення “Яблуко Симиренка” буде приблизно дорівнювати: O(m*n*d*k) = O(6*6*500+9*6*500) = O(45000). Враховуючи те що в магазині багато є 18 різних категорій, складність пошуку одного запита по всій базі товарів збільшиться ще в один-два десятка разів.

При такому підході до пошуку необхідно додатково шукати можливості до оптимізації, бо споживання часу та пам’яті такого варіанту реалізації алгоритму завелике, як для одного єдиного запиту. Один із варіантів полягає в тому, щоб відсікати ті варіанти слів, які відрізняються більше ніж на t операцій, тоді складність алгоритму набуває вигляду: O(t min(m*n)*d*k).

Також можна модифікувати алгоритм, поділивши вартість операції заміни на два варіанта, де перший варіант повертає 0,5 якщо необхідна літера стоїть поруч з порівнюваною на клавіатурі, і повертає 1 коли літера стоїть далі. Швидкості алгоритму не додасться, але це дозволить збільшити актуальність шуканої інформації.

Під час пошуку “Томат Чорний принц” по всій базі даних магазину Варус, що містить 7100 унікальних токенів, було витрачено в сумі 0,027 секунд використовуючи класичний алгоритм відстані Левенштейну (при чому, для пошуку цього запиту було запущено алгоритм тричі – для кожного окремого слова). Але застосовуючи модифікацію відсікання на t операціях, швидкість збільшилася і становить 0,018 секунд на пошук того самого запиту.

Розглянемо можливості використання методу N-грам в системах електроної комерції.

Метод N-грам був придуманий досить давно, і є найбільш широко використовуваним, тому що його реалізація вкрай проста, і він забезпечує досить хорошу продуктивність. Алгоритм ґрунтується на наступному принципі: "Якщо слово А збігається зі словом Б з урахуванням кількох помилок, то з великою часткою ймовірності у них буде хоча б 1 загальний підряд довжини N".

Використовуючи цей принцип у пошуку, необхідно спочатку розбити кожне окреме слово на N-грами та записати в загальний словник, при чому кожній N-грамі будуть відповідати усі індекси товарів БД, слова в описі яких містять ці частинки. Під час пошуку запит також розбивається на N-грами, і для кожної з них проводиться послідовний перебір та порівняння із елементами словника N-грам.

Оцінка часової складності алгоритму N-грам залежить від кількох факторів, зокрема від довжини опису товарів (L), розміру n (довжина N-грам), кількості товарів в базі даних (N), довжини запиту (Q) та специфічних операцій, які виконуються над N-грамами (генерація, порівняння, збереження). Часова складність алгоритму для пошуку n-грам разраховується так:

1) часова складність генерація N-грам для запиту: O(Q);

2) часова складність порівняння N-грам з базою даних: $O(L \cdot N)$;

3) загальна часова складність: $O(Q) + O(L \cdot N) = O(L \cdot N)$.

Цей метод, на відміну від методу, що використовує відстань Левенштейна, дозволяє шукати повнотекстово, враховуючи випадки, коли запити користувачів мають різну структуру чи порядок слів, відрізняючись від того як опис товару зберігається в БД. Це означає, що під час розбиття запита користувача на N-грами, не має потреби шукати кожне слово окремо по всій БД, як це було в попередньому методі. Кожна N-грама запиту проходить окремий цикл по словнику N-грам, що є значно меншим за всю БД за рахунок повторюваності частинок в різних словах.

Слід враховувати, що саме триграми дають максимальну ефективність під

час пошуку, оскільки вони забезпечують кращий баланс між точністю контексту та обчислювальними ресурсами.

Хоча біграми (2-грам) і будуть займати меньше місця і часу на перебор (за рахунок числа перестановок літер абетки P(33, 2) = 1056), але вони є надто короткими, часто зустрічаються в різних контекстах і можуть призводити до більшої кількості хибних збігів, тоді як чотириграми (4-грам) та більші n-грам, хоч і забезпечують ще точніші результати, потребують значно більше обчислювальних ресурсів та пам'яті (за рахунок збільшеного словника N-грам, кількість елементів якого може складатися з числа перестановок P(33, 4), але по факту ця кількість значно менше, залежно від обсягу БД).

Триграми ж дозволяють знизити неоднозначність в текстах, краще враховують морфологічну варіативність слів і створюють індекси помірного розміру, що робить їх оптимальним вибором для задач пошуку та аналізу текстів в ЕК.

Під час аналогічного пошуку “Томат Чорний принц” по словнику усіх N-грам бази даних (усього 7786 унікальних триграм) магазину Варус (7100 унікальних токенів), було витрачено 0,001 секунду на пошук запиту.

І хоча цей метод майже в 20 разів швидше впорався із пошуком запита “Томат Чорний принц” ніж метод LD, релевантність пошуку запиту “Хліб” хоча б із однією помилкою різко впаде, бо це слово має тільки дві триграми, одна з яких не буде валідною. В той самий час метод LD легко впорається з цим завданням, хоча і витратить більше часу.

Розглянемо можливості використання фонетичних методів обробки тексту в системах ЕК.

Наступним викликом для ПС є зміна розкладки клавіатури користувача. Є дві очевидні проблеми, що пов’язані з цим. Перша – коли користувач хоче знайти товар, який має назву в БД англійською, але він не переключає розкладку і друкує це слово українською так, як воно чується. Наприклад, пошук слова “рошен” має надати усі товари, де в назві зустрічається “Roshen”, але для методів LD та N-грам ці два слова не мають жодної подібності.

Друга проблема полягає в тому, що користувач може випадково набрати український текст англійськими літерами, як часто буває, коли використовуєш комп’ютер. Наприклад запит “ijrjkflys werthrb hjity” насправді означає, що користувач шукає “шоколадні цукерки рошен”. В цьому випадку спочатку необхідно розпізнати помилку та перевести всі літери в правильну розкладку, а потім вже шукати запит. Для ПС виникає необхідність асоціювати усі три варіанта “рошен”, “Roshen” та “hjity” як одне і те саме.

Для вирішення цієї проблеми можна скористатися фонетичними методами обробки тексту, такими як: SoundEx або Metaphone.

Алгоритм SoundEx був розроблений для індексації слів на основі їх фонетичної подібності, що дозволяє знайти схожі на звучання слова, навіть якщо вони написані по-різному. Алгоритм перетворює слово в чотиризначний код, який складається з першої літери слова і трьох цифр, що представляють інші звуки. Після видалення всіх голосних та деяких приголосних (h, w, y), кожна залишкова буква замінюється на цифру за визначеними правилами, що зображені в першому стовпчику рис. 2.2. Повторювані цифри зводяться до однієї, а код доповнюється нулями або обрізається, щоб завжди мати довжину в чотири символи.

Правила перетворення літери на цифру			
SoundEx:		**Покращений SoundEx:**	
B, P, F, V	1	B, P	1
C, S, K, G, J, Q, X, Z	2	F, V	2
D, T	3	C, S, K	3
L	4	G, J,	4
M, N	5	Q, X, Z	5
R	6	D, T	6
		L	7
		M, N	8
		R	9

Слово:	SoundEx:	Покращений SoundEx:
Roshen	R250	R0308
Raffaello	R140	R02070
Nutella	N340	N06070
Fratelli	F634	F906070
Maasdam	M235	M03608
Brivais	B612	B90203
Barbacoa	B612	B091030
Millennium	M450	M070808
Milani	M450	M07080

Рисунок 2.2 – Правила та приклади методів базового та покращеного SoundEx

Неважко помітити, що після всіх цих процедур залишається всього лише 7 тисяч різних варіацій такого чотиризначного коду, що тягне за собою безліч

абсолютно нічим не схожих один на одного слів. Таким чином, результат у більшості випадків включає в себе велику кількість «хибно-позитивних» значень.

Саме тому був розроблений покращений SoundEx, що використовує більше правил: літери розбиті на більшу кількість груп, голосні замінюються на 0, код транскрипції не має фіксованої довжини і не обрізається. Ці зміни значно покращують розгалуженість між кодами різних слів і на рис. 2.2 можна побачити, що код деяких слів із БД в базовому алгоритмі однаковий, на відміну від покращеної версії, де ті ж слова мають різні транскрипції.

Основним недоліком цього методу є те, що він не враховує особливостей звучання англійських слів під час створення коду слова. Це означає, що неможливо створити однозначну відповідність українських літер до англійських. Наприклад слово “knight”, звучить як “найт”, але коди обох слів виглядають як: “K80406” та “H046” – що для ПС є зовсім різними словами.

Оскільки розроблювана ПС орієнтована обробку тексту з бази даних, що містить переважну більшість слів українською мовою, і тільки близко 31% англійських слів, доцільніше буде системі перетворювати англійські слова в транскрипцію українськими символами, а не навпаки. Користувачі з більшою вірогідністю будуть шукати товар з іноземною назвою набираючи в пошуку текст рідною мовою, ніж будуть вгадувати, як ця назва правильно пишеться.

За основу для такого перетворення можна взяти метод Metaphone. Він використовує основні правила звучання англійської мови в різних випадках та перетворить його на код, що містить все ті ж символи англійського алфавіту, тільки з однозначним варіантом його вимови. Наприклад слово “knight” перетвориться на “nait”, що доволі однозначно можна перетворити на “найт”.

Тепер, достатньо зробити окремий словник транскрипції для усіх англійських слів і порівнювати з запитом користувача одним із методів подібності текстів. А у випадку коли користувач забуде переключити розкладку і введе запит українською мовою, але англійськими символами, можна перетворити текст в українські літери, і спочатку спробувати знайти запит як воно є, і при відсутності результату – використати коротку функцію, що замінить

символи англійською на відповідні українською, так як вони є на клавіатурі, та провести пошук з початку.

Розглянемо можливості використання методу тезауруса в системах е-комерції.

Тезаурус у контексті ЕК являє собою структуровану базу даних синонімів, пов'язаних термінів та інших семантичних зв'язків між словами і фразами. Він допомагає зрозуміти різноманітні запити користувачів і надає більш точні результати пошуку.

Одна з основних переваг методу тезауруса полягає в його здатності розпізнавати синоніми та варіації запитів. Наприклад, користувач може шукати "мобільний телефон", "смартфон" або навіть конкретну модель, як-от "iPhone". Використовуючи тезаурус, ПС здатна розпізнати, що всі ці терміни відносяться до однієї категорії товарів, і надасть релевантні результати незалежно від точного формулювання запиту. Це особливо корисно в мультикультурних і багатомовних контекстах, де різні користувачі можуть використовувати різні терміни для позначення одного і того ж продукту.

Крім того, метод тезауруса може використовуватися для управління категоріями товарів та навігації по сайту. Створюючи семантичні зв'язки між різними категоріями і підкатегоріями, тезаурус дозволяє автоматично пропонувати користувачам пов'язані товари, покращуючи крос-продажі. Наприклад, при перегляді сторінки товару "ноутбук", система може запропонувати аксесуари, такі як "чохли для ноутбуків", "мишки" або "зовнішні жорсткі диски", завдяки зв'язкам, встановленим у тезаурусі.

Тезауруси, хоч і корисні, мають значні недоліки. Основні мінуси включають обмеженість і статичність даних, що призводить до швидкого застарівання інформації і недостатньої кількості синонімів. Вони не враховують контекст, що може викликати неправильне тлумачення запитів, особливо з омонімами та полісемією. Підтримка актуальності тезаурусів, особливо в контексті постійного оновлення асортименту товарів магазину, вимагає значних ресурсів і спеціалізованих знань.

2.3 Дослідження можливостей поєднання різних методів нечіткого пошуку.

Кожен МНП має свої унікальні переваги та обмеження, що впливають на їх ефективність в різних контекстах. Метод Левенштейна, наприклад, чудово працює для виявлення орфографічних помилок та незначних варіацій у написанні слів. Проте він може бути неефективним при роботі з великими обсягами тексту, оскільки його обчислювальна складність збільшується з довжиною слів.

N-грам метод, у свою чергу, ефективно обробляє довгі тексти, розбиваючи їх на підрядки, загальна кількість яких, як правило, займає менше місця ніж БД загалом, що дозволяє витрачати менше часу на пошук. Однак, в контексті пошуку товарів, короткі запити з помилками не будуть знайдені через відсутність співпадінь. А зменшивши число N для утворення менших підрядків призведе до більшої кількості хибних збігів.

Метод Metaphon забезпечує можливість обробки слів використовуючи правила фонетичних структур різних природних мов, але це може бути занадто складним для реалізації в деяких випадках. Причому швидкість цього метода залежить від кількості правил, що необхідно перебрати для створення фонетичної транскрипції слів. Враховуючи особливості англійської мови, складно створити такі інструкції, щоб однозначно описували звучання слів, бо існує багато винятків різного читання однієї літери в різних обставинах.

Розглядаючи слово “ocean”, можна зіткнутися з проблемою різного читання, хтось прочитає його правильно “оушен”, хтось не знайомий з правилами читання літери “c” та прочитає як “океан”. Скоріш за все є необхідність запам’ятовувати оба варіанта, використовуючи тезаурус. В той самий час не існує чітких правил під читання таких винятків, як “Island” (читається айленд, замість “айсленд” або “ісленд”), залишається тільки запам’ятати таке слово цілком. При чому, таких слів винятків може бути багато, і тоді системи необхідно додати стільки нових правил, скільки слів винятків може зустрітись в БД, що значно уповільнює процес формування транскрипції слів.

Отже, для досягнення релевантних результатів ПС в е-комерції, необхідно комбінувати різні методи нечіткого пошуку, використовуючи найбільш придатні методи в ситуаціях, коли інший метод не в змозі знайти шуканий текст. При цьому, необхідно якісно організувати порядок застосування методів, заздалегідь привести базу даних в відповідні форми, які необхідні для роботи різних методів, а також знайти баланс між швидкістю, складністю та релевантністю ПС.

3 РОЗРОБКА КОМБІНОВАНОГО МЕТОДУ ПОШУКУ ТЕКСТОВОЇ ІНФОРМАЦІЇ

3.1 Етапи комбінованого методу пошуку текстової інформації е-комерції

Будемо розглядати процес пошуку користувачем товарів в базі даних в контексті клієнт-серверної архітектури. Тобто база даних, алгоритми обробки та пошуку тексту – все це працює на сервері, тоді як користувач взаємодіє з системою через клієнтський інтерфейс.

Сервер відповідає за отримання запиту від клієнта, обробку цього запиту та пошук відповідних товарів у базі даних за допомогою алгоритмів (що застосовуються в комбінованому методі) та повернення результатів клієнту. Клієнтський інтерфейс, у свою чергу, відображає ці результати користувачу, забезпечуючи зручність взаємодії та можливість уточнення запиту. Цей підхід дозволяє ефективно розподілити обчислювальні ресурси і забезпечити швидкий та точний пошук товарів, враховуючи специфіку запитів користувачів.

Розробка комбінованого методу пошуку передбачає не тільки поєднання різних алгоритмів та технологій для забезпечення точного та релевантного пошуку товарів, а й пре- та після- обробку даних. Тому основні етапи методу (рис. 3.1) можна представити таким чином:

1) оновлення бази даних та уніфікування характеристик товарів;

2) обробка бази даних та створення словників перед запуском сервера;

3) пошук інформації згідно з запитом користувача та оцінка релевантності результатів;

4) видача результатів пошуку користувачеві.

Кожен з цих етапів відіграє важливу роль у забезпеченні максимальної точності та ефективності пошуку.

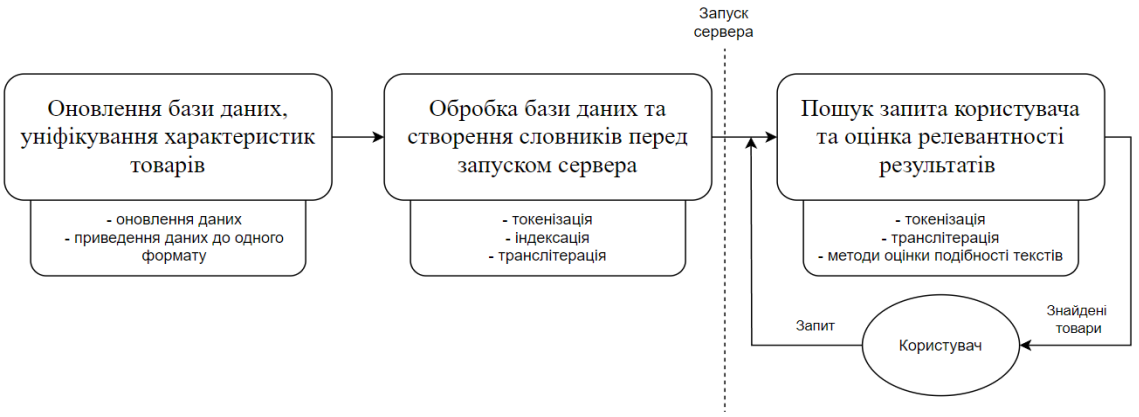

Рисунок 3.1 – Основні етапи роботи комбінованого методу

Розглянемо кожен етап методу більш детально.

3.1.1 Оновлення бази даних, уніфікування характеристик товарів

Для дослідження МНП була використана база даних українського магазину продовольчих товарів Сільпо. Перед початком запуску та роботи серверу, що буде обробляти користувацькі запити, необхідно переконатись, що база даних відповідає наступним критеріям.

1. Актуальність даних.

Актуальність даних забезпечується регулярним збором даних про товари з різних джерел, таких як постачальники, партнери та власні каталоги. За відсутності комунікації можна розглянути методи автоматизованого збору даних (веб-скрапінг, API інтеграції) та ручного введення інформації.

Для отримання БД Сільпо був застосований метод парсингу веб сторінок магазину. Усього під час цього процесу було зібрано 44140 товарів.

2. Очищення, нормалізація та уніфікація даних.

Після збору даних, необхідно очистити їх від помилок, дублювання та зайвої інформації, забезпечити стандартизацію форматів, перетворення одиниць

вимірювання (єдиний формат: літри, кілограми) та уніфікацію назв категорій та атрибутів товарів.

Після уніфікації усіх зібраних товарів, в БД зберігається наступні характеристики товару: id категорії, id підкатегорії, назва, вага, одиниці виміру ваги (л, кг, шт), посилання на картинку, посилання на товар сайту Сільпо. Треба зазначити, що марка товару та інші характеристики не розбиваються на окремі атрибути, а перелічені безпосередньо в назві товару.

3. Категоризація товарів.

Важливо, щоб в магазині була єдина системи категорій та підкатегорій, це забезпечує узгодженість та структурованість бази даних, така система легко масштабується та оновлюється.

Структура БД Сільпо має 24 категорії та 230 підкатегорій товарів. Ієрархія та назви категорій занесени в окрему таблицю БД, що дозволяє займати меньше місця. На рис. 3.2 зображено структуру товарів на веб сайті магазина Сільпо.

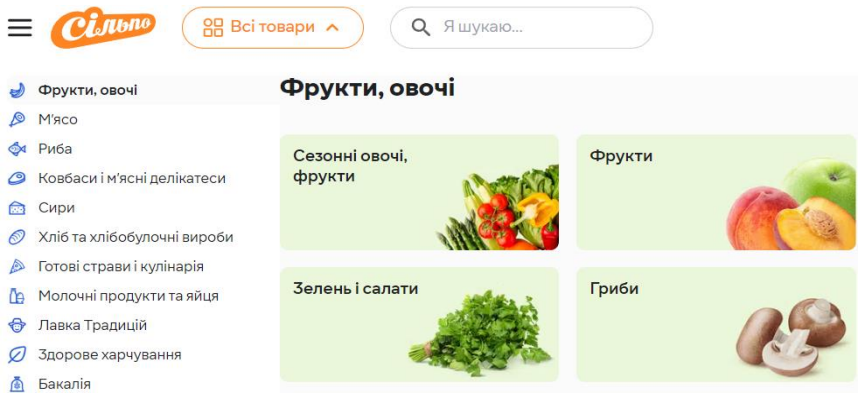

Рисунок 3.2 – Категорії та підкатегорії магазину Сільпо

4. Доступ до БД.

Для того, щоб взаємодіяти з БД (прочитати, записати, видалити чи оновити дані) з будь якого модулю серверу, без повторення великих обсягів коду, необхідно використовувати модуль з відповідними функціями, що реалізують команди по зверненню до БД. На рис. 3.3 зображено частку використаних функцій з швидкою комунікацією з БД.

```
2 usages
def create_table(name, path=path_db):...
1 usage
def add_data_to_table(name, items, crash=False, path=path_db):...

def update_data(name, what, value, where=None, path=path_db):...

def delete_data(name, where=None, ask=False, path=path_db):...
15 usages
def get_data_from_table(name, select=None, where=None, path=path_db):
    Items = []
    with sqlite3.connect(path) as db:
        query = f"SELECT {select if select else '*'} FROM {name}{f' WHERE {where}' if where else ''}"
        data = db.execute(query)
        for el in data:
            Items.append(el)
    return Items
```

Рисунок 3.3 – Функції, що дозволяють взаємодіяти з базою даних

3.1.2 Обробка бази даних та створення словників перед запуском сервера

Перед запуском сервера, але після очищення, нормалізації та уніфікації інформації про товари, необхідно обробити ці дані – перетворити їх у формат, в якому вони будуть використовуватися в методах нечіткого пошуку запиту користувача.

По перше, необхідно позначити, що усі текстові операції необхідно виконувати, як над назвою, так і над характеристиками товару. Тому для всіх товарів необхідно спочатку утворити об’єднаний опис. Оскільки магазин Сільпо не має окремих характеристик товарів, та зберігає назву, торгову марку та складники його продукту безпосередньо в назві товару, то опис набуває наступний вигляд: назва + вага. Пошук по ціні товару зазвичай не виконується з міркувань зворотного пошуку, тобто користувач хоче знайти товар, щоб дізнатися його ціну, а не навпаки.

Весь процес розділення БД на різні словники зображений на рис. 3.4 та складається з наступних етапів.

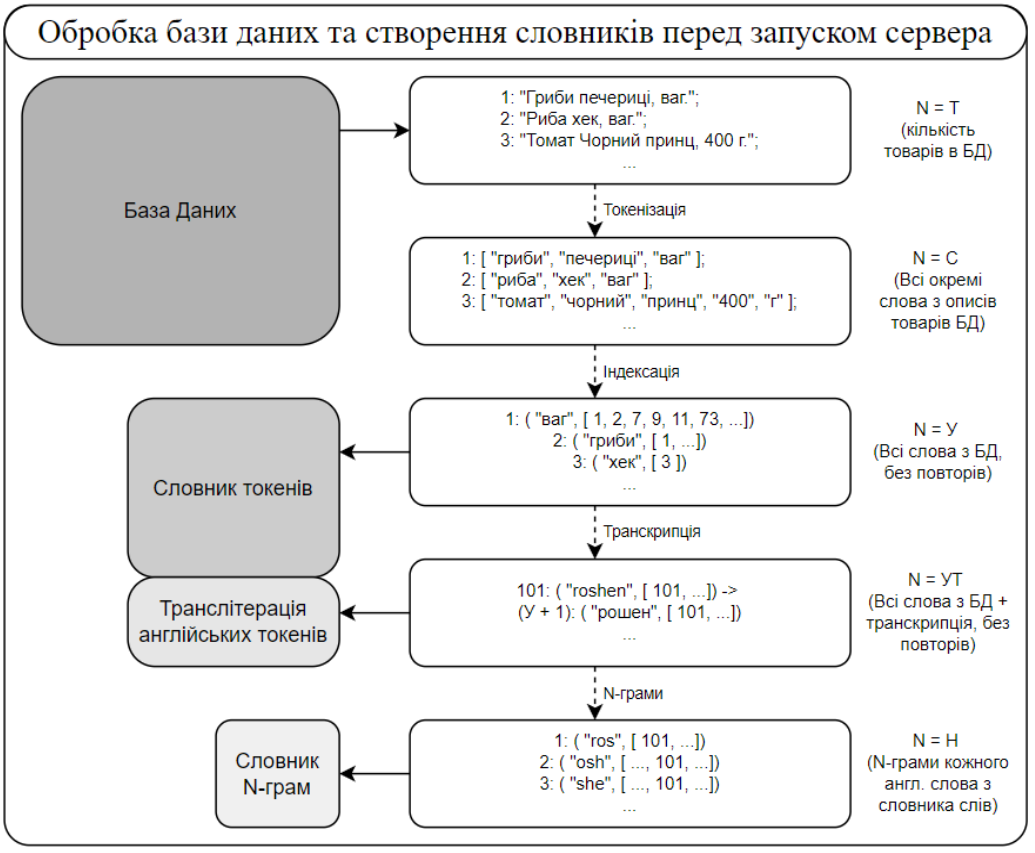

Рисунок 3.4 – Основні етапи обробки та розділення БД

1. Токенізація товарів.

Оскільки основою комбінованого метода пошуку буде використання відстані Левенштейна, необхідно застосувати поділення опису товару на окремі слова, для їх подальшого порівняння та оцінки подібності.

З першого погляду цей процес виглядає легким – поділити текст по пробілам, комам, крапкам та іншим знакам, якими закінчують речення. Але існує ряд винятків, що пов’язані з використанням крапки не тільки як символ поділу між реченнями. На рис. 3.5 зображено винятки, що використовуються в англійській мові, та варіанти регулярних виразів, які допоможуть розпізнати такі крапки, на які не треба ділити речення. В базі товарів Сільпо також зустрічаються подібні слова: Dr.Pepper, Dr.Oetker, Happy.com, тощо.

Виняток	Приклад	Регулярний вираз
Акроніми	U.S.A	([A-Z][.][A-Z][.](?:[AZ][.])?)
Префікси	Mr. John	(Mr|St|Mrs|Ms|Dr)[.]
Число	9.0 31.006%	/^\d*\.+\d*$/
Суфікси	John Johnson Jr. Nike Inc.	(Inc|Ltd|Jr|Sr|Co)
Посилання	Google.com wikipedia.org	[.](com|net|org|io|gov|ru|ua)

Рисунок 3.5 – Винятки поділення речення по крапкам

Для поділення описів товарів на правильні токени було використано бібліотеку статистичної обробки природної мови nltk.tokenize та її функцію word_tokenize(text). Наприклад, опис товару “Пакет для подарунку Happy.com паперовий 18x22 см” буде поділено на наступні слова ['Пакет', 'для', 'подарунку', 'Happy.com', 'паперовий', '18x22', 'см'].

2. Індексація слів.

Після токенізації відразу можна помітити велику кількість повторів токенів між різними товарами: [“Томат”, “червоний”], [“Томат”, “рожевий”], [“Томат”, “на”, “гілці”], тощо. Слід також пам’ятати, що “Томат” та “томат” – різні строчки, тому всі слова в подальшому будуть переведені до нижньої розкладки. Отже, наступним кроком необхідно скласти словник усіх унікальних токенів БД. Цей процес є простим і доволі швидким, функція створення словника показана на рис. 3.6.

```
44  def indexation(items):
45      # процес розкладання усіх слів БД в словник, де слову відповідає масив з індексами
46      index_dict = {}
47      for ind, item in enumerate(items):
48          for token in word_tokenize(item):
49              token = token.lower()
50              if token not in index_dict.keys():
51                  index_dict[token] = []
52              index_dict[token].append(ind)
53      return index_dict
```

Рисунок 3.6 – Функція токенізації та індексації

Можна помітити, що на рис. 3.4 обсяг Бази даних візуально більше ніж у Словника слів. Якщо порахувати довжину тексту усіх не індексованих товарів по БД, обсяг буде дорівнювати 1.551 млн символів. На відміну від цього, загальна довжина усіх слів Словника слів складає 214 тис слів, що зменшує в 7 разів область перебору слів методу відстані Левенштейна.

3. Транскрипція слів.

Тепер необхідно зробити транскрипцію усіх не українських слів, частка яких складає 31%. Як було зазначено раніше, україномовні користувачі скоріше напишуть англійське слово українськими літерами, так як воно для них звучить. Тож для процесу транскрипції було використано основну ідею алгоритму Metaphone – використання фонетичних правил англійської мови та адаптація під українську абетку.

Було проаналізовано та опрацьовано основні особливості звучання англійських слів та складено перелік правил, які будуть застосовуватись в процесі транскрипції. Перелік усіх правил, регулярних виразів та прикладів перетворення можна побачити в табл. 3.1.

Таблиця 3.1 – Правила транскрипції англійських буквосполучень

Регулярний вираз	Правило	Приклад
(r'kn', 'н')	kn -> н	'knight' -> 'найт'
(r'igh', 'ай')	igh -> ай	'light' -> 'лайт'
(r'wr', 'р')	wr -> р	'write' -> 'райт'
(r'ph', 'ф')	ph -> ф	'phone' -> 'фон'
(r'sh', 'ш')	sh -> ш	'irish' -> 'іріш'
(r'tion', 'шн')	tion -> шн	'action' -> 'екшн'
(r'ight', 'айт')	ight -> айт	'knight' -> 'найт'
(r'ough', 'аф')	ough -> аф	'enough' -> 'енаф'
(r'ght', 'т')	ght -> т	'knight' -> 'найт'
(r'gh$', '')	gh в кінці слова ігнорується	'though' -> 'доу'
(r'gh', 'г')	gh -> г	'ghost' -> 'гоуст'
(r'ch', 'ч')	ch -> ч	'cheese' -> 'чіз'
(r'qu', 'кв')	qu -> кв	'queen' -> 'квін'

Кінець таблиці 3.1

Регулярний вираз	Правило	Приклад
(r'u(?=[c])', 'a')	u перед ck -> a	'duck' -> 'дак'
(r'c(?=[eiy])', 'c')	c перед e, i, y -> c	'cent' -> 'сент'
(r'ea', 'i')	ea -> i	'eat' -> 'іт'
(r'th', 'з')	th -> з	'this' -> 'зіс'
(r'oo', 'y')	oo -> y	'food' -> 'фуд'
(r'ee', 'i')	ee -> i	'see' -> 'сі'
(r'ai', 'ей')	ai -> ей	'rain' -> 'рейн'
(r'ou', 'ay')	ou -> ay	'out' -> 'аут'
(r'sion', 'жн')	sion -> жн	'vision' -> 'віжн'
(r'ge', 'дж')	ge -> дж	'orange' -> 'орандж'
(r'gy$', 'джи')	gy -> дж	'psychology' -> 'псічолоджи'
(r'gy', 'дж')	gy -> дж	'gym' -> 'джим'
(r'ck', 'к')	ck -> к	'duck' -> 'дак'
(r'x', 'кс')	x -> кс	'box' -> 'бокс'
(r'[^aeiou]h', '')	h, якщо після неї йде приголосна, видаляється	'ghost' -> 'гоуст'
(r'y$', 'i')	y на кінці слова -> i	'happy' -> 'хепі'
(r'(?<=[aeiou])h', '')	h після голосної видаляється	'oh' -> 'о'
(r'^y', 'й')	y на початку слова -> й	'yellow' -> 'йелоу'
(r'([aeiouy])\1', r'\1')	подвоєні голосні зводяться до одного	'feel' -> 'філ'
(r'([bcdfghjklmnpqrstvwxyz])\1', r'\1')	подвоєні приголосні зводяться до одного	'bottle' -> 'ботл'
(r'ng', 'нг')	ng -> нг	'ring' -> 'ринг'
(r'oi', 'ой')	oi -> ой	'oil' -> 'ойл'
(r'au', 'о')	au -> о	'autumn' -> 'отомн'
(r'ow', 'оу')	ow -> оу	'now' -> 'нау'
(r'aw', 'о')	aw -> о	'saw' -> 'со'
(r'j', 'дж')	j -> дж	'jam' -> 'джем'

Усі правила, зазначені в табл. 3.1, зберігаються в класі Phonetics та застосовуються по черзі в функції транскрипції, в тому порядку, в якому вони перелічені. Зазначена послідовність необхідна для коректного перетворення

буквосполучень, бо спочатку йдуть більш специфічні правила, а в кінці більш загальні. Наприклад, для слова “knight”, спочатку актуальніше застосувати правило “kn -> н” та “ight -> айт”, чим правила “gh -> г” або “k -> к”.

Окрім функції транскрипції, також створені допоміжні функції, що перевіряють розкладку, якою написано текст, та функція заміни розкладки (перетворює “z,kerj” на “яблуко”). Повний код класу Phonetics зображено на рис. 3.7.

```
7   class Phonetics:
8       transcription_rules = [...]
52      trans_letters = str.maketrans("aeiouybcdfghklmnprstvwz", "аеіоуібкдфгхклмнпрстввз")
53      ukrainian_chars = 'йцукенгшщзхїфівапролджєячсмитьбю'
54      english_chars = 'qwertyuiop[]asdfghjkl;\'zxcvbnm,.'
55      english_layout = str.maketrans(ukrainian_chars, english_chars)
56      ukrainian_layout = str.maketrans(english_chars, ukrainian_chars)
57
        1 usage
58      def transcribe(self, text):
59          text = text.lower()
60          for pattern, replacement in self.transcription_rules:
61              text = re.sub(pattern, replacement, text, flags=re.IGNORECASE)
62          text = text.translate(self.trans_letters)
63          return text
64
        4 usages
65      def check_en_layout(self, text):...
        1 usage
69      def check_uk_layout(self, text):...
        2 usages (1 dynamic)
73      def change_layout(self, text):
74          if self.check_en_layout(text):
75              return text.translate(self.ukrainian_layout)
76          else:
77              return text.translate(self.english_layout)
```

Рисунок 3.7 – Код класу Phonetics

Після того, як було застосовано функцію транкципції до всіх не українських слів, словник унікальних слів необхідно збільшити на кількість цих слів. Зробивши один раз транскрипцію усіх слів, не доведеться це робити під час пошуку слів, що б значно сповільнило роботу алгоритму.

4. Створення англійських N-грам.

У останньому кроці вирішено використовувати Алгоритм N-грам, для того, щоб перевірити чи є шукане англійське слово у базі даних. Оскільки цей

алгоритм має велику швидкість пошука (в 30-40 разів швидше ніж відстань Левенштейна, при умові однакової задачі пошука), його можна використати як попередню перевірку існування слова.

Така перевірка забезпечить швидку оцінку використання неправильної розкладки клавіатури. Якщо користувач ввів свій запит в англійській розкладці, а метод N-грам не знаходить переконливих співпадінь (коефіцієнт оцінки нижче 0.5 серед можливих [0, 1]), то необхідно “змінити розкладку” у слова та спробувати знайти основним алгоритмом. Така перевірка поширюється тільки слова написані англійською мовою.

Після токенізації назв, індексації слів, транскрипції англійських слів та створення словника англійських N-грам, етап попередньої обробки даних завершується і можна приступати безпосередньо до застосування комбінованого методу пошуку запитів.

3.1.3 Пошук інформації згідно з запитом користувача та оцінка релевантності результатів

Основою ПС є метод оцінки відстані Левенштейна та допоміжні методи попередньої обробки запиту користувача, ті самі, що були застосовані до всієї бази даних. Додатково проводиться оцінка релевантності результатів пошуку шляхом агрегації результатів роботи LD. Весь процес пошуку запита користувача представлений на рис. 3.8.

Розглянемо цей процес більш детально.

ПС отримує запит, який відразу поділяється на токени. Наступні дії будуть застосовуватись до кожного токена по черзі.

По перше, відбувається перевірка токена функцією check_en_word(text, dict_ngrams, items), яка перевіряє методом N-грам наявності токена в заздалегідь створеному словарі англійських слів, а далі, або повертає назад токен без змін,

або змінює його розкладку. Цей процес відбувається швидко, і займає близько 0,001 секунди для слів будь якої довжини в запиті (швидкість такого пошуку майже повністю залежить від обсягу словника N-грам).

Рисунок 3.8 – Порядок обробки слів запиту, їх пошуку та оцінки релевантності результатів пошуку

Далі створюється список співпадінь цього токена з іншими унікальними токенами БД. Довжина цього списка дорівнює числу усіх товарів з БД, при чому початкові значення по кожному порівнянню проставляються як 0. Тепер необхідно зробити оцінку подібності слів методом оцінки відстані метрикою Левенштейна. Однак до цієї версії було додано наступні модифікації:

– операція заміни дорівнює 0.33, якщо порівнюються дві голосні букви;

– операція заміни дорівнює 0.5, якщо порівнюються “б” та “п”, “в” та “ф”, “д” та “т”, “м” та “н”;

– операція перестановки сусідніх літер дорівнює 0.5 (алгоритм Дамерау).

Відбувається порівняння доповненим методом LD цього токена з усіма

елементами словника унікальних токенів, але в список співпадінь записується не число операцій по перетворенню одного слова в інше, а нормалізований варіант цієї оцінки (grade). Цей показник означає ступінь подібності, при якому 1 – повна ідентичність двох токенів та 0 – якщо немає жодного співпадіння та розраховується за формулою:

$$grade = 1 - \frac{distance}{max(len(user\_word),\ len(word\_in\_db)])}, \quad (3.1)$$

де distance – число операцій методу LD;

max(a, b) – функція, що обирає найбільше число з двох варіантів;

len(x) – функція розрахунку довжини строки;

user_word – токен в запиті користувача;

word_in_db – токен із словника токенів бази даних.

Обов'язково зустрінуться випадки, коли в описі товару багато слів і до кожного з них було розраховано оцінку LD. В цьому випадку обирається найбільша з оцінок. Описаний процес повторюється для кожного токену з запиту, а списки співпадінь запам'ятовуються для подальшої агрегації.

Коли всі слова отримали оцінки, необхідно створити загальний список співпадінь по всьому запиту користувача. Довжина цього списку також дорівнює числу усіх товарів з БД, а початкові значення дорівнюють 0. Тепер для кожного товару розраховується рейтинг (rating) по кількості та якості співпадінь за формулою:

$$rating = \sum_{m=0}^{n=words} \begin{cases} grade, \text{ якщо } grade \geq k \text{ та } len(user\_word) > 2 \\ \frac{grade}{\text{len(user\_sentence)}} * len(user\_word), \text{ якщо } grade < k' \end{cases} \quad (3.2)$$

де grade – ступінь подібності слів запиту до всіх слів БД;

n – кількість елементів сумування;

m – число, з якого треба почати сумувати;

words – кількість токенів бази даних;

k – пороговий коефіцієнт подібності;

len(x) – функція розрахунку довжини строки;

user_word – токен в запиті користувача;

user_sentence – весь текст запита користувача.

З формули (3.2) можна побачити, що загальний рейтинг релевантності запиту користувача до назви товару дорівнює сумі оцінок, з поправкою на умову гарного порівняння k та довжини слова. Тобто, якщо слово відрізнялося менше ніж на 1-k від порівнюваного, та його довжина від трьох літер, його релевантність необхідно залишити високою. В іншому випадку, ступінь подібності (grade) необхідно нормалізувати, тобто зменшити його оцінку на основі значущості цього слова в межах усього речення (відношення кількості букв слова до кількості букв речення).

Оптимальним значенням порогового коефіцієнта подібності було обрано 0,75, що дозволяє зменшити рейтинг товарам, слова в описі яких значно відрізнялися від шуканих, та збільшити рейтинг тих товарів, в назві яких було знайдено точну відповідність слів або з різницею не більше ніж 25%. Обмеження на короткі слова (менше 3 літер) зроблено заради того, щоб обмежувати вплив не значущих слів, таких як “з”, “зі”, “і”, “та”, “у”, та інші.

Після використання формули 3.2 до усіх оцінок слів, результати суми будуть в діапазоні [0, N], де N - кількість токенів в запиті. Наприклад запит “томат чорний принц” буде мати рейтинг 3.0 для товару, який називається “Томат Чорний принц, ваговий”, а запит “томат відбірний”, для цього ж товару, буде мати оцінку 1.28, оскільки “томат” дає максимальну оцінку (1.0), а рейтинг “відбірний” після застосування формули зменшується з 0.44 до 0.28.

Саме такі розрахунки дозволяють зменшити вірогідність потрапляння мало подібних слів до вершини рейтингу. Відсортувавши список результатів пошуку за рейтингом можна зробити відсікання не релевантних товарів.

Використовуючи поправку на перерахунок оцінки на основі відношення довжини слова до довжини усього запиту, сума усіх оцінок слів (за винятком оцінки, що вище k) буде дорівнювати менше ніж 0.75, тобто нижче ніж значення

порогового коефіцієнту подібності. Це означає, що усі слова, які не пройшли порогове значення можна прибрати і не відображати користувачу в результатах пошуку.

Готовий перелік товарів, можна відсортувати по найвищій оцінці та поділити по категоріях знайдених товарів, при чому категорії необхідно показувати в порядку убивання найвищої оцінки серед усіх товарів даної категорії. Таким чином користувачу, що шукав “томат чорний принц” буде показано спочатку усі товари з категорії “Овочі та фрукти” (в порядку найвищої оцінки серед даної категорії), а тільки потім інші категорії, де зустрічалось те чи інше слово з запиту.

У випадку, коли оцінки однакові (якщо в запиті було одне слово і є багато результатів з оцінкою 1), можна відсортувати однакові оцінки за довжиною строки, що дозволить товару з назвою “Імбир” потрапити вище в рейтингу ніж “Чай трав’яна Лавка імбир та чебрець”, під час пошуку слова “імбир”.

Код основної функції комбінованого методу текстового пошуку в е-комерції представлений на рис. 3.9.

```
184  def find_user_text(user_text, unique_tokens, items, en_ngram_dict):
185      all_matches_count = {i: 0 for i in range(len(items))}
186      tokens = tokenise(user_text)
187      # токенізація користувацького запиту
188      len_utext = len(''.join(tokens))
189      for token in tokens:
190          if token:
191              len_tok = len(token)
192              # перевірка наявності англійських слів методом N-грам
193              token = replace_en_word(token, en_ngram_dict, items)
194              matches_count = {i: 0 for i in range(len(items))}
195              for u_token in unique_tokens.keys():
196                  # використання модиціkованого метода відстані Левенштейна
197                  distance = levenstein(token.lower(), u_token.lower())
198                  grade = 1 - distance / max(len(token), len(u_token))
199                  for i in unique_tokens[u_token]:
200                      matches_count[i] = grade if matches_count[i] < grade else matches_count[i]
201              for match in matches_count:
202                  grade = matches_count[match]
203                  all_matches_count[match] += grade if grade >= k and len_tok > 2 else grade / len_utext * len_tok
204      # фільтрація та сортування результатів пошуку
205      find_items = list(filter(lambda x: x[1] >= k, list(all_matches_count.items())))
206      find_items.sort(key=lambda x: x[1], reverse=False)
207      return find_items
```

Рисунок 3.9 – Функція пошуку користувацького запиту

4 ЕКСПЕРИМЕНТАЛЬНА ПЕРЕВІРКА ЕФЕКТИВНОСТІ ВИКОРИСТАННЯ ЗАПРОПОНОВАНОГО КОМБІНОВАНОГО МЕТОДУ

4.1 Розробка пошукової системи на основі комбінованого методу

Для перевірки роботи комбінованого метода вживимо його в типовий застосунок ЕК “Добрі ціні”, заснований на клієнт-серверній архітектурі, де клієнтська частина має строку пошуку та відображає товари, надіслані сервером.

Агрегацію комбінованого метода було здійснено в вже існуючу функцію find(user_text). Принцип роботи цієї функції наступний: отримання текстового пошукового запиту користувача, передача функції пошуку текст запиту та заздалегідь сформовані списки описів товарів, словників унікальних слів та N-грам, отримання результатів рейтингу товарів, відправка клієнту сформованого переліку товарів.

За основу даних, що будуть оброблятись, взято інформацію про товари з веб-сайту магазину Сільпо. Кількісні характеристики обраної бази даних зазначено в табл. 4.1.

Таблиця 4.1 – Кількісні характеристики бази даних Сільпо

Характеристика	Значення	Одиниця вимірювання
Кількість товарів в БД	44140	шт.
Кількість унікальних токенів	28888	шт.
Триграми усіх англійських слів	7722	шт.
Сума довжин описів товарів БД	1551117	літ.
Сума довжин унікальних токенів	55293	літ.
Середня довжина опису товару	35.14	літ.
Середня довжина одного слова	7.46	літ.

Запустивши сервер та клієнт можна побачити інтерфейс застосунку та приклади введення типових запитів в пошукову строку, що можна побачити на рис. 4.1.

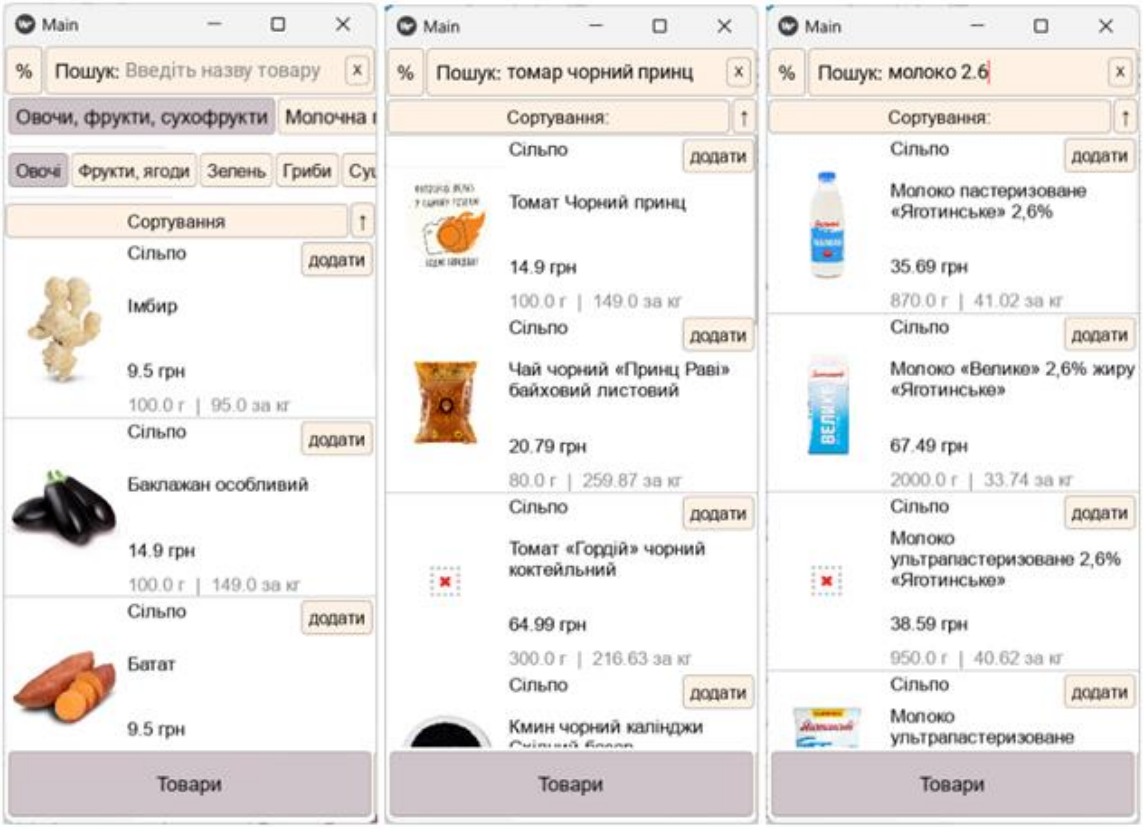

Рисунок 4.1 – Приклади роботи пошукової системи в застосунку

4.2 Порівняння ефективності роботи розробленої системи з пошуковою системою Сільпо

Для оцінки роботи комбінованого метода, необхідно порівняти його роботу з роботою іншої ПС, використовуючи якомога більше різних варіантів запитів, які можуть шукати користувачі, включаючи запити з помилками. Для порівняння було обрано ПС веб-сайту Сільпо.

Основні критерії порівняння: швидкість роботи ПС, наявність в

результатах пошуку необхідного товару, кількість знайдених товарів, актуальність інших товарів знайдених за запитом. В процесі порівняння було враховано наступні характеристики тексту запиту:

– короткі запити без помилок та з 1 помилкою (“рис”, “ріс”);

– запити середнього розміру без помилок, з двома помилками, з неправильною розкладкою (“картопля”, “катропня”, “rfhnjgkz”);

– запити великого розміру (6-7 слів);

– використання в запиті англійського тексту та транскрипції англійських слів (“дольче густо”, “dolce gusto”, “ рошен”, “roshen”, “якобс”);

– запити, з пропущеним пробілом, неправильним написанням слів з дефісом, різним формулюванням слів (“крембрюле” та “крем брюле” замість “крем-брюле”).

Усі зазначені приклади запитів та результати роботи ПС можна побачити в табл. 4.2.

Таблиця 4.2 – Порівняльний аналіз пошукових систем

Пошукова система	КМ	Сільпо	КМ	Сільпо	КМ	Сільпо	КМ	Сільпо
Пошуковий запит та назва необхідного товару	Час роботи, с		Чи наявний товар, що необхідно знайти, так/ні		Кількість знайдених товарів, шт		Актуальність інших пошук. результатів, гарна/середня/ погана	
“рис” - (Рис Хатинка довгий)	0.109	0.141	так	так	157	369	гарна	гарна
“ріс” - (Рис Хатинка довгий)	0.109	0.217	так	ні	160	260	гарна	погана
“картопля” - (Картопля біла)	0.112	0.134	так	так	267	276	гарна	гарна
“катропня” - (Картопля біла)	0.113	0.139	так	так	83	155	гарна	гарна
“rfhnjgkz” - (Картопля біла)	0.112	0.177	так	так	267	275	гарна	гарна
“томат рожевий” - (Томат рожевий)	0.206	0.377	так	так	630	9	середня	середня

Кінець таблиці 4.2

Пошукова система	КМ	Сільпо	КМ	Сільпо	КМ	Сільпо	КМ	Сільпо
Пошуковий запит та назва необхідного товару	Час роботи, с		Чи наявний товар, що необхідно знайти, так/ні		Кількість знайдених товарів, шт		Актуальність інших пошук. результатів, гарна/середня/ погана	
“шоколад рошен” - (Шоколад молочний Roshen)	0.205	0.436	так	так	1285	41	гарна	середня
“шокола droshen” - (Шоколад молочний Roshen)	0.204	0.438	так	ні	1093	1	гарна	погана
“шоколадroshen” - (Шоколад молочний Roshen)	0.127	0.833	ні	ні	0	0	погана	погана
“крембрюле” - (Десерт Крем-брюле)	0.160	0.199	так	ні	16	0	гарна	погана
“крем брюле” - (Десерт Крем-брюле)	0.215	0.235	так	так	747	17	гарна	середня
“кава мелена dolce gusto капучино 16 капсул” - (Кава мелена Dolce Gusto Ristretto смажена 16 капсул)	0.703	0.172	так	так	2136	2	середня	погана
“сухарікі флінт беконові” - (Сухарики Flint з беконом)	0.277	0.616	так	так	69	1	гарна	погана
“молоко 2 %” - (Молоко пастеризоване 2%)	0.309	0.537	так	так	328	205	середня	гарна
“згущене молоко” - (Молоко згущене Лавка традицій)	0.206	0.086	так	так	1374	35	середня	гарна
“якобс 3 в 1” - (Напій кавовий Jacobs 3в1)	0.441	0.185	так	так	40	14	гарна	середня
“ytcrfat 3d1” - (Напій кавовий Nescafe 3в1)	0.460	0.105	так	так	71	6	гарна	середня

Графік порівняння часу, витраченого на виконання пошуку між системою “Сільпо” та комбінованим методом, зображено на рис. 4.2.

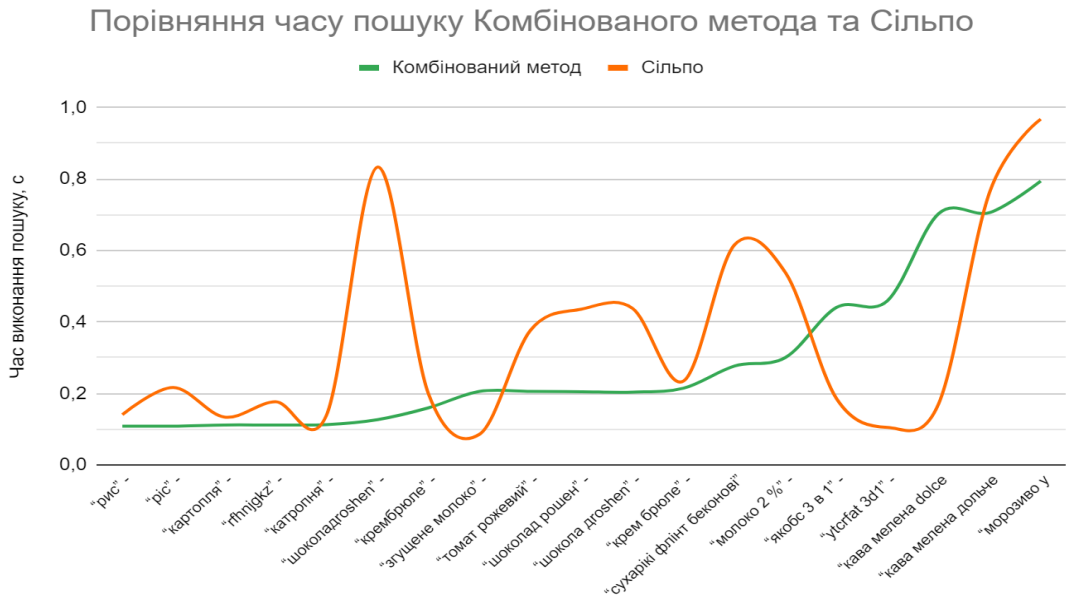

Рисунок 4.2 – Графік залежності часу пошуку від розміру словника та довжини запиту

Додатково було згенеровано 13 тестових вибірок БД різного розміру, кількість унікальних токенів яких варіюється від 100 до 100 000 шт, середній розмір токену складає 7 літер. На основі чотирьох запитів різної довжини було протестовано час роботи комбінованого метода для кожної тестової вибірки. Результати цього тесту можна побачити в табл. 4.3 а також на відповідному графіку на рис. 4.2.

Таблиця 4.3 – Залежність часу пошуку від кількості унікальних токенів

Кількість унікальних токенів, шт	Час пошуку запиту, с			
	“рис”	“картопля”	“томат рожевий”	“морозиво у вафельному стаканчику”
100	0,00099	0,00099	0,0026	0,004
500	0,0061	0,0039	0,01	0,019

Кінець таблиці 4.3

Кількість унікальних токенів, шт	Час пошуку запиту, с			
	“рис”	“картопля”	“томат рожевий”	“морозиво у вафельному стаканчику”
1000	0,011	0,0095	0,021	0,039
2500	0,028	0,025	0,052	0,098
5000	0,056	0,048	0,1	0,19
7500	0,086	0,074	0,15	0,29
10000	0,11	0,098	0,19	0,39
25000	0,27	0,25	0,56	1
50000	0,56	0,49	1,074	1,99
75000	0,84	0,75	1,59	3,047
100000	1,16	0,99	2,14	4,055

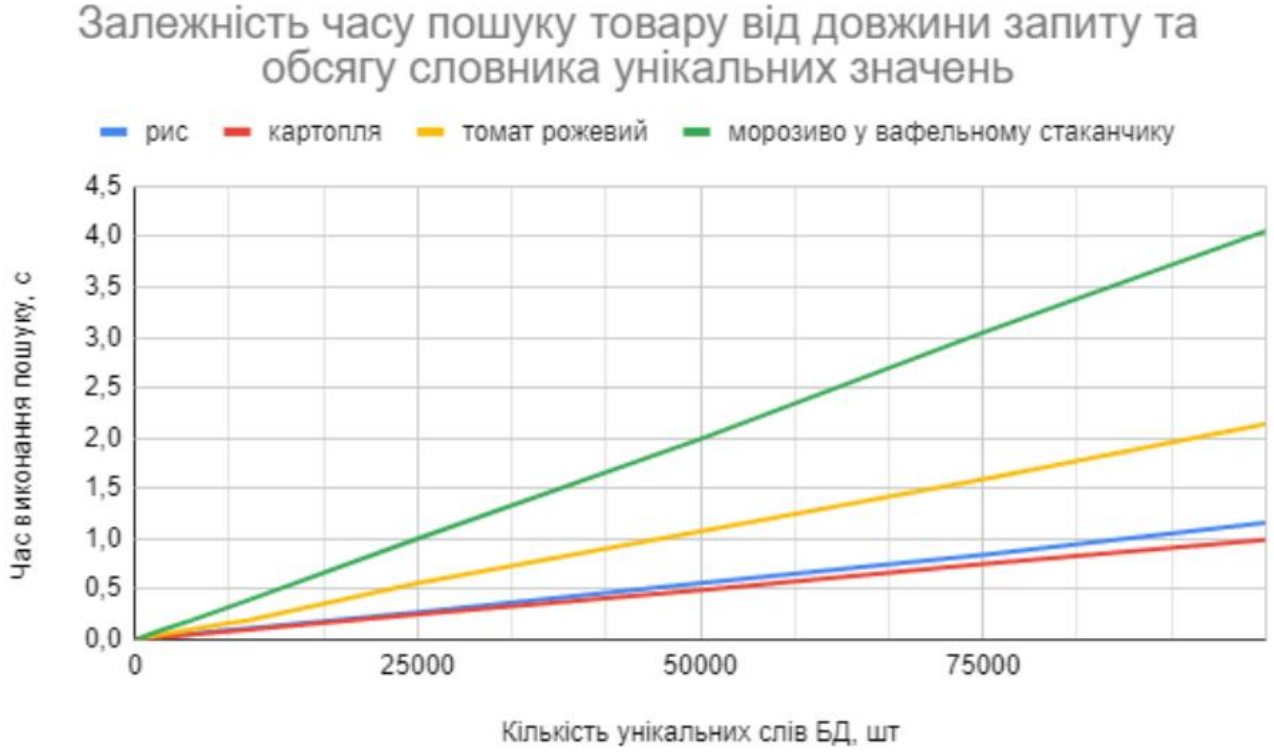

Рисунок 4.3 – Графік залежності часу пошуку від розміру словника та довжини запиту

4.3 Оцінка отриманих результатів

На основі табл. 4.2, що містить детальне порівняння двох ПС, КМ (Комбінований метод) та Сільпо, за різними пошуковими запитами, можна оцінити результати виконання пошуку за часом, наявністю товару, кількістю знайдених товарів та актуальністю інших пошукових результатів.

Оскільки для тестів була обрана база даних, що охоплює всі товари в веб-магазині Сільпо, можна припустити що розміри використаних БД однакові.

Час виконання пошуку.

Час пошуку системи Сільпо був визначений за допомогою браузерного інструмента розробника, на основі параметра “Час очікування відповіді від сервера” URL-запису “search”. Необхідно враховувати, що цей параметр включає не тільки безпосередньо пошук запита, а і час на пересилання запита, можливе очікування черги, його обробку, формування відповіді та надсилання клієнту.

У більшості випадків комбінований метод показав швидший час пошуку, ніж система Сільпо. З графіку на рис. 4.3 помітно, що швидкість пошуку “рис”, “картопля” та “шоколадroshen” майже однакова, тобто можна зробити висновок, що кількість символів в слові мало впливає на роботу алгоритму.

Натомість кількість окремих токенів в запиті безпосередньо впливають на швидкість. Можна побачити, що “шоколадroshen” було опрацьовано КМ в півтора рази швидше, ніж “шоколад рошен”. А запит “ytcrfat 3d1”, що візуально має 2 токени, насправді поділяється алгоритмом на 4 токена та кожен проходить по всім словам словаря, що збільшує час пошуку майже в 4 рази порівняно з пошуком “шоколадroshen”.

Також варто зазначити, що система Сільпо більш швидко знаходить запити, що точно відповідають тому, як написано в описі товару, і витрачає більше часу на опрацювання слів, що мають інше формулювання. В той час як КМ не враховує точну відповідність слів, і має перевагу в часі для запитів де дещо інше закінчення, невеликі помилки, англійські слова написані українською.

Наявність товару.

Комбінований метод частіше знаходив необхідний товар порівняно з системою Сільпо. Наприклад, для запиту "шокола droshen" Сільпо нічого не знайшов, тоді як КМ - так, бо токени запиту мають різницю 1 до оригінальних слів, що надає їм перевагу. Така ж ситуація спостерігається для запиту "крембрюле", де товар було знайдено в КМ, але не в Сільпо. Такі результати

Кількість знайдених товарів та їх актуальність.

Комбінований метод також демонструє перевагу в кількості знайдених товарів. Однак, в одних ситуаціях велика кількість знайдених товарів може бути корисною, в інших ні. Наприклад для запита “сухарікі флінт беконові” КМ знаходить усі товари де в назві є “Сухарики”, “Flint” та переважно снеки зі смаком бекону. Це може надати користувачу додатковий простір для вибору, що позитивно впливає на реалізацію товарів. Натомість для запиту “згущене молоко” було знайдено не тільки актуальні товари, а і усі молочні товари.

КМ переважно показував кращу або середню актуальність результатів, тоді як Сільпо частіше показував середню або погану актуальність. Наприклад, для запиту “ріс”, Сільпо не знайшов жодного товару, де в назві є “рис”, натомість видав усі товари де є частка “ріс”. В той час КМ гарно розпізнав помилку і першими результатами пошуку були переважно товари з категорії Бакалія.

4.4 Актуальність впровадження розробленої системи та можливості її вдосконалення

Комбінований метод пошуку є особливо актуальним для магазинів з невеликим обсягом бази даних, оскільки забезпечує швидку та ефективну обробку запитів. В умовах, коли база даних містить обмежену кількість товарів, лінійна залежність часу пошуку від розміру бази даних не створює суттєвих затримок. Це означає, що навіть при збільшенні кількості товарів у базі даних,

час обробки пошукових запитів залишається прийнятним, що сприяє підвищенню задоволеності користувачів та ефективності роботи магазину.

Розроблений КМ також є оптимальним для магазинів з помірним рівнем активності користувачів ПС (до 10 запитів в секунду для системи з розміром словника токенів N = 20 000). В таких умовах система може швидко реагувати на запити, забезпечуючи користувачам актуальну та точну інформацію про наявність товарів. Це дозволяє підтримувати високий рівень обслуговування клієнтів, не перевантажуючи систему і не створюючи затримок у обробці запитів.

Метод також дозволяє легко адаптувати систему до різних умов і вимог. Наприклад, він може бути налаштований для різних типів магазинів і баз даних, забезпечуючи при цьому високу ефективність і точність пошуку.

КМ має значну актуальність завдяки своїй здатності забезпечувати високу точність пошуку, гнучкість і зручність для користувачів, проте існують можливості для його подальшого вдосконалення.

За аналогією роботи ПС Сільпо, можна впровадити попередній точний пошук по словам, щоб під час співпадіння токенів із словником вже не застосовувати метод LD і зберегти час.

Впровадження функції автоматичного перекладу користувацьких запитів дозволить розширити можливості пошуку. Наприклад, якщо користувач введе слово "ежевика", система автоматично зможе знайти товари під назвою "ожина". Це значно підвищить зручність користування системою для багатомовних користувачів.

Якщо переписати алгоритм Левенштейна мовою програмування C++, він буде працювати швидше, ніж на Python, через те, що C++ є компільованою мовою, яка перетворює код у високоефективний машинний код, застосовує оптимізації на рівні компілятора, забезпечує низькорівневий контроль над пам'яттю і використовує статичну типізацію, що разом значно підвищить продуктивність і швидкість ПС.

Інтеграція методу тезауруса дозволить асоціювати контекст запиту з відповідною категорією та підкатегорією і надавати більш актуальні результати.

Це особливо корисно для запитів, що містять синоніми або різні варіанти назв одного і того ж товару.

Впровадження методів контекстного аналізу дозволить системі краще розуміти запити користувачів. Аналіз семантичної близькості, заснований на використанні векторних моделей слів, таких як Word2Vec або GloVe, дозволить системі оцінювати схожість між словами і фразами. Наприклад, якщо користувач шукає "шоколад з горіхами", система зможе розпізнати, що "шоколад з мигдалем" є релевантним результатом.

Отже, існує ще багато шляхів поліпшення розробленого комбінованого метода пошуку текстової інформації. Але навіть у нинішньому вигляді він ефективно справляється із завданням пошуку, забезпечуючи високу точність і швидкість обробки запитів.

ВИСНОВКИ

На основі виконаних досліджень та аналізу методів нечіткого пошуку текстової інформації було розроблено удосконалений комбінований метод (КМ), що поєднує алгоритм оцінки відстані Левенштейна, метод N-грам та фонетичний метод транскрипції. Розроблений КМ дозволяє ефективно розпізнавати та обробляти запити користувачів, враховуючи орфографічні помилки та багатомовність. Проведене порівняння з пошуковою системою Сільпо показало, що КМ забезпечує більш швидкий час пошуку, особливо для запитів з помилками та багатомовними запитами.

Крім того, КМ продемонстрував кращу актуальність і релевантність результатів пошуку. Це досягається завдяки оптимізації алгоритму, який не лише враховує точну відповідність слів, але й здатний ефективно працювати з запитами, що мають різні закінчення, невеликі помилки та англійські слова, написані українською. Це робить КМ надзвичайно актуальним у сучасних умовах, коли користувачі очікують високої точності та швидкості пошуку.

Розроблений комбінований метод пошуку може бути впроваджений у системах електронної комерції, особливо в інтернет-магазинах з невеликим та помірним обсягом баз даних. Завдяки своїй здатності швидко і точно обробляти запити, метод забезпечує покращення користувацького досвіду, що сприяє збільшенню продажів та конкурентоспроможності магазинів. КМ можна легко адаптувати до різних типів баз даних та специфічних умов роботи інтернет-магазинів.

Крім того, КМ може бути використаний у інших інформаційних системах, де важлива точність і швидкість пошуку текстової інформації. Наприклад, він може бути інтегрований у системи управління контентом, бібліотечні каталоги, медичні інформаційні системи та інші сфери, де необхідний швидкий доступ до текстових даних. У цих галузях КМ може значно підвищити ефективність роботи систем, забезпечуючи користувачам точні та релевантні результати пошуку.

Наукова новизна роботи полягає в інтеграції та оптимізації методів нечіткого пошуку для підвищення продуктивності та точності пошукових систем у електронній комерції. Це дослідження дозволяє вирішувати проблеми пошуку в умовах наявності орфографічних помилок та багатомовності, що має значний вплив на підвищення ефективності інформаційних систем. Використання таких методів дозволяє значно покращити користувацький досвід, зменшуючи час пошуку та підвищуючи точність результатів, що є важливим фактором для сучасних інформаційних систем.

Соціально-економічна значущість роботи полягає в покращенні доступності товарів для користувачів, що сприяє збільшенню продажів та покращенню загального рівня обслуговування клієнтів. Удосконалення пошукових систем дозволяє користувачам швидше знаходити потрібні товари, що підвищує їх задоволеність і лояльність до платформи. Це, в свою чергу, сприяє зростанню прибутковості компаній, що займаються електронною комерцією, та підвищує їх конкурентоспроможність на ринку.

Продовження досліджень у напрямку вдосконалення методів нечіткого пошуку є доцільним, оскільки існує потенціал для подальшої оптимізації алгоритмів та підвищення їх ефективності. Подальші дослідження можуть включати інтеграцію додаткових методів контекстного аналізу та машинного навчання, що може значно підвищити адаптивність та точність пошукових систем, роблячи їх більш ефективними у розпізнаванні складних запитів.

Також варто дослідити можливості масштабування розробленого методу для застосування в умовах великих баз даних та високого навантаження на систему. Це дозволить адаптувати КМ для використання у великих інтернет-магазинах та інших системах з високим рівнем активності користувачів. Продовження досліджень у цьому напрямку може значно розширити сфери застосування розробленого методу, забезпечуючи його ефективність та надійність у різних умовах експлуатації.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. ДСТУ 3008:2015. Інформація та документація. Звіти у сфері науки і техніки. Структура і правила оформлювання. – Чинний від 22.06.2015. – Київ: ДП «УкрНДНЦ», 2016. 1 – 31 с.

2. ДСТУ 8302:2015. Інформація та документація. Бібліографічні посилання. Загальні положення та правила складання. – Чинний від 04.03.2016. – Київ: ДП «УкрНДНЦ», 2016. 1 – 20 с.

3. 7 Юзабіліті-помилок у ecommerce, які заважають купити. URL: https://www.promodo.ua/blog/7-yuzabiliti-pomilok-u-ecommerce-yaki-zavazhayut-kupiti (дата звернення: 10.02.2024).

4. Пошук за допомогою пошукової системи. URL: https://owl.purdue.edu/owl/research_and_citation/conducting_research/searching_online/searching_with_a_search_engine.html (дата звернення: 10.02.2024).

5. Де ви втрачаєте покупців та «зливаєте» продажі: реальні ситуації з інтернет-магазинів. URL: https://mc.today/blogs/sezon-rasprodazh-vyderzhit-li-poisk-v-vashem-internet-magazine/ (дата звернення: 12.04.2024).

6. Manning C. D., Raghavan P., Schütze H. Introduction to information retrieval. Cambridge University Press, 2008. 24 – 28 с.

7. Kleshch K., Shablii V. Comparison of fuzzy search algorithms based on Damerau-Levenshtein automata on large data. Technology audit and production reserves, 2023. 27 – 32 с.

8. Saumya S., Amrita, B., Parth A., Ankit M. N-Gram fuzzy keyword search on encrypted user data in cloud. International Journal of Open Information Technologies, 2017. 21 – 26 с.

9. Bigram, Trigram, and NGram Models in NLP. URL: https://www.exploredatabase.com/2020/04/bigram-trigram-and-ngram-language-model-in-nlp.html (дата звернення: 15.04.2024).

10. Pinto D., Vilarino D., Alemán Y., Gómez H., Loya N. The soundex phonetic

algorithm revisited for sms-based information retrieval. In II Spanish conference on information retrieval CERI, 2012. 15 – 19 с.

11. Vijayarani S., Janani R. Text mining: open source tokenization tools-an analysis. Advanced Computational Intelligence (ACII), 2016. 37 – 47 с.

12. Глибовець А., Точицький В. Алгоритм токенізації та стемінгу для текстів українською мовою. 2017. 1 – 5 с.

13. Бісікало О. В. Асоціативний пошук завдань навчання з урахуванням електронного тезаурусу образів. 2009. 28 – 33 с.

14. Григорова Т. А., Ляшенко В. П., Москаленко О. О. Дослідження методів машинного навчання для пошуку інформації. Прикладні питання математичного моделювання. Кременчук, 2021. 2 – 9 с.

15. Семенець К.Є. Дослідження методів пошуку текстової інформації та їх використання в іт-проєктах електронної комерції / К. Є. Семенець; наук. керівник д.т.н., проф. Петров К.Е. // Радіоелектроніка та молодь у XXI столітті : матеріали 27-го Міжнар. молодіж. форуму, 10–12 травня 2023 р. – Харків : ХНУРЕ, 2023. – Т. 6, ч.1. – С. 240–241.

16. Luo X., Yang Y., Zhu, K. Q., Gong Y., Yang K. Conceptualize and infer user needs in e-commerce. In Proceedings of the 28th ACM International Conference on Information and Knowledge Management, 2019. 2517 – 2525 с.

17. Guarda T., Augusto M., Silva C. Competitive advantage in e-commerce: the case of database marketing. In Business, Economics, Financial Sciences, and Management. 2012. 123 – 130 с.

18. Що таке коефіцієнт конверсії Amazon? URL: https://www.sellerassistant.app/ru/blog/what-is-the-amazon-conversion-rate. (дата звернення: 02.05.2024).

19. Релевантні запити: що це таке, як розрахувати та підвищити релевантність сторінки. URL: https://wezom.com.ua/ua/blog/scho-take-relevantni-poshukovi-zapiti (дата звернення: 02.05.2024).

20. Духновська К. К., Страшок Я. А., Шило П. В. Інформаційна технологія для проведення лематизації і стемінгу в україномовних текстах. Прикладні

системи та технології в інформаційному суспільстві, 2022. 119 – 127 с.

21. Методичні вказівки щодо розробки та оформлення кваліфікаційної роботи другого (магістерського) рівня вищої освіти за освітньо-науковою програмою «Управління проектами в галузі інформаційних технологій» / Упоряд.: Петров К.Е., Левикін В.М., Чалий С.Ф., Євланов М.В., Міхнов Д.К., Міхнова А.В., Чала О.В. – Харків: ХНУРЕ, 2024. – 24 с.