

**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»**

Факультет інформатики та обчислювальної техніки

Кафедра інформатики та програмної інженерії

«На правах рукопису»
УДК 004.852

«До захисту допущено»
В.о. завідувача кафедри
_____ Едуард ЖАРИКОВ
«__» _____ 2021 р.

Магістерська дисертація

на здобуття ступеня магістра

**за освітньо-професійною програмою «Інженерія програмного
забезпечення інформаційних систем»**

зі спеціальності 121 «Інженерія програмного забезпечення»

**на тему: «Програмне та математичне забезпечення для генерації
діалогових систем з підтримкою української мови»**

Виконала:

студентка II курсу, групи ІІІ-з01мп
Корольова Людмила Вікторівна _____

Керівник:

Старший викладач кафедри ІІІ
Халус Олена Андріївна _____

Рецензент:

доцент кафедри ІСТ, к.т.н., доц.,
Писаренко Андрій Володимирович _____

Засвідчую, що у цій магістерській
дисертації немає запозичень з праць
інших авторів без відповідних посилань.
Студентка _____

Київ – 2021 року

**Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»**

Факультет інформатики та обчислювальної техніки

Кафедра інформатики та програмної інженерії

Рівень вищої освіти – другий (магістерський)

Спеціальність – 121 «Інженерія програмного забезпечення»

Освітньо-професійна програма «Інженерія програмного забезпечення інформаційних систем»

ЗАТВЕРДЖУЮ

В.о. завідувача кафедри

_____ Едуард ЖАРІКОВ

«___» _____ 2021р.

**ЗАВДАННЯ
на магістерську дисертацію студенту
Корольовій Людмилі Вікторівні**

1. Тема дисертації «Програмне та математичне забезпечення для генерації діалогових систем з підтримкою української мови», науковий керівник дисертації Халус Олена Андріївна, старший викладач кафедри ІПП, затверджені наказом по університету від «25» жовтня 2021 р. № 3575-с
2. Термін подання студентом дисертації «6» грудня 2021 р.
3. Об'єкт дослідження – засоби розробки для обробки природної мови, пристосовані до української мови.
4. Предмет дослідження – програмне та математичне забезпечення для генерації діалогових систем, що пристосовані до української мови.
5. Перелік завдань, які потрібно розробити – виконати аналітичний огляд існуючих наукових робіт, які направлені на реалізацію програмного та математичного забезпечення генерації діалогових систем; розробка алгоритму підготовки даних для їх аналізу моделлю; пристосування алгоритму кластеризації до аналізу природної мови; розробка алгоритму перетворення проаналізованих даних у речення; програмна реалізація розроблених алгоритмів.
6. Орієнтовний перелік графічного (ілюстративного) матеріалу – схема бази даних, UML діаграма класів.
7. Орієнтовний перелік публікацій – SoftTech-2021.

8. Консультанти розділів дисертації

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

9. Дата видачі завдання «30» вересня 2020 р.

Календарний план

№ з/п	Назва етапів виконання магістерської дисертації	Термін виконання	Примітка
1	Пошук та аналіз аналогів	7.10.2021	
2	Аналіз існуючих програмних засобів	14.10.2021	
3	Розробка моделі	21.10.2021	
4	Адаптація моделі	28.10.2021	
5	Програмна реалізація	1.11.2021	
6	Тестування	1.11.2021	
7	Виконання експериментальних досліджень	8.11.2021	
8	Оформлення пояснювальної записки	15.11.2021	
9	Подання дисертації на попередній захист	22.11.2021	
10	Подання дисертації на захист	6.12.2021	

Студент

Людмила КОРОЛЬОВА

Науковий керівник

Олена ХАЛУС

РЕФЕРАТ

Розмір пояснювальної записки – 98 аркушів, містить 15 ілюстрацій, 53 таблиці, 4 додатків.

Актуальність теми. У роботі розглянуто проблему відсутності розвинених систем генерації діалогів українською мовою, показано основні особливості існуючих рішень для генерації діалогів, їх переваги та недоліки. Виявлено потребу в розробці програмного забезпечення пристосованого до української мови.

Мета дослідження. Основною метою є розширення функціоналу простих моделей генерації діалогів для того щоб покривати більший перелік запитів користувачів та зробити діалог більш природним

Об'єкт дослідження: засоби розробки для обробки природної мови, пристосовані до української мови.

Предмет дослідження: програмне та математичне забезпечення для генерації діалогових систем, що пристосовані до української мови.

Для реалізації поставленої мети **сформульовані наступні завдання:**

- виконати аналітичний огляд існуючих наукових робіт, які направлені на реалізацію програмного та математичного забезпечення генерації діалогових систем;
- розробка алгоритму підготовки даних для їх аналізу моделлю;
- пристосування алгоритму кластеризації до аналізу природної мови;
- розробка алгоритму перетворення проаналізованих даних у речення;
- програмна реалізація розроблених алгоритмів.

Наукова новизна результатів магістерської дисертації полягає в тому, що запропоновано алгоритм для генерації діалогів українською мовою, що використовує остові дерева як основу та кластеризацію як функцію оцінки результатів.

Результат досягнутий шляхом розробки модернізованого алгоритму з адаптацією під роботу з базою даних.

Практичне значення отриманих результатів полягає в тому, що реалізовано алгоритм пристосований до української мови який дозволяє генерувати діалог на різні теми. Також було розроблено програмне забезпечення яке дозволяє використовувати данні з бази даних при генерації діалогу. Дане програмне забезпечення може бути використане розробниками чат-ботів у різних сферах.

Зв'язок з науковими програмами, планами, темами. Робота виконувалась на кафедрі інформатики та програмної інженерії Національного технічного університету України "Київський політехнічний інститут імені Ігоря Сікорського".

Апробація. Наукові положення дисертації пройшли апробацію на всеукраїнській науково-практичній конференції молодих вчених та студентів «Інженерія програмного забезпечення і передові інформаційні технології» (SoftTech2021) – м. Київ.

Публікації. Наукові положення дисертації опубліковані в:

Корольова Л.В. Алгоритм навчання діалогової системи з використанням остових дерев / Л.В. Корольова, О.А. Халус // Матеріали всеукраїнської науково-практичної конференції молодих вчених та студентів «Інженерія програмного забезпечення і передові інформаційні технології» (SoftTech2021) – м. Київ: НТУУ «КПІ ім. Ігоря Сікорського», 2021 р.

Ключові слова: КЛАСТЕРИЗАЦІЯ, ОБРОБКА ПРИРОДНОЇ МОВИ, ДІАЛОГ, ОСТОВІ ДЕРЕВА

ABSTRACT

Explanatory note size – 98 pages, contains 15 illustrations, 53 tables, 4 applications.

Topicality. Examines the problem of the lack of developed dialogue generation systems in the Ukrainian language, shows the main features of existing solutions for dialogue generation, their advantages and disadvantages. The need for the development of software adapted to the Ukrainian language has been identified.

The aim of the study. The main target is to extend the functionality of simple models for generating dialogs in order to cover a larger list of user requests and make the dialogue more natural.

Object of research: development tools for natural language processing, adapted to the Ukrainian language.

Subject of research: software and mathematical software for generating dialog systems adapted to the Ukrainian language.

To achieve this goal, the **following tasks** were formulated:

- perform an analytical review of existing research papers aimed at implementing software and mathematical software for generating dialog systems;
- development of algorithm of data preparation for their analysis by model;
- adaptation of the clustering algorithm to the analysis of natural language;
- development of an algorithm for converting the analyzed data into sentences;
- software implementation of the developed algorithms.

The scientific novelty of the results of the master's dissertation is that an algorithm for generating dialogues in the Ukrainian language is proposed, which uses skeleton trees as a basis and clustering as a function of evaluating the results.

The result was achieved by developing an upgraded algorithm with adaptation to work with the database.

The practical value of the obtained results is that the implemented algorithm is adapted to the Ukrainian language which allows to generate a dialogue

on various topics. Software has also been developed that allows the use of database data when generating a dialog. This software can be used by chatbot developers in various fields.

Relationship with working with scientific programs, plans, topics. Work was performed at the Department of Informatics and Software Engineering of the National Technical University of Ukraine «Kyiv Polytechnic Institute. Igor Sikorsky».

Approbation. The scientific provisions of the dissertation were tested at the All-Ukrainian scientific-practical conference of young scientists and students "Software Engineering and Advanced Information Technologies" (SoftTech2021) - Kyiv.

Publications. The scientific provisions of the dissertation published in:

Korolova LV Algorithm for learning a dialog system using skeletal trees / L.V. Korolova, O.A. Khalus // Proceedings of the All-Ukrainian scientific-practical conference of young scientists and students "Software Engineering and Advanced Information Technologies" (SoftTech2021) - Kyiv: NTUU "KPI. Igor Sikorsky ", 2021

Keywords: CLUSTERIZATION, NATURAL LANGUAGE PROCESSING, DIALOGUE, TREE SCAFFOLDING

ЗМІСТ

ВСТУП	12
1 ЗАСОБИ ОБРОБКИ ПРИРОДНОЇ МОВИ	14
1.1 Опис предметної області	14
1.1.1 Синтаксичні задачі.....	14
1.1.2 Семантика.....	14
1.1.3 Дискурс.....	15
1.1.4 Мовлення.....	15
1.2 Аналіз існуючих рішень	16
1.2.1 Загальні комплексні моделі.....	16
1.2.2 Прості моделі	17
1.3 Аналіз існуючих технічних рішень.....	17
1.4 Аналіз існуючого програмного забезпечення	18
1.4.1 Морфологічний аналіз слова	18
1.4.2 Кластеризація	19
Висновки по розділу	19
2 МАТЕМАТИЧНЕ ЗАБЕЗПЕЧЕННЯ	20
2.1 Морфологічний розбір слова.....	20
2.2 Синтаксичний аналіз речення	20
2.3 Підготовка даних для моделі.....	21
2.4 Нечіткі множини	26
2.5 Кластеризація	27
2.5.1 K-means.....	27
2.5.2 Ієрархічна кластеризація	28
2.6 Використання остових дерев в моделі	30
2.7 Побудова граматично правильних речень.....	32
2.8 Функція оцінки результату.....	32

Висновки по розділу	33
3 РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ	34
3.1 Загальна архітектура.....	34
3.2 Архітектура рішення.....	35
3.2.1 Архітектура моделі	35
3.2.2 Архітектура адаптеру	36
3.2.3 Опис архітектури	37
Висновки по розділу	44
4 АНАЛІЗ ЕФЕКТИВНОСТІ	45
4.1 Підготовка даних	47
4.1.1 Розмір даних.....	47
4.1.2 Обробка даних.....	48
4.2 Модель.....	51
Висновки до розділу	51
5 МАРКЕТИНГОВИЙ АНАЛІЗ СТАРТАП-ПРОЄКТУ	53
5.1 Опис ідеї проекту.....	53
5.2 Формування команди стартапу	54
5.3 Морфологічна карта.....	57
5.4 Розроблення ринкової стратегії проекту	65
5.5 Розроблення маркетингової програми стартап-проекту.....	67
5.6 Розроблення маркетингової програми стартап-проекту.....	69
5.7 Виробничий план	74
5.8 Організаційний план.....	80
5.9 Оформлення авторського права на програму.....	84
Висновки по розділу	85
ВИСНОВКИ	86
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ.....	88

ДОДАТКИ	91
ДОДАТОК А	91
ДОДАТОК Б	92
ДОДАТОК В	93
ДОДАТОК Г	104

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

Користувач – людина, що зайшла в інтернет-магазин та хоче щось запитати у онлайн-консультанта;

Співробітник – людина що від імені магазину консультує покупців як онлайн-консультант;

Модель – програмне та математичне забезпечення що містять у собі данні для навчання, алгоритм навчання та алгоритм визначення цільової функції відповідно до призначення моделі;

Fit – обробка вхідних даних перед навчанням моделі;

Train – навчання моделі на основі наданих даних;

БД – база даних;

API (application programming interface) – програмний інтерфейс.

ВСТУП

У сучасному світі все більшого розвитку набувають технології машинного навчання. Вони вже використовуються у абсолютно різних галузях та непомітно покращують наше життя. Серед галузей, в яких застосовується машинне навчання можна виокремити таку велику та комплексну тему як обробка природної мови.

Для кожної галузі машинного навчання застосовуються різні алгоритми та необхідні різні набори даних. Щодо галузі обробки природної мови, для різних мов необхідні різні дані та моделі, оскільки для різних сімейств мов по-різному формуються речення та слова. Також свої особливості є у кожній окремій мові.

На сьогоднішній день найбільш розвиненою є обробка англійської мови, також високорозвиненою є обробка російської мови. У меншій мірі розвиненою є обробка української мови. [1]

З іншого боку багато людей користуються інтернетом для пошуку та надання послуг. Часто на таких сайтах існує функція онлайн-чату. В багатьох випадках це окремий працівник, який відповідає на всі виникаючі питання. Доволі часто ці питання прості та повторювані, тому на них можна було б відповідати автоматично. Проте це і не кілька питань які можна виписати завчасно та відавати заготовлену відповідь.

Однак покупців може відштовхувати механічне спілкування, у якому користувач серед переліку варіантів обирає необхідний. В такій ситуації бажано знайти баланс між живим спілкуванням, зручним та комфортним для людини та автоматизованими відповідями які потребують менше людської уваги.

Метою роботи є розширення функціоналу простих моделей генерації діалогів для того щоб покривати більший перелік запитів користувачів та зробити діалог більш природним. Для досягнення мети необхідно вирішити наступні задачі:

- виконати аналітичний огляд існуючих наукових робіт, які направлені на реалізацію програмного та математичного забезпечення генерації діалогових систем;
- розробка алгоритму підготовки даних для їх аналізу моделлю;
- пристосування алгоритму кластеризації до аналізу природної мови;
- розробка алгоритму перетворення проаналізованих даних у речення;
- програмна реалізація розроблених алгоритмів.

Створена модель повинна відповідати наступним вимогам:

Функціональні вимоги:

- формування граматично правильних речень;
- «запам'ятовування» контексту впродовж діалогу;
- формування стилістично та лексично природних речень;
- можливість пристосування для різних тем.

Нефункціональні вимоги:

- швидкість роботи;
- простота у використанні.

Об'єктом дослідження є засоби розробки для обробки природної мови, пристосовані до української мови.

Предметом дослідження є програмне та математичне забезпечення для генерації діалогових систем, що пристосовані до української мови.

1 ЗАСОБИ ОБРОБКИ ПРИРОДНОЇ МОВИ

1.1 Опис предметної області

У галузі обробки природної мови існує декілька основних комплексних задач. Хоча всі вони споріднені один з одним, ми розглянемо їх види.

1.1.1 Синтаксичні задачі

Типи синтаксичних завдань:

- граматична індукція: створення формальної граматики, яка описує синтаксис мови;
- лематизація: видалення лише флективних закінчень та знаходження основної словникової форми слова, яка також відома як лема;
- розбір: визначення дерева розбору (граматичний аналіз) даного речення;
- порушення речення: знаходження меж речення, маючи фрагмент тексту.

Межі речень часто позначаються крапками або іншими розділовими знаками, але ці самі символи можуть служити іншим цілям.

- та ін.

1.1.2 Семантика

Типи семантичних завдань:

- лексична семантика: визначення значення окремих слів у контексті;
- машинний переклад: автоматичний переклад тексту;
- породження природної мови: перетворення інформації з комп'ютерних баз даних або семантичних структур у зрозумілу людську мову;
- оптичне розпізнавання символів (OCR): за зображенням, що представляє друкований текст, визначення відповідного тексту;

- та ін.

1.1.3 Дискурс

Автоматичне підведення підсумків – створення резюме фрагмента тексту. Часто використовується для передачі короткого змісту тексту відомого типу, наприклад статей у фінансовому розділі газети.

Розв’язання кореферентності. За реченням або більшим фрагментом тексту визначення, які слова неоднозначні слова відносяться до яких сутностей. Розв’язування анафори є конкретним прикладом цього завдання, яке спеціально стосується зіставлення займенників з іменниками чи іменами, до яких вони відносяться. Більш загальне завдання розділення кореференцій також включає визначення так званих «перемикаючих зв’язків», що включають посилальні вирази. Наприклад, у реченні, як-от «Він увійшов до дому Джона через вхідні двері», «вхідні двері» є виразом посилення, а зв’язок, який слід визначити, полягає в тому, що двері, про які йдеться, є вхідними дверима будинку Джона. будинок (а не якусь іншу споруду, на яку також можна згадати).

Аналіз дискурсу. Ця рубрика включає ряд пов’язаних завдань. Одне із завдань – визначити структуру дискурсу зв’язного тексту, тобто характер дискурсивних зв’язків між реченнями (наприклад, розробка, пояснення, контраст). Іншим можливим завданням є розпізнавання та класифікація мовленнєвих актів у фрагменті тексту (наприклад, так-ні запитання, змістове запитання, твердження тощо).

1.1.4 Мовлення

Типи задач:

- розпізнавання мови: визначення текстового представлення промови за аудіозаписом, на якому говорить людина або люди;

- сегментація мовлення: розділення аудіозапису людської мови на слова;
- синтез мовлення: створення аудіозапису людської мови на основі тексту;
- та ін.

1.2 Аналіз існуючих рішень

В загальному існуючі рішення можна поділити на 2 типи: комплексні багатоцільові моделі та прості моделі.

1.2.1 Загальні комплексні моделі

Існують такі моделі як GPT3 та ruGPT-3, які дозволяють:

- генерувати тексти;
- відповідати на питання;
- семантичний аналіз;
- семантичний пошук;
- та ін.

Не всі ці функції необхідні для генерації діалогів, проте зручно що в одній моделі знаходиться багато інструментів для роботи з природною мовою.

Переваги:

- мають широкий спектр можливостей для застосування;
- навчені на великому обсязі даних що дозволяє генерувати лексично різноманітні діалоги.

Недоліки:

- на розробку моделей пішло багато років;
- не існує моделі для української мови.

1.2.2 Прості моделі

Частіше в інтернет-магазинах використовують прості моделі які містять у собі статичний перелік питань та відповідей і якщо користувач запитує щось, що не входить у цей перелік, вже необхідна допомога людини-оператора.

Переваги:

- прості у створенні та підтриманні;
- існують моделі, підтримуючі українську мову.

Недоліки:

- обмежені за функціоналом;
- однотипні питання та відповіді можуть відштовхнути користувача.

Постановка задачі

1.3 Аналіз існуючих технічних рішень

Рішення поставленої в цій роботі проблеми передбачає інтелектуальну обробку даних. Цей аналіз даних не є складним, тому немає необхідності використовувати неklasичні інструменти для автоматизації процесів.

В даний час вже існує достатня кількість бібліотек і фреймворків, які полегшують створення програмного забезпечення. І згодом їх кількість буде зростати. Що стосується аналізу даних, то класичними інструментами є MATLAB, R, Python, Octave. Ми використовували мову програмування Python.

MATLAB — це мультипарадигмальне обчислювальне середовище з власною мовою програмування, розробленою MathWorks. MATLAB дозволяє виконувати матричні маніпуляції, будувати функції та дані, реалізовувати алгоритми, створювати інтерфейси та взаємодіяти з програмами, написаними іншими мовами.

R — це безкоштовне програмне середовище для статистичних обчислень та графіки. Octave — це програмне забезпечення, що містить мову програмування

високого рівня, призначене в основному для чисельних, математичних розрахунків та аналізу даних. Octave допомагає чисельно розв'язувати лінійні та нелінійні задачі, а також проводить численні інші експерименти, використовуючи мову, переважно сумісну з MATLAB.

Octave має перевагу в тому, що він вільний.

Python — це мова програмування з багатьма бібліотеками, реалізованими для обробки даних. Особливістю Python є використання векторних обчислень, які значно прискорюють обробку даних. Для цієї програми були використані такі бібліотеки, як numpy, math і scikit-learn. Numpy — бібліотека для зручної роботи з даними, яка дозволяє зберігати їх у масиві та виконувати їх перетворення та статистичні обчислення за ними. Math — це бібліотека стандартних математичних функції (sin, cos, ln тощо) та констант (e, pi та ін.). Scikit-learn — це бібліотека для аналізу даних, як-от регресійні моделі або кластеризація.

Також в рамках цієї роботи необхідно використовувати збережені дані, а дані зазвичай зберігаються в базах даних. Реляційна база даних підходить для зберігання даних для інтернет-магазину. База даних була обрана серед систем з відкритим кодом, оскільки платні системи не мають принципових переваг для веб-додатків з продажу одягу. PostgreSQL є однією з найпопулярніших систем з відкритим кодом, оскільки вона більш гнучка з даними.

1.4 Аналіз існуючого програмного забезпечення

1.4.1 Морфологічний аналіз слова

Існують бібліотеки, які можуть проводити морфологічний аналіз слова, пристосовані до української мови. Наприклад rutmorphu2. Проте дана бібліотека аналізує слово без контексту, тому для більш точних результатів необхідно враховувати взаємозв'язки слів у реченні.

1.4.2 Кластеризація

Для обробки даних існує багато моделей реалізованих на різних мовах програмування. Наприклад для мови програмування Python існує бібліотека `scikit-learn`, а для Java – `Commons Math` та багато інших.

Висновки по розділу

У першому розділі проведено аналіз існуючих систем генерації діалогів та предметної області загалом, проаналізовані основні сучасні моделі та програмні рішення, які можна використати для їх створення, визначено переваги та недоліки цих рішень. В результаті проведеного аналізу сформульована постановка задачі, наведене призначення, цілі та задачі розробки.

2 МАТЕМАТИЧНЕ ЗАБЕЗПЕЧЕННЯ

2.1 Морфологічний розбір слова

Для морфологічного аналізу слова використовується той факт що у різних частин мови є різні особливості. Наприклад прикметники часто закінчуються на «ий», дієслово на «ить» і т. д. Таким чином аналізуючи префікс, суфікс та закінчення слова можна з деякою вірогідністю визначити до якої частини мови належить слово.[6]

Проте іноді з однієї лише форми слова важко сказати до якої частини мови воно належить. Для прикладу два речення:

- для виробництва медичних інструментів використовується багато сталі;
- виробництво має сталі показники якості продукту.

У першому випадку сталі – родовий відмінок слова сталь – сплав заліза з вуглецем. У другому випадку сталі – це слово сталий у множині – прикметник який характеризує предмет як той що зберігає свою форму, склад, зміст тощо.

У випадку такої неоднозначності слід враховувати не тільки статистичний аналіз структури слова, а й статистичний аналіз його місцезнаходження у реченні – чи є після спірного слова ще слова, чи є перед ним слова, якщо є, то до яких частин мови вони відносяться. Це хоча і не гарантує правильний результат, та допоможе зменшити вірогідність помилки.[7]

2.2 Синтаксичний аналіз речення

Перед тим як аналізувати слова, слід спочатку проаналізувати структуру речення, оскільки воно може бути як простим так і складним. Якщо речення складне, або містить дієприкметникові / дієприслівникові звороти, слід розбити його на прості частини і вже кожен з них аналізувати окремо. Крім того слід проаналізувати взаємозв'язок між цими складовим – це рівноправні частини речення, або одне лише

доповнює інше. Якщо виокремлена частина – це дієприкметникові / дієприслівникові зворот, то її можна прирівняти до прикметника / дієприслівника відповідно. Якщо частини є повноцінними реченнями то за допомогою сполучників можна виявити, складносурядне це речення чи складнопідрядне.

Зберігши всі ці структурні данні тексту можна сформулювати більш точний взаємозв'язок між словами та використати ці данні при побудові речень для діалогу.

2.3 Підготовка даних для моделі

Обробка вхідного тексту відбувається за наступним алгоритмом:

Пронумерувати кожен фрагмент тексту/абзацу (в залежності від формату тексту для навчання) – у кожного фрагменту має бути свій ідентифікатор.

Для кожного слова визначити частину мови, до якої воно належить – це допоможе виявити основні та другорядні члени речення.

Виокремити речення без займенників – аналіз речень з займенниками буде після побудови основних графів.

Побудувати направлений граф С (С - collocation), в якому кожна вершина представляє собою слово, а кожне ребро з вершини 1 у вершину 2 - кількість разів яку слово 2 зустрічалось після слова 1.

На сьогоднішній день багато людей
мають домашніх таврин. Вони
піднімають настрій, проте
накладають обов'язки на своїх
власників.

Рисунок 2.1 – Фрагмент тексту

Наприклад для фрагменту тексту рисунок 2.1 граф буде виглядати як на рисунку 2.2.

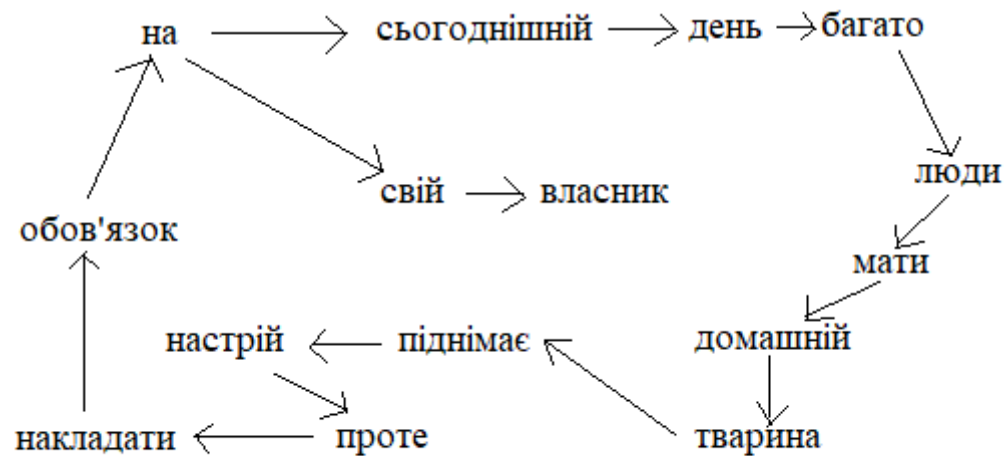


Рисунок 2.2 – Граф словосполучень

У кожному реченні визначити підмет та присудок. Підмет – це об'єкт про який надається інформація в реченні, а присудок – це його стан або дія яку виконує предмет, або яка виконується над предметом. Ці члени речення несуть основний зміст, другорядні – лише доповнюють інформацію. Для прикладу розбір речення на рисунку 2.3.

На сьогоднішній день багато людей
мають домашніх таврин. Вони
піднімають настрій, проте
накладають обов'язки на своїх
власників.

Рисунок 2.3 – Визначення підмета та присудка

Побудувати дводольний граф М (М - main members of the sentence), у якій вершини – це підмети та присудки, а ребра – кількість разів, яку зустрічалась така пара підмета та присудка (рисунок 2.4). Надалі цей граф буде використовуватися для заміни займенників на ті слова, що вони заміняли.

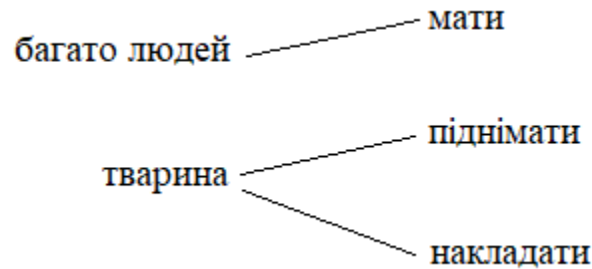


Рисунок 2.4 – Граф підметів і присудків

Побудувати неорієнтований граф Т (Т -text), у якому кожною вершиною є пара підмет і присудок, а ребра – це кількість разів яку ці пари зустрічались в одному невеликому тексті/абзаці (рисунок 2.5). Цей граф дозволить в подальшому визначати контекст та тему розмови.

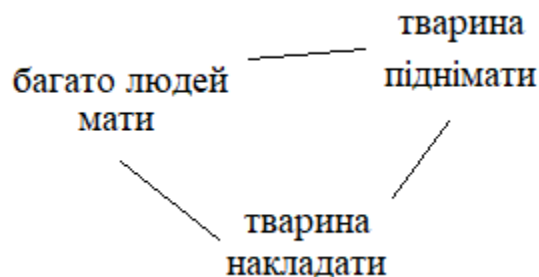


Рисунок 2.5 – Граф тексту

Побудувати дводольний граф S (S – secondary members of the sentence), у якому вершини – це пари підмет і присудок та другорядні члени речення а ребра – це кількість разів у яку зустрічалась така комбінація підмета і присудка та другорядних членів речення.

Наприклад для тексту рисунок 2.6 граф буде мати вигляд рисунку 2.7. Цей граф дозволить обирати характеристики предметів або дії враховуючи контекст.

На сьогоднішній день багато людей
мають домашніх тварин. Вони
піднімають настрій, проте
накладають обов'язки на своїх
власників.

Рисунок 2.6 – Повний розбір речення

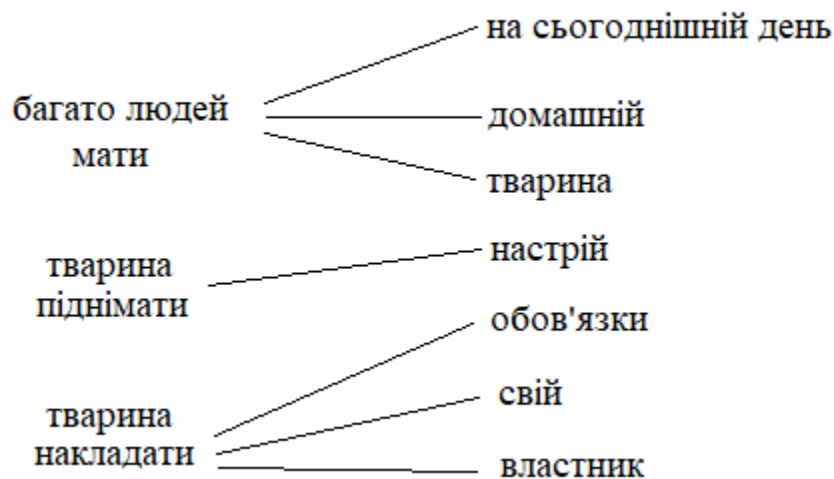


Рисунок 2.7 – Граф всіх членів речення

Для кожного речення з займенником:

- визначити це підмет чи додаток;
- визначити список іменників L які зустрічались в абзаці до цього речення;
- для знаходження найбільш підходящого слова використати теорему Баєса (детальніше в наступних кроках);
- якщо займенник виступає підметом, то серед списку іменників L обираємо той, для якого вірогідність P_2 є максимальною.

$$P_1(A|B) = \frac{P(B|A)*P(A)}{P(B)} \quad (2.1)$$

, де

A – підметом $\in l_i$,

B – присудком $\in j$

$$P_2(D|C) = \frac{P(C|D)*P(D)}{P(C)} \quad (2.2)$$

, де

D – підметом $\in l_i$ і присудком $\in j$,

C – другорядними членами речення $\in K$.

Тому, фінальне обчислення:

$$P_2(A|C) = \frac{P(C|D)*P_1}{P(C)} \quad (2.3)$$

Сформулюємо кожне з обчислень:

$$P(A) = \frac{1}{\sum l_i} \quad (2.4)$$

,де

$\sum l_i$ - кількість іменників у списку L.

$$P(B) = \frac{\sum_i m_{ij}}{\sum m_{ij}} \quad (2.5)$$

,де

$\sum_i m_{ij}$ – кількість разів яку зустрічався присудок j ,

$\sum m_{ij}$ – кількість разів яку взагалі зустрічались присудки.

$$P(C) = \prod \frac{s_k}{\sum s_k} \quad (2.6)$$

, де

s_k – кількість разів яку другорядний член речення k зустрівся з підметом l_i та присудком j в одному реченні,

$\sum s_k$ - кількість разів, яку взагалі зустрічався другорядний член речення k .

$$P(B|A) = \frac{m_{ij}}{\sum m_{ij}} \quad (2.7)$$

, де

m_{ij} – кількість разів яку зустрічалась пара підмет l_i та присудок j ,

$\sum m_{ij}$ – кількість разів яку взагалі зустрічались пари підмет і присудок.

$$P(C|D) = \prod P(C_k|A) \quad (2.8)$$

, де

$P(C_k|A)$ – вірогідність зустрічі другорядного члену речення C_k в реченні при умові, що підметом є l_i .

Слід зазначити, що дане формулювання обчислення виходять з гіпотези, що вірогідність зустрічі у реченні для кожного другорядного члена речення не залежить від іншого другорядного члена речення.

$$P(C_k|D) = \prod \frac{s_{ik}}{\sum_i s_{ik}} \quad (2.9)$$

, де

s_{ik} – кількість разів яку другорядний член речення k зустрівся з підметом l_i та присудком j в одному реченні,

$\sum_i s_{ik}$ – кількість разів, яку взагалі зустрічався другорядний член речення k .

Речення з заміненними займенниками використати для доповнення матриць М, Т, S.

2.4 Нечіткі множини

Нечітка множина - ключове поняття нечіткої логіки. Нехай E - універсальна множина, x - елемент E , а R - деяка властивість. Звичайна (чітка) підмножина A універсальної множини E , елементи якої задовольняють властивості R , визначається як множина упорядкованих пар

$$A = \{ \mu_A(x) / x \}$$

Де $\mu_A(x)$ — характеристична функція, що приймає значення 1, якщо x задовольняє властивість R , та 0 – якщо не задовольняє.

Нечітка підмножина відрізняється від звичайної тим, що для елементів $x \in E$ немає однозначної відповіді «так-ні» щодо властивості R . У зв'язку з цим нечітка підмножина A універсальної множини E визначається як множина упорядкованих пар

$$A = \{ \mu_A(x) / x \}$$

Де $\mu_A(x)$ — характеристична функція приналежності (або просто функція приналежності), що приймає значення в певній цілком упорядкованій множині M (наприклад, $M = [0, 1]$).

Функція приналежності вказує ступінь (або рівень) приналежності елемента x підмножини A . Множину M називають множиною належності. Якщо $M = \{0, 1\}$, то нечітка підмножина A може розглядатися як звичайна чи чітка множина.

2.5 Кластеризація

2.5.1 K-means

Метод k-середніх використовується для кластеризації даних на основі алгоритму розбиття векторного простору на певну кількість кластерів k .

Алгоритм є ітераційною процедурою, в якій виконуються такі кроки:

- вибирається кількість кластерів k ;
- з вихідної множини даних випадковим чином вибираються k спостережень, які служитимуть початковими центрами кластерів;
- для кожного спостереження вихідної множини визначається найближчий до нього центр кластера (відстань вимірюється в метриці Евкліда). У цьому записи, «притягнуті» певним центром, утворюють початкові кластери.

- обчислюються центроїди – центри тяжкості кластерів. Кожен центроїд — це вектор, елементи якого є середніми значеннями відповідних ознак, обчислені за всіма записами кластера.
- центр кластера зміщується до його центроїду, після чого центроїд стає центром нового кластера;
- 3-й та 4-й кроки ітеративно повторюються. Очевидно, що на кожній ітерації відбувається зміна меж кластерів та усунення їх центрів. В результаті мінімізується відстань між елементами усередині кластерів та збільшуються міжкластерні відстані.

Зупинка алгоритму проводиться тоді, коли межі кластерів та розташування центроїдів не перестануть змінюватися від ітерації до ітерації, тобто. на кожній ітерації в кожному кластері залишатиметься той самий набір спостережень. Насправді алгоритм зазвичай знаходить набір стабільних кластерів за кілька десятків ітерацій.

Перевагою алгоритму є швидкість та простота реалізації. До недоліків можна віднести невизначеність вибору початкових центрів кластерів, а також те, що число кластерів має бути задано спочатку, що може вимагати деякої апріорної інформації про вихідні дані.

2.5.2 Ієрархічна кластеризація

Серед алгоритмів ієрархічної кластеризації виділяються два основні типи: висхідні та низхідні алгоритми. Низхідні алгоритми працюють за принципом «зверху-вниз»: на початку всі об'єкти поміщаються в один кластер, який потім розбивається на дрібніші кластери. Найбільш поширені висхідні алгоритми, які на початку роботи поміщають кожен об'єкт в окремий кластер, а потім об'єднують кластери в дедалі більші, поки всі об'єкти вибірки не будуть утримуватися в одному кластері. У такий

спосіб будується система вкладених розбиття. Результати таких алгоритмів зазвичай у вигляді дерева – дендрограми. Класичний приклад такого дерева – класифікація тварин та рослин.

У разі використання ієрархічних алгоритмів постає питання, як поєднувати між собою кластери, як обчислювати «відстань» між ними. Існує кілька метрик.

Одиночний зв'язок (відстань найближчого сусіда).

У цьому методі відстань між двома кластерами визначається відстанню між двома найближчими об'єктами (найближчими сусідами) у різних кластерах. Результуючі кластери мають тенденцію поєднуватися в ланцюжки.

Повний зв'язок (відстань найбільш віддалених сусідів).

У цьому методі відстані між кластерами визначаються найбільшою відстанню між будь-якими двома об'єктами в різних кластерах (тобто найбільш віддаленими сусідами). Цей метод зазвичай працює дуже добре, коли об'єкти походять з окремих груп. Якщо ж кластери мають подовжену форму або їх природний тип є «ланцюжковим», цей метод непридатний.

Незважена попарна середня.

У цьому методі відстань між двома різними кластерами обчислюється як середня відстань між усіма парами об'єктів у них. Метод ефективний, коли об'єкти формують різні групи, однак він працює однаково добре і у випадках протяжних («ланцюжкового» типу) кластерів.

Зважена попарна середня.

Метод ідентичний методу незваженого попарного середнього, за винятком того, що при обчисленнях розмір відповідних кластерів (тобто кількість об'єктів, що містяться в них) використовується як ваговий коефіцієнт. Тому цей метод має бути використаний, коли передбачаються нерівні розміри кластерів.

Незважений центроїдний метод.

У цьому методі відстань між двома кластерами визначається як відстань між їхніми центрами тяжіння.

Зважений центроїдний метод (медіана)

Цей метод ідентичний попередньому, крім того, що з обчислення використовуються ваги для обліку різниці між розмірами кластерів. Тому, якщо є або підозрюються значні відмінності у розмірах кластерів, цей метод виявляється кращим за попередній.

Для обчислення відстаней між кластерами найчастіше користуються двома відстанями: одиночним або повним зв'язком.

До недоліків ієрархічних алгоритмів можна віднести систему повних розбиття, яка може бути зайвою в контексті задачі, що розв'язується.

2.6 Використання остових дерев в моделі в моделі

Для формування моделі необхідні всі графи отримані до цього. Оскільки зараз вага ребра кожного з графів прямо пропорційна до взаємозв'язку між словами, необхідно трансформувати ваги так щоб залежність стала оберненою. Це дозволить побудувати мінімальні остові дерева. Тому вага кожного ребра у графах T та S e_i буде замінена на $\frac{1}{e_i}$. Після побудови остових дерев необхідно повернути початкові ваги ребер.

Параметрами моделі є коефіцієнт затухання k та глибина проходження d .

Коефіцієнт затухання k варіюється від 0 до 1 та є деяким аналогом «пам'яті» - показує на скільки зменшується вага ребер для тих вершин(членів речення) яку зустрічались раніше у діалозі. Таким чином ваги ребер для першого у діалозі речення будуть меншими ніж для останнього. Вага ребер під час побудови діалогу обчислюється як $e_i * k * \frac{1}{n}$ де n – порядковий номер речення.

Глибина проходження d – глибина розширення графу який формується під час побудови діалогу. Для кожного графу ці параметри задаються окремо.

Алгоритм імітації діалогу:

- серед вершин графу T обрати ті, що вже зустрічались в діалозі;
- до цих вершин додати ті, що знаходяться на відстані не більше d_t ребер;
- до ребер утвореного графу застосувати коефіцієнт затухання k_t ;
- замінити поточні ваги ребер e_i на $\frac{1}{e_i}$;
- знайти мінімальний шлях до кожної з вершин на глибині d_t ;
- обрати вершину з мінімальним найкоротшим шляхом – таким чином для речення обрано підмет і присудок;
- серед вершин графу S обрати ті, що вже зустрічались в діалозі з обраними підметом і присудком;
- до ребер утвореного графу застосувати коефіцієнт затухання k_s ;
- обрати кількість другорядних членів речення які будуть використані (наприклад випадково);
- сформуванати речення на основі обраних слів.

Таким чином під час побудови діалогу буде формуватися контекст, проте те про що йшла мова 10 речень назад не матиме такого самого впливу як те, про що мова йде зараз.

Також слід задати окремо задати умови при яких система ставитиме питання. Одним з варіантів є випадково вибирати раз у скільки речень задавати питання. Іншим варіантом є ввести метрику різноманіття – якщо в діалог не поступає нової інформації, необхідно задати питання. Такою метрикою може бути максимальна відстань між вершинами у графі T за останні n речень. Параметр n буде регулювати інтенсивність питань – чим він більший, тим меншою буде інтенсивність.

2.7 Побудова граматично правильних речень

Для формування граматично правильних речень буде використовуватись такі підходи:

- використання таблиці з частотами зустрічі слів у певній послідовності;
- використання статичних правил формування речень;
- використання статичних правил створення словосполучень.

У результаті на основі ключових слів та правил будуть формуватися речення для імітації діалогу.

2.8 Функція оцінки результату

Для оцінки результатів застосовують два типи метрик:

- зовнішні;
- внутрішні.

Зовнішні метрики – оцінка результату, незалежна від даних, наданих моделі. Якщо в діалозі кожному реченню давати оцінку власноруч, або кожному діалогу взагалом, то це буде зовнішня оцінка.

Внутрішні метрики – метрики які засновуються на тих самих даних що і навчання моделі (або на спеціально виокремленій тестовій частині даних), проте ці метрик не використовувались для навчання моделі.

Для оцінки роботи діалогової моделі буде використовуватись внутрішні метрики, так як мануальна оцінку результату є затратною.

Для обчислення метрик необхідно побудувати матриці L_1 та L_2 . Для матриці L_1 кожен стовбець та рядок представляють собою слово, а $l1_{ij}$ – кількість разів яку слова i та j зустрічались в одному тексті. Для матриці L_2 кожен стовбець та рядок також представляють собою слово, а $l2_{ij}$ – кількість разів яку слова i та j зустрічались в одному реченні.

На основі даних таблиць проводяться дві відповідні кластеризації. Далі для кожного речення згенерованого системою проводяться дві оцінки.

Оцінка цілісності побудови речень – члени речення кластеризуються відповідно матриці L_1 та якщо вони потрапляють у різні класи то речення не є цілісним. Для кожного слова необхідно обрати топ n кластерів до яких вони належать, та для речення обраховується мінімальна кількість кластерів у яку вкладаються всі слова. Чим менший цей показник, тим однорідніші речення.

Оцінка однорідності тексту – кожен член речення кластеризується відповідно матриці L_2 - для кожного слова обраховується його степінь приналежності до кластеру. Для кожного речення обраховується приналежність до кластеру як

$$l2 = \max\left(\frac{\sum_i^n l2_i}{n}\right) \quad (2.10)$$

, де

$\sum_i^n l2_i$ – сума приналежностей до кластеру,

n – кількість слів у реченні.

Для тексту обраховується кількість кластерів, які зустрічаються у реченнях. Чим менше кластерів – тим більш однорідний текст.

Висновки по розділу

В даному розділі розглянуто і проаналізовано граматичний аналіз слів та синтаксичний аналіз речень в цілому, наведено опис і характеристики різних алгоритмів кластеризації, їх основні переваги та недоліки. Також сформульовано аналізу даних для навчання та формування відповідей у діалозі на основі навченої моделі.

3 РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

3.1 Загальна архітектура

Для створення системи генерації діалогів, що допоможе клієнтам інтернет-магазинів, необхідно щоб у навченої моделі залишався взаємозв'язок з актуальною базою товарів. Таким чином буде поєднуватись як актуальна інформація з магазину, так і данні отримані під час навчання моделі. Для цієї мети рішення структуро поділено на 2 частини: власне модель генерації діалогу та адаптер до бази даних. З одного боку дані для навчання будуть поступати в модель, і якщо необхідно розширити список тем про які може йти мова, то слід просто додати нові дані для навчання на обрану тему. З іншого боку модель буде постійно використовувати дані з бази даних для отримання актуальної інформації, та генерувати дані роботи з клієнтами (рисунок 3.1).

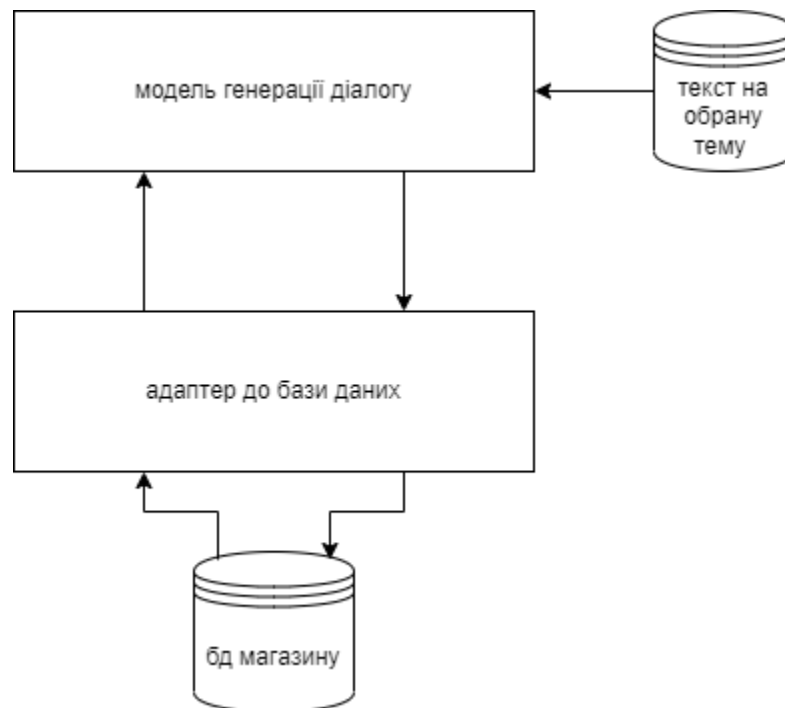


Рисунок 3.1 – Загальна архітектура

3.2 Архітектура рішення

3.2.1 Архітектура моделі

Для адаптації до бази даних модель матиме два режими формування речень: залежний від бази режим, та незалежний від бази режим. Незалежний режим буде використовувати лише данні на яких навчена модель, та буде застосовуватись при загальних питаннях.

Залежний від бази режим буде використовуватись при конкретних типах питань, які потребують даних магазину, таких як:

- який найпопулярніший холодильник;
- яка найдешевша пральна машина;
- та ін.

Цей режим матиме меншу варіативність відповідей, проте комбінація цих двох варіантів роботи дозволить водночас чітко і розгорнуто відповідати на питання, поставлені користувачем.

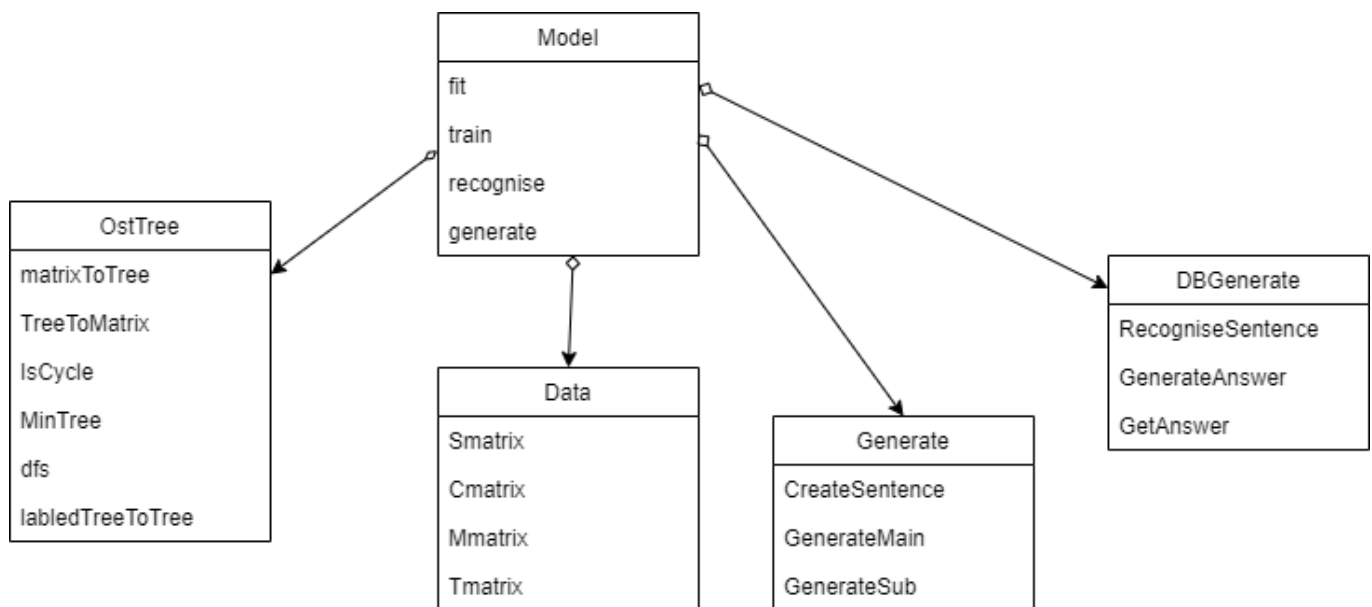


Рисунок 3.2 – Архітектура моделі

3.2.2 Архітектура адаптеру

Основна задача адаптера – це обмін даними між базою та моделлю. З одного боку в модель повинні надходити дані про кількість наявного товару, частоту покупки товару, належність товару до категорії та ін. З іншого боку буде корисним зберігати у базі про які категорії товарів частіше питають користувачі та які товари розглядають. Також такі дані можуть слугувати внутрішньою оцінкою ефективності моделі – якщо категорії товарів про які йде мова в діалогах почнуть купувати більше або стільки ж з урахуванням росту ринку, то модель дійсно ефективна.

Виходячи с задач, архітектура адаптера представлена на рис.3.3.

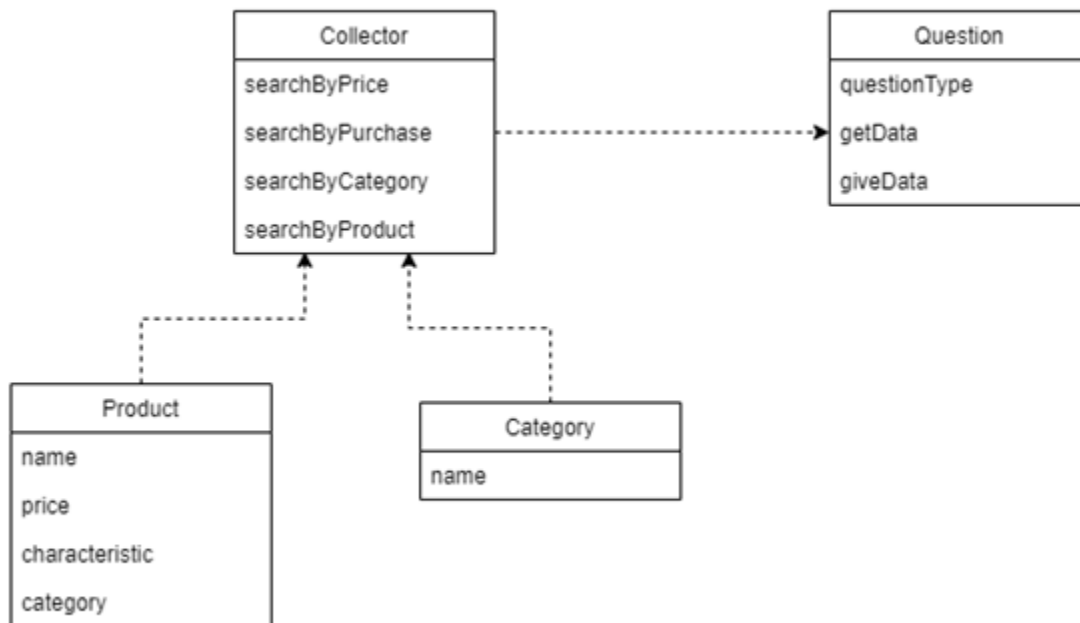


Рисунок 3.3 – Архітектура адаптеру

3.2.3 Опис архітектури

У таблицях 3.1-3.4 наведена структура таблиць бази даних, схема бази даних наведена у додатку графічного матеріалу.

Таблиця 3.1 — Опис таблиці client

Назва стовпця	Опис	Тип даних	Довжина	Первинний	Зовнішній
name	ID каталогу	INT			
gender	Стать користувача	VARCHAR	10		
age	Вік користувача	INT			
datefrom	Дата реєстрації користувача	DATETIME			
mail	Електронна пошта користувача	VARCHAR	50		
id	ID користувача	INT		X	

Таблиця 3.2 — Опис таблиці dialog

Назва стовпця	Опис	Тип даних	Довжина	Первинний	Зовнішній
datefrom	Дата розмови	DATETIME			
client_id	ID користувача	VARCHAR	50		X
id	ID розмови	INT		X	

Таблиця 3.3 — Опис таблиці dialog_detail

Назва стовпця	Опис	Тип даних	Довжина	Первинний	Зовнішній
text	Текст речення	VARCHAR	1000		
category_id	ID категорія товарів, згадана в реченні	INT			X
product_id	ID товару, згаданий в реченні	INT			X
dialog_id	ID діалогу	INT			X
id	ID речення	INT		X	

Таблиця 3.4 — Опис таблиці category

Назва стовпця	Опис	Тип даних	Довжина	Первинний	Зовнішній
isArchive	Активна категорія чи ні	BOOL			
name	Назва категорії товару	VARCHAR	100		
id	ID категорії товару	INT		X	

Таблиця 3.5 — Опис таблиці order_detail

Назва стовпця	Опис	Тип даних	Довжина	Первинний	Зовнішній
id	ID позиції у замовленні	INT		X	
price_id	ID поточної ціни товару	INT	20		X
order_id	ID замовлення	INT			X

Таблиця 3.6 — Опис таблиці products

Назва стовпця	Опис	Тип даних	Довжина	Первинний	Зовнішній
name	Назва товару	VARCHAR	100		
category_id	ID категорії товару	INT			X
quantity	Кількість товару на складі	INT			
isArchive	Чи активний товар	BOOL			
characteristic	Характеристика товару	VARCHAR	1000		
id	ID товару	INT		X	

Таблиця 3.7 — Опис таблиці price

Назва стовпця	Опис	Тип даних	Довжина	Первинний	Зовнішній
price	Ціна товару	MONEY			
datetill	Кінець періоду, у якому даний товар має таку ціну	DATETIME			
datefrom	Початок періоду, у якому даний товар має таку ціну	DATETIME			
product_id	ID товару	INT			X
id	ID ціни	INT		X	

Таблиця 3.8 — Опис таблиці order

Назва стовпця	Опис	Тип даних	Довжина	Первинний	Зовнішній
datefrom	Дата створення замовлення	BOOL			
status	Статус замовлення	VARCHAR	20		
id	ID замовлення	INT		X	
client_id	ID користувача	INT			

Діаграма класів програмного продукту наведена у графічному матеріалі.

Детальний опис класів та функцій наведений нижче в таблицях.

Таблиця 3.9 — Опис класів

Клас	Опис
Model	Клас представляє собою модель, та містить функції навчання та генерації речень для діалогу
OstTree	Клас агрегує функції для роботи з остовими деревами
Data	Клас представляє собою результат обробки вхідного тексту для навчання моделі
Generate	Клас агрегує функції для генерації речення на основі навченої моделі та отриманих даних

Закінчення таблиці 3.9

DBGenerate	Клас агрегує у собі функції для розпізнавання питань орієнтованих на базу, визначення типу питань та отримання відповіді
Question	Клас представляє собою питання, задане користувачем та містить функції для управління пошуком товарів для відповіді
Collector	Клас агрегує у собі пошук товарів та категорії по різним характеристикам
Product	Клас представляє собою товар
Category	Клас представляє собою категорію товару

Таблиця 3.10 — Опис методів класів

Клас	Метод	Опис
Model	fit	Метод який трансформує данні у необхідний для моделі вигляд та зберігає їх всередині моделі
Model	train	Метод який навчає модель на основі даних які в ній містяться
Model	recognize	Метод який розпізнає, задане питання є відкритим чи стосується бази даних
Model	generate	Метод який генерує відповідь на питання
OstTree	matrixToTree	Метод який трансформує матричне представлення графу у власне граф

Продовження таблиці 3.10

OstTree	TreeToMatrix	Метод який трансформує граф у матричне представлення
OstTree	IsCycle	Метод, який перевіряє граф на цикли
OstTree	MinTree	Метод, який шукає мінімальне остове дерево графу
OstTree	dfs	Метод, який здійснює обхід графу вглибину
OstTree	labeledTreeToTree	Внутрішній метод для отримання графу який містить лише відмічені ребра вхідного графу
Data	SetSgraph	Метод який формує S граф для заданого тексту
Data	SetCgraph	Метод який формує C граф для заданого тексту
Data	SetMgraph	Метод який формує M граф для заданого тексту
Data	SetTgraph	Метод який формує T граф для заданого тексту
Data	GetSgraph	Метод який повертає S граф для заданого тексту
Data	GetCgraph	Метод який повертає C граф для заданого тексту

Продовження таблиці 3.10

Data	GetMgraph	Метод який повертає М граф для заданого тексту
Data	GetTgraph	Метод який повертає Т граф для заданого тексту
Generate	CreateSentence	Метод який генерує граматично правильне речення на основі результату роботи моделі
Generate	GenerateMain	Метод який генерує підмет і присудок на основі поточного діалогу
Generate	GenerateSub	Метод який генерує другорядні члени речення на основі вже сгенерованих підмета та присудка
DBGenerate	RecogniseSentence	Метод для визначення типу спеціального речення, а саме: самий популярний товар, самий дешевий товар, самий дорогий товар, що є схоже на даний товар, яка категорія підходить під запит користувача
DBGenerate	GenerateAnswer	Метод який повертає результати пошуку відповіді на питання у базі
DBGenerate	GetAnswer	Метод для пошуку товарів відповідно типу питання
Question	getData	Метод який повертає результати пошуку відповіді на питання у базі для конкретного типу питання

Закінчення таблиці 3.10

Question	setData	Метод який отримує результат пошуку для конкретного питання
Collector	searchByPrice	Метод для пошуку товару за заданою ціною
Collector	searchByPurchase	Метод для пошуку товару за заданою кількістю покупок
Collector	searchByCategory	Метод для пошуку товару за категорією
Collector	searchByProduct	Метод для пошуку товару за назвою
Product	getProduct	Метод для представлення продукту
Product	setProduct	Метод для визначення продукту
Category	getCategory	Метод для представлення категорії
Category	setCategory	Метод для визначення категорії

Висновки по розділу

В третьому розділі описано етап розробки алгоритмічного та програмного забезпечення системи генерації діалогу. Наведено загальну архітектуру рішення та детальну архітектуру окремих його частин. Описана структура тестової бази даних та структура класів. Результатом є адаптація загального алгоритму генерації діалогу до конкретної бази даних та створення адаптивної моделі генерації діалогу.

4 АНАЛІЗ ЕФЕКТИВНОСТІ

При обиранні алгоритму для реалізації задачі важливим фактором є його швидкість, яка оцінюється обчислювальною складністю алгоритму. Складність алгоритму – це функція яка описує кількість операцій що відбуваються під час роботи алгоритму в залежності від об'єму вхідних даних. Вона в основному оцінюється асимптотично. Існує три способи асимптотичної оцінки:

- оцінка зверху;
- оцінка знизу;
- оцінка зверху і знизу.

Нехай алгоритм має складність $f(n)$, оцінка зверху позначається як $f(n) \in O(g(n))$ та означає що $\exists(C > 0), n_0: \forall(n > n_0) |f(n)| \leq |C * g(n)|$, рисунок 4.1.

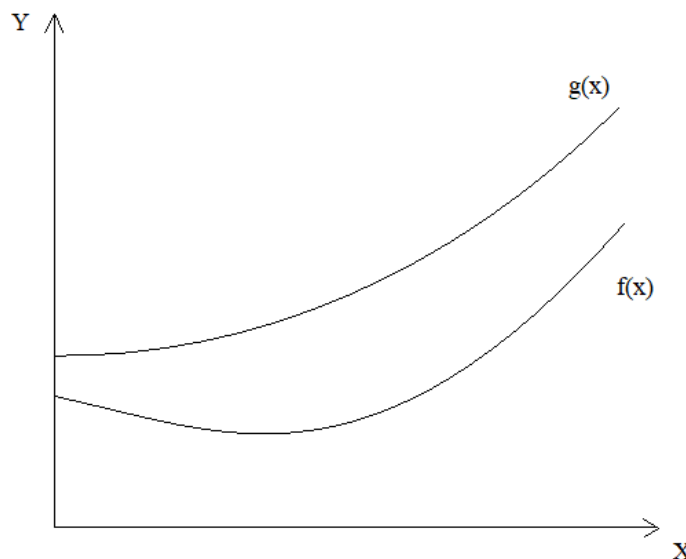


Рисунок 4.1 – Оцінка зверху

Нехай алгоритм має складність $f(n)$, оцінка знизу позначається як $f(n) \in \Omega(g(n))$ та означає що $\exists(C > 0), n_0: \forall(n > n_0) |f(n)| \geq |C * g(n)|$, рисунок 4.2.

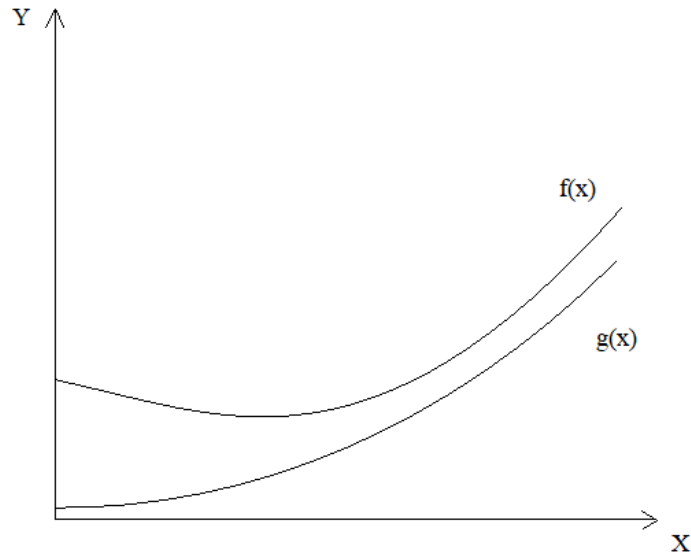


Рисунок 4.2 – Оцінка знизу

Оцінка зверху і знизу позначається як $f(n) \in \Theta(g(n))$ та означає що $\exists(C_1, C_2 > 0), n_0: \forall(n > n_0) |C_1 * g(n)| \geq |f(n)| \geq |C_2 * g(n)|$, рисунок 4.3.

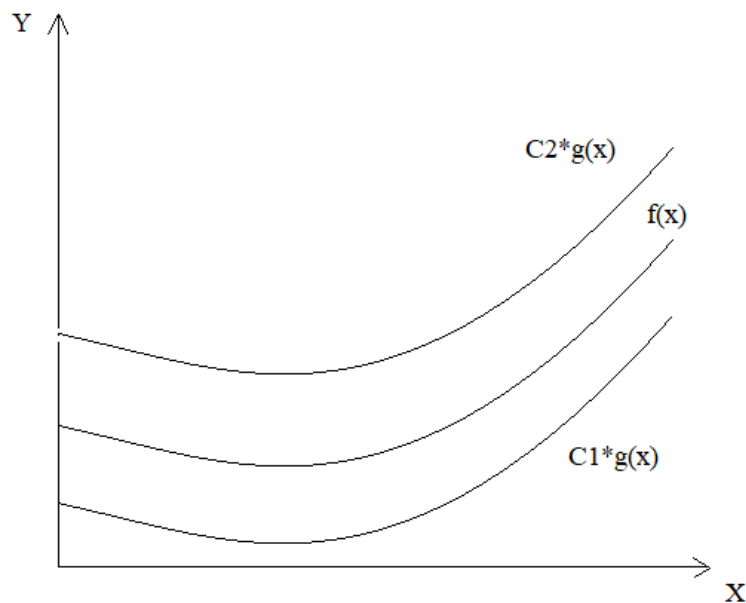


Рисунок 4.3 – Оцінка зверху і знизу

Найчастіше використовують оцінку зверху та саме вона буде використовуватись для оцінки алгоритму у даній роботі.

4.1 Підготовка даних

Для повної оцінки підготовки даних, слід оцінити як швидкість обробки даних так і об'єм пам'яті яку вони будуть займати під час роботи алгоритму.

4.1.1 Розмір даних

Для моделі необхідно створювати одні і ті самі данні, проте в залежності від формату представлення їх обробка може бути більш або менш ефективною. У даному випадку буде розглядатись матричне представлення даних та представлення даних у вигляді графів.

При обробці вхідних даних формуються 4 сутності які відображають:

- частоту зустрічі одного слова після іншого;
- частоту зустрічі підмета і присудка в одному реченні;
- частоту зустрічі пар підмет і присудок в одному тексті;
- частоту зустрічі другорядного члена речення з певною парою підмета і присудка в одному реченні.

При представленні у вигляді графу вершинами будуть слова або пари слів відповідно до сутності, а ребра – власне кількість разів яку зустрічались разом слова або пари слів і вагою ребра буде ця кількість.

При матричному представленні кожний рядок та стовбець буде представляти собою слово або пару слів відповідно до сутності, а значення комірки матриці – кількість разів яку слово або пара слів рядка зустрілась разом зі словом або парою слів зі стовбця.

Розмір представлення у вигляді матриці не залежить від взаємозв'язків між словами – при n слів у вхідних даних завжди буде формуватися матриця $n * n$ вне залежності від зв'язків між словами.

Представлення у вигляді графу є більш економічним по пам'яті оскільки лише у найгіршому випадку коли кожне слово зв'язане з кожним словом, розмір матриці буде складати $3 * C_2^n = \frac{3*n!}{2!(n-2)!} = \frac{3*(n-1)*n}{2} = 1.5n^2 - 1.5n - 3$ стовпці (вершина 1, вершина 2, вага ребра) * кількість пар вершин які можна вибрати з набору вершин n без урахування порядку, тобто C_2^n . Якщо ж кожне слово зв'язане, наприклад, лише з половиною інших слів, то розмір даних що необхідно зберігати в пам'яті стає ще меншим.

4.1.2 Обробка даних

Для оцінки необхідно розглянути використання як алгоритму кластеризації так і для пошуку мінімального остового дерева.

Для алгоритму кластеризації необхідно шукати відстані між словами, у даному випадку під відстанню мається на увазі $\frac{1}{m}$, де m – кількість разів яку слова зустрічались разом. Також необхідно знаходити слова які будуть виступати центроїдами кластерів. Для цього в межах кластера необхідно буде шукати відстані від всіх слів до всіх, щоб знайти те слово від якого найменша відстань до всіх інших.

Для алгоритму пошуку мінімального остового дерева необхідно шукати ребра графу, які мають найменшу вагу та відмічати їх як використані. Після першого ребра кожне наступне необхідно перевіряти на те, чи утворює воно цикл в новоутвореному графі.

При матричному представленні пошук мінімальної відстані до елемента має складність $O(n)$ – лінійний пошук у рядку мінімального елемента. Для кожного кластера знайти елемент, середня відстань до якого мінімальна – для кожного елемента обчислити середню відстань до всіх інших має складність $O(n^2)$: обчислити середню відстань для одного елемента $O(n)$ n разів. Тобто складність однієї ітерації складає

$$O(n^2) + O(n) \quad (4.1)$$

Для пошуку мінімального остового дерева в одній ітерації необхідно знайти ребро з мінімальною вагою ,оцінити чи утворює це ребро цикл з вже відміченими вершинами та перевірити, чи всі вершини потрапляють у новий граф.

Складність пошуку мінімального елемента у матриці $O(n^2)$ – лінійний пошук по n^2 елементам. Для перевірки входження вершин у новий граф необхідно сформувати новий одновимірний масив у якому кожне значення – чи відмічена вершина. Складність пошуку у такому масиві елемента складає $O(n)$. Для перевірки на циклічність необхідно здійснити пошук вглибину. При матричному представленні даних для кожної вершини буде здійснюватись перевірка, чи можна повернутись у неї більше ніж за 2 кроки(щоб виключити сусідні вершини у неорієнтованому графі). Тобто для кожної вершини розглянути всі вершини до яких існує шлях від неї. У найгіршому випадку це буде означати що необхідно для кожної вершини розглянути всі вершини, тобто складність цієї операції буде складати $O(n^2)$, тобто сумарно складність ітерації буде складати

$$O(n^2) + O(n^2) + O(n) = O(2n^2) + O(n) \quad (4.2)$$

Для пошуку мінімальної відстані від кожного елемента до центроїда кластеру, необхідно застосувати алгоритм пошуку найкоротшого шляху k разів, де k – кількість кластерів. Для пошуку найкоротшого шляху можна використати алгоритм Дейкстри, його складність в загальному випадку складає $O(n^2)$. Тоді складність пошуку мінімальної відстані до центроїдів усіх кластерів складатиме $O(k * n^2)$.

Для пошуку середньої відстані до всіх сусідніх вершин достатньо пройти по списку ребер, складність такого пошуку в найгіршому випадку буде складати $O(n^2)$, а в найкращому - $O(n)$, за середній взяти випадок, коли у кожній вершині є зв'язок з половиною вершин, то ребер буде $C_2^{0.5*n} = \frac{\frac{n!}{2}}{2! \left(\frac{n}{2}-2\right)!} = \frac{\left(\frac{n}{2}\right) * \left(\frac{n}{2}-1\right)}{2} = \frac{n * (n-2)}{8} = \frac{1}{8}n^2 - \frac{1}{4}n$ у випадку якщо n – парне число і $C_2^{0.5*(n+1)} = \frac{\frac{(n+1)!}{2}}{2! \left(\frac{n+1}{2}-2\right)!} = \frac{\left(\frac{n+1}{2}\right) * \left(\frac{n+1}{2}-1\right)}{2} = \frac{(n+1) * (n-1)}{8} = \frac{1}{8}n^2 - \frac{1}{8}n$. За результатами даного пошуку можна сформувати матрицю розмірності n та обрати нові центроїди кластерів. Складність пошуку у такій матриці складатиме $O(n)$. Тобто у будь-якому випадку складність буде $O\left(\frac{1}{8}n^2\right)$. Тоді в середньому випадку складність ітерації буде

$$O(k * n^2) + O\left(\frac{1}{8}n^2\right) + O(n) \quad (4.3)$$

Пошук (обхід вглибину або вширину) у графі має складність $O(n+v)$ де n – кількість вершин, а v – кількість ребер, при найгіршому варіанті складність буде являти собою $O\left(n + \frac{(n-1)*n}{2}\right) = O\left(n + \frac{1}{2}n^2 - \frac{1}{2}n\right) = O\left(\frac{1}{2}n^2 + \frac{1}{2}n\right)$. Лінійний пошук ребра з найменшою вагою матиме складність $O(n)$. Тому складність ітерації складатиме

$$O\left(\frac{1}{2}n^2 + \frac{1}{2}n\right) + O(n) \quad (4.4)$$

Отже серед формул 4.1-4.4 найменшу складність має формула 4.4, тобто пошук мінімального остового дерева при представленні у вигляді графу.

4.2 Модель

При використанні простого алгоритму кластеризації k -mean складність кожної ітерації складає $O(2kn)$, де k – кількість кластерів, тобто константа, а n – кількість слів. Проте кількість ітерацій залежить від того на скільки чітко виокремленими є кластери та від налаштувань моделі – максимальної кількості ітерацій та точності. Віднесення до кластеру нового слова матиме складність $O(n)$.

Складність пошуку мінімального остового дерева для алгоритму Крускала складає $O(n \cdot \ln(n))$. Складність пошуку елемента буде залежати від конфігурації отриманого дерева. У найгіршому випадку – ланцюг – складність становитиме $O(n)$, у найкращому – зірка – $O(1)$. Середній випадок буде визначатись середньою кількістю зв'язків для слова та середня складність буде становити $O(\log_m n)$, де m – середня кількість зв'язків для слова, а n – кількість слів.

Як результат, при підході представлення даних як графу та застосування алгоритму пошуку мінімального остового дерева є більш ефективним як при навчанні моделі так і при подальшій її роботі.

Висновки до розділу

Отже у даному розділі було порівняно два способи зберігання оброблених даних з боку зайнятого об'єму пам'яті та зручності подальшого використання у різних моделях. Також було порівняно дві моделі, та складність навчання цих моделей та формування результатів роботи після навчання. У результаті такого

порівняння було обрано зберігання даних у вигляді графу та модель, що використовує остові дерева.

5 МАРКЕТИНГОВИЙ АНАЛІЗ СТАРТАП-ПРОЄКТУ

5.1 Опис ідеї проєкту

Таблиця 5.1 – Інформаційна карта проєкту

1. Назва проєкту	Програмне та математичне забезпечення для генерації діалогових систем з підтримкою української мови
2. Автори проєкту	Корольова Людмила Вікторівна
3. Коротка анотація	В основі ідеї проєкту лежить використання програмного застосунку для генерації діалогу з підтримкою української мови для використання у інтернет-магазинах
4. Термін реалізації проєкту	12 місяців
5. Необхідні ресурси	<p>Інтелектуальні:</p> <p>методичне забезпечення: підручники, статті</p> <p>Фінансові: оплата за інтернет – 1188 грн, Оренда приміщення - 181000грн Оплата підписки на AWS – 7000 грн, Оплата стартаперів $61000+61000*0,22+1100*12=87620$ грн канцелярія – 4000 грн.</p> <p>оформлення авторського права, торгової марки – 32000 грн.</p> <p>Матеріальні: ноутбуки x2 Lenovo – 80 000 грн, ноутбуки x2 Asys – 44 000 грн.</p> <p>Всього: 374 808 грн.</p>

Закінчення таблиці 5.1

6. Опис проблеми, яку вирішує проект	Відсутність розвинених систем для генерації діалогу з підтримкою української мови
7. Головні цілі та завдання проекту	<p>Ціль: створення програмного застосунку для аналізу дорожнього трафіку</p> <p>Завдання:</p> <ol style="list-style-type: none"> 1. Створення зручного інтерфейсу користувача. 2. Створення робочого програмного застосунку. 3. Забезпечення візуалізації дорожнього трафіку 4. Забезпечення визначення засобів оптимізації дорожньої системи міста.
8. Очікувані результати	
Розроблене програмне забезпечення дасть можливість візуалізації та оптимізації дорожнього трафіку.	

5.2 Формування команди стартапу

Таблиця 5.2 – Формування стартапу

Спеціальність	Роль
ІТ-спеціаліст	Генератор ідеї
Маркетолог	Дипломат
Проектний менеджер	Організатор
Розробник	Реалізатор

Таблиця 5.3 – Поставлені завдання та час на їх виконання

	Завдання	Час
1	Дослідження методів реалізації	2
2	Розробка Технічного завдання	1,25
3	Розподіл ролей	0,25
4	Розробка програмного продукту	5
5	Тестування продукту	4
6	Впровадження системи	0,5
7	Пошук інвесторів	2
8	Рекламна компанія	2
9	Аналітика	1
10	Вибір мат.моделі	3

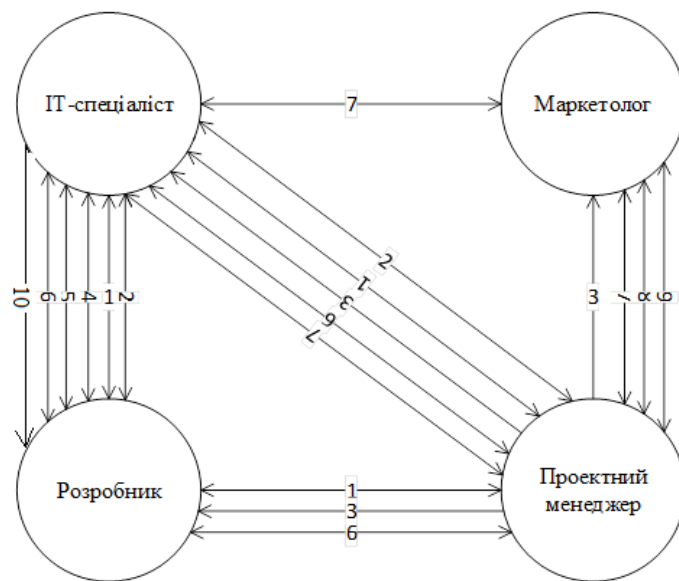


Рисунок 5.1 – Граф розподілення завдань між учасниками команди

Розрахунок завантаженості учасників команди:

Завантаженість = Вхід / Вихід

ІТ-спеціаліст = $11 / 12 = 0,91$

Маркетолог = $5 / 4 = 1.25$

Проектний менеджер = $10 / 12 = 0,83$

Розробник = $9 / 7 = 1,29$

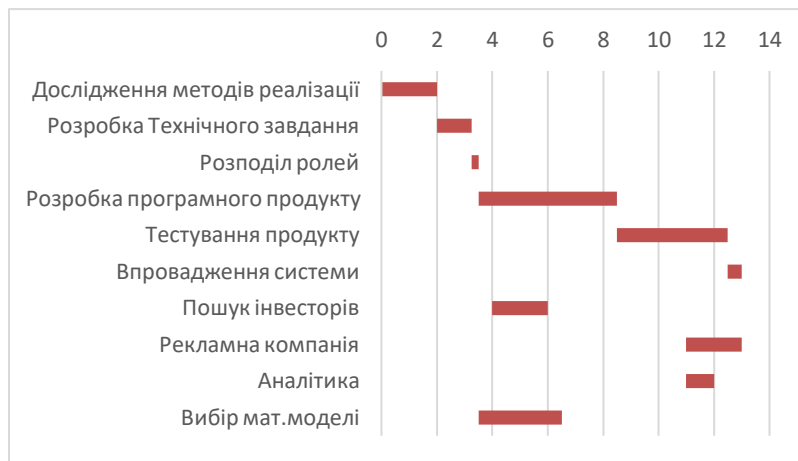


Рисунок 5.2 – Графік розподілення часу

Таблиця 5.4 – Важливість факторів

Фактор	Вага (важливість)
Ідея	6
Підготовка бізнес плану	8
Компетентність	10
Залученість і ризику	9
Обов'язки	7
Залучення партнерів	9

Таблиця 5.5 – Оцінювання особистого внеску

Фактор	Вага	ІТ-спеціаліст	Маркетолог	Проектний менеджер	Розробник
Ідея	6	10	1	4	2
Підготовка бізнес плану	8	4	4	7	3
Компетентність	10	9	3	6	10
Залученість і ризику	9	7	2	6	8
Обов'язки	7	8	1	5	5
Залучення партнерів	5	2	6	7	1

Таблиця 5.6 – Визначення дольової участі учасника в проєкті

Фактор	ІТ-спеціаліст	Маркетолог	Проектний менеджер	Розробник	
Ідея	60	6	24	18	
Підготовка бізнес плану	32	32	56	24	
Компетентність	100	30	60	100	
Залученість і ризику	63	18	54	72	
Обов'язки	56	7	35	35	
Залучення партнерів	10	30	35	5	
Разом	321	123	264	254	962
Процент	33,4%	12,8%	27,4%	26,4%	100,0%

5.3 Морфологічна карта

За допомогою використання морфологічних карт згенеруємо ідею системи, що буде виконувати математичні обчислення. Система повинна бути легкою в налаштуванні, кросплатформною, та мати сучасний зручний інтерфейс.

Визначення основних функцій (їх можна визначити як за результатами наукових досліджень, так і суто інтуїтивно). У цьому випадку їх визначено за комбінацією названих методів:

- швидкість інтернету 70 – 120 Mbps;
- процесор ПК (чим потужніше тим краще);
- кросплатформенність системи (підтримка як можна більше систем);
- версія браузеру повинна бути як можна новіше;
- адаптивність системи 1- 10 розширень екрану.

Таблиця 5.7 – Морфологічна карта

Основні параметри	Проміжні рішення				
	1-ше	2-ге	3-тє	4-те	5-те
Швидкість інтернету	70	85	95	100	120
Процесор ПК	Intel Pentium	Intel Core i3	Intel Core i5	Intel Core i7	Intel Core i9
Кросплатформенність системи	1	2	5	10	20
Версія браузеру	Internet Explorer 10	Internet Explorer 11	Opera	Mozilla Firefox	Google Chrome
Адаптивність системи	1	2	5	7	10

Таким чином, ідею нового продукту можна сформулювати так: розробка програмного та математичного забезпечення для генерації діалогів українською мовою.

Задум товару:

Товар за задумом. Розробка нової бібліотеки для генерації діалогу українською, який підтримується 20-ма системами, безперебійно працює у браузері Google Chrome при процесорі IntelCore i9.

Товар у реальному виконанні. Бібліотека для генерації діалогу українською, який підтримується 10-ма системами, безперебійно працює у браузері Internet Explorer 11 при процесорі IntelCore i7.

Товар з підкріпленням. Вдосконалення макету застосунку для підтримання адаптивності. Розробка блоку для більш точних розрахунків. Подальша технічна підтримка.

Етапи еволюції захисту даних за методом MVP.

Головна проблема: Відсутність рішень для генерації діалогу українською мовою.

1-й MVP: Наймати співробітників для розмови з клієнтами які працюють у магазині

2-й MVP: Наймати співробітників для розмови з клієнтами які працюють у call-центрі.

3-й MVP: Наймати співробітників для розмови з клієнтами які працюють у call-центрі та чаті в інтернет-магазині.

4-й MVP: Створення чат-боту для розмови з клієнтами який працює за принципом автовідповідача.

5-й MVP: Створення чат-боту для розмови з клієнтами який може вести природну розмову.

6-й MVP: Створення чат-боту для розмови з клієнтами який може вести природну розмову та володіє інформацією з бази даних магазину.

Таблиця 5.8 – Визначення стратегії

№ п/п	Чи є проект «першопрохідцем» на ринку?	Чи буде компанія шукати нових споживачів, або забирати існуючих у конкурентів?	Чи буде компанія копіювати основні характеристики товару конкурента, і які?	Стратегія конкурентної поведінки*
	Так	Буде забирати у існуючих конкурентів	Буде використовувати вже існуючі алгоритми з їх подальшою реалізацією	Стратегія заняття конкурентної ніші

Таблиця 5.9 – Удосконалення продукту

№ з/п	Запитання	Відповідь
1	Частиною яких систем є продукт?	Обчислювальних систем.
2	Які функції надсистеми може виконувати продукт? Як їх з ним пов'язати?	При розширенні продукт може стати самостійним чат-ботом
3	Чи можна розділити продукт на етапи?	Так. Кожен крок алгоритму виступатиме окремим етапом.
4	Чи можна об'єднати (агрегувати) кілька елементів продукту в один.	Можливо на одному етапі реалізувати декілька кроків алгоритму.
5	Яким має бути ідеальний продукт?	API з різними функціями

Закінчення таблиці 5.9

6	Що відбудеться, якщо вилучити цей продукт? Чим його можна замінити?	Вилучення даного застосунку, можливо замінити роботою співробітників
7	Яким цей продукт був у минулому?	Раніше не існував.
8	На розвиток яких функцій було спрямоване удосконалення продукту?	Продукт був спрямований на розробку бібліотеки для генерації діалогів українською
9	Які функції залишилися «недорозвиненими»?	Ведення вільного діалогу на будь-яку тему
10	Як можна натепер розвинути ці функції?	Вдосконалити алгоритм що лежить в основі моделі

Далі здійснюється відбір найцікавіших ідей і формується їх список.

Розглянемо ідеї реалізації застосунку:

Ідея 1. Бібліотека для генерації відповідей на шаблонні питання з підключенням до бази даних.

Ідея 2. Бібліотека для генерації діалогів на задану тему.

Ідея 3. Бібліотека для генерації діалогів на будь-яку тему.

Ідея 4. Бібліотека для аналізу розмов з користувачами.

Ідея 5. Бібліотека для генерації відповідей на шаблонні питання з підключенням до бази даних з заданими проте різними варіантами відповіді.

Ідея 6. Бібліотека для генерації відповідей на шаблонні питання з підключенням до бази даних. З урахуванням вподобань користувачів.

Наступним кроком є об'єднання знайдених ідей, їх агрегування або комбінування.

Агрегування 1. При об'єднанні ідеї 1 та ідеї 2, можна отримати бібліотеку для генерації діалогів на задану тему з підключенням до бази даних.

Агрегування 2. При об'єднанні ідеї 1 та ідеї 3 отримаємо бібліотеку для генерації діалогів на будь-яку тему з підключенням до бази даних.

Агрегування 3. При об'єднанні ідеї 4 та 6 отримаємо бібліотеку для генерації відповідей на шаблонні питання з підключенням до бази даних. З урахуванням вподобань користувачів з можливістю аналізу розмов з користувачами.

Таким чином отримано три додаткові ідеї. Кількість продуктивних та унікальних ідей збільшено до дев'яти. Навіть без їх ретельного аналізу, якщо скласти морфологічну таблицю, то можна отримати близько 30–40 якісно нових ідей.

Відбираємо найбільш працездатні ідеї, перевіряємо їх на своєчасність. Для цього необхідно за кожною з отриманих ідей відповісти на три запитання: що вийшло; де це можна використати; кому це потрібно.

Ідея 1. Бібліотека для генерації відповідей на шаблонні питання з підключенням до бази даних.

Отримуємо бібліотеку пристосовану до бази даних магазину:

- можна використати при розробці чат-боту в інтернет-магазині;
- розробникам чат-ботів інтернет-магазину.

Ідея 2. Бібліотека для генерації діалогів на задану тему.

Отримуємо загальну бібліотеку:

- можна використати при розробці чат-боту в інтернет-магазині;
- розробникам будь-яких чат-ботів.

Ідея 3. Бібліотека для генерації діалогів на будь-яку тему.

Отримуємо загальну бібліотеку:

- можна використати при розробці чат-боту в інтернет-магазині;
- розробникам будь-яких чат-ботів.

Ідея 4. Бібліотека для аналізу розмов з користувачами. Отримуємо бібліотеку пристосовану до бази даних магазину:

- можна використати при розробці чат-боту в інтернет-магазині;
- розробникам чат-ботів інтернет-магазину.

Ідея 5. Бібліотека для генерації відповідей на шаблонні питання з підключенням до бази даних з заданими проте різними варіантами відповіді.

Отримуємо бібліотеку пристосовану до бази даних магазину:

- можна використати при розробці чат-боту в інтернет-магазині;
- розробникам чат-ботів інтернет-магазину.

Ідея 6. Бібліотека для генерації відповідей на шаблонні питання з підключенням до бази даних. З урахуванням вподобань користувачів.

Отримуємо бібліотеку пристосовану до бази даних магазину:

- можна використати при розробці чат-боту в інтернет-магазині;
- розробникам чат-ботів інтернет-магазину.

.Агрегування 1.

Отримуємо бібліотеку пристосовану до бази даних магазину проте з можливістю генерації не прив'язаних до неї діалогів:

- можна використати при розробці чат-боту в інтернет-магазині;
- розробникам чат-ботів інтернет-магазину.

Агрегування 2.

Отримуємо бібліотеку пристосовану до бази даних магазину проте з можливістю генерації не прив'язаних до неї діалогів:

- можна використати при розробці чат-боту в інтернет-магазині;
- розробникам чат-ботів інтернет-магазину.

Агрегування 3.

Отримуємо бібліотеку пристосовану до бази даних магазину з можливістю аналізу діалогів з користувачем:

- можна використати при розробці чат-боту в інтернет-магазині;
- розробникам чат-ботів інтернет-магазину.

Надалі потрібно сформулювати та синхронізувати завдання.

Перелічимо знайдені нами рішення в тому порядку, у якому їх можна пропонувати ринку. Таким чином, у табл. наведено цілі організації на найближчі 5, 15, 25 років. Крім того, є план виведення на ринок нового продукту і його постійного відновлення. Якби було пройдено всі етапи ретельно, без пропусків, то одержали б стратегічний план розвитку продукту із щорічною програмою відновлення.

Маючи подібну програму, можна до кожного сезону пропонувати чергове нововведення. Як тільки конкуренти сприймуть таку ініціативу і почнуть тиражувати нововведення, можна виходити на ринок і пропонувати споживачам оновлений продукт, випереджаючи конкурентів.

Таблиця 5.10 – Синхронізація завдань

<i>Етапи</i>	<i>Продукти (послідовність заміщення)</i>		
Минуле століття	MVP 1: Робота співробітника у магазині	MVP 2: Чат-бот з можливістю виклику співробітника	MVP 3: Повністю самостійний чат-бот
Сьогодні	Ідея 1	Ідея 2	Ідея 6
Завтра	Ідея 3	Ідея 4	Агрегування 1
Післязавтра	Ідея 5	Агрегування 2	Агрегування 3
Застосунок XXI століття	Ідея 7 – самостійна бібліотека на рівні GPT3		

5.4 Розроблення ринкової стратегії проєкту

Таблиця 5.11 – Вибір цільових груп потенційних споживачів

№ п/п	Опис профілю цільової групи потенційних клієнтів	Готовність споживачів сприйняти продукт	Орієнтовний попит в межах цільової групи (сегменту)	Інтенсивність конкуренції в сегменті	Простота входу у сегмент
1.	Державні установи, що системою транспорту	Середня, бо державні установи не завжди зацікавлені у освоєнні нових технологій.	60% - середній попит	Відсутня конкуренція	Середньо, бо державні установи не завжди готові витратити гроші на новинки

Продовження таблиці 5.11

2.	Фізичні особи	Низька, бо споживачі не займаються оптимізацією транспортної системи	10% - високий попит на продукцію	Відсутня конкуренція	Важко, це не потрібно фізичним особам
3.	Приватні фірми, що займаються енергетикою	Середня, бо компанії установи не завжди зацікавлені у освоєнні нових технологій.	60% - середній попит на продукцію	Відсутня конкуренція	Середньо, бо немає аналогів
Які цільові групи обрано: приватні фірми, держ. установи					

Таблиця 5.12 – Визначення базової стратегії розвитку

№ п/п	Обрана альтернатива розвитку проекту	Стратегія охоплення ринку	Ключові конкурентоспроможні позиції відповідно до обраної альтернативи	Базова стратегія розвитку*
1.	Виготовлення додаткового програмного забезпечення для створення готових варіантів оптимізації	Стратегія диференційованого маркетингу	Зручність та легкість у користування, не потребує налаштування	Стратегія диференціації

Таблиця 5.13 – Визначення стратегії позиціонування

№ п/п	Вимоги до товару цільової аудиторії	Базова стратегія розвитку	Ключові конкурентоспроможні позиції власного стартап-проекту	Вибір асоціацій, які мають сформулювати комплексну позицію власного проекту (три ключових)
1.	Відсутність налаштування програмного продукту	Стратегія диференціації	Не потрібно розгортати продукт на кожній робочій станції	Видкий вхід в систему Доступність для будь-яких користувачів Безплатформеність
2.	Легкість у використанні	Стратегія диференціації	User-friendly інтерфейс	Інтуїтивний інтерфейс Інтегрованість Інтеракція з клієнтом через підказки

5.5 Розроблення маркетингової програми стартап-проекту

Таблиця 5.14 – Визначення ключових переваг концепції потенційного товару

№ п/п	Потреба	Вигода, яку пропонує товар	Ключові переваги перед конкурентами (існуючі або такі, що потрібно створити)
1.	Українська мова	Модель пристосована до української мови	Українська мова
2.	Адаптованість	Можливість імітувати діалог на різні теми	Більша кількість питань на яку може відповісти система
3.	Пристосованість	Можливість імітувати діалог використовуючи данні з бд	Одночасне використання даних клієнта та даних навчання

Таблиця 5.15 – Опис трьох рівнів моделі товару

Рівні товару	Сутність та складові		
I. Товар за задумом	Розробка нової бібліотеки для генерації діалогів українською мовою		
II. Товар у реальному виконанні	Властивості/характеристики	М/Нм	Вр/Тх /Тл/Е/Ор
	1. Зручний інтерфейс	М	Тх/Ор
	2. Адаптованість	М	Тх/Ор
	4. Швидкість інтернету	Нм	Тх/Вр
	5. Процесор ПК	М	Тх/Вр
Якість: Відповідає рекомендаціям по вдосконаленню сайтів комітетів Верховної Ради України			

II. Товар у реальному виконанні	Пакування: Електронна документація, що містить таку інформацію: <ul style="list-style-type: none"> • загальна назва продукту, власна назва; • найменування та адреса виробника і місце виготовлення; • товарний знак; • Інструкція щодо користування. • Необхідні технічні вимоги.
	Марка: OptiTraff
III. Товар із підкріпленням	До продажу: Адаптивний застосунок з блоком для більш точних обрахунків
	Після продажу: технічна підтримка
За рахунок чого потенційний товар буде захищено від копіювання: патенту та авторського права	

Висновки: За задумом проект імітує діалог відповідаючи на питання покупців.

За рахунок патенту та авторських прав товар буде захищено від копіювання.

Таблиця 5.16 – Визначення меж встановлення ціни

№ п/п	Рівень цін на товари-замінники	Рівень цін на товари-аналоги	Рівень доходів цільової групи споживачів	Верхня та нижня межі встановлення ціни на товар/послугу
	5000 грн	-	340000 грн	4500 – 7000 грн

Таблиця 5.17 – Формування системи збуту

№ п/п	Специфіка закупівельної поведінки цільових клієнтів	Функції збуту, які має виконувати постачальник товару	Глибина каналу збуту	Оптимальна система збуту
	клієнти купують продукт безпосередньо у компанії-розробника	Аналіз ринку Обслуговування	0-Канал нульового рівня (виробник безпосередньо продає товар клієнту)	Через сайт виробника

Таблиця 5.18 – Концепція маркетингових комунікацій

Специфіка поведінки цільових клієнтів	Канали комунікацій, якими користуються цільові клієнти	Ключові позиції, обрані для позиціонування	Завдання рекламного повідомлення	Концепція рекламного звернення
Клієнти дізнаються про товар з листівок та інтернету	Інтернет Роздаткова реклама	Абсолютно новий продукт	Проінформувати про існування продукту	Прорив в сфері оптимізації транспортної системи

5.6 Розроблення маркетингової програми стартап-проекту

Таблиця 5.19 – Попередня характеристика потенційного ринку стартап-проекту

№ п/п	Показники стану ринку (найменування)	Характеристика
1	Кількість головних гравців, од	4
2	Загальний обсяг продаж, грн/ум.од	5000грн*200од. =1000000 грн
3	Динаміка ринку (якісна оцінка)	Зростає
4	Наявність обмежень для входу (вказати характер обмежень)	Відсутні
5	Специфічні вимоги до стандартизації та сертифікації	Наявні. Читабельний шрифт. Якісні зображення. Динамічність.
6	Середня норма рентабельності в галузі (або по ринку), %	$(349200/1000000)*100=35\%$

Таблиця 5.20 - Характеристика потенційних клієнтів стартап-проекту

№ п/п	Потреба, що формує ринок	Цільова аудиторія (цільові сегменти ринку)	Відмінності у поведінці різних потенційних цільових груп клієнтів	Вимоги споживачів до товару
	Відсутність відкритих систем для генерації діалогів українською мовою	Фізичні особи Приватні фірми, що торгівлею в інтернеті	Більше можливостей системи	Можливість використання на різних платформах, Легкий доступ

Таблиця 5.21 – Фактори загроз

№ п/п	Фактор	Зміст загрози	Можлива реакція компанії
1.	Поява конкурентів	Поява могутніх компаній-конкурентів	Запропонувати більш досконалий продукт
2.	Недостатня рекламна компанія	Недостатнє розповсюдження реклами та недостатнє зацікавлення цільової аудиторії	Вдосконалити власну рекламну компанію

Таблиця 5.22 – Фактори можливостей

№ п/п	Фактор	Зміст можливості	Можлива реакція компанії
1.	Нові технології для створення сучасного додатку	Зростання пошуку нових технологій	Відхід від старих технологій
2.	Відсутність конкурентів на вітчизняному ринку	Вільний ринок	Можливість швидкого розвитку
3.	Консультація з потенційними клієнтами	Можна дізнатися який функціонал користувачі хочуть бачити	Створення додатку пристосованого до користувача

Таблиця 5.24 – Ступеневий аналіз конкуренції на ринку

Особливості конкурентного середовища	В чому проявляється дана характеристика	Вплив на діяльність підприємства (можливі дії компанії, щоб бути конкурентоспроможною)
Монополія	Один головний товаровиробник	Здійснення контролю над цінами. Захищеність від конкуренції
Локальний рівень	Конкуренти всередині країни	Пошук однодумців лише в Україні

Продовження таблиці 5.24

Внутрішньогалузева	Конкуренція спостерігається в сфері енергетики	Розробка продукту, що вирішує специфічні задачі
Товарно-родова	Конкуренція між товарами різного виду	Додавання нового функціоналу
Нецінова	Конкуренція проводиться за рахунок вдосконалення якості продукту	Вдосконалення продукту
Немарочна	Роль торгової марки незначна	Заохочення клієнтів якістю товару, а не брендом.

Таблиця 5.25 – Аналіз конкуренції в галузі за М. Портером

Складові аналізу	Прямі конкуренти в галузі	Потенційні конкуренти	Постачальники	Клієнти	Товари-замінники
	Відсутні	Монополія	Розміри поставок	Дешева ціна	Ціна
Висновки:	Конкурентна боротьба відсутня, що зменшує стимул вдосконалення продукту	Можливості виходу на ринок є оскільки не існує аналогів, а товар-замінник має значно гірший функціонал	Незначні обсяги замовлень поставок	Доступна ціна	Обмежень немає

Таблиця 5.26 – Обґрунтування факторів конкурентоспроможності

№ п/п	Фактор конкурентоспроможності	Обґрунтування (наведення чинників, що роблять фактор для порівняння конкурентних проектів значущим)
1.	Технічне обслуговування	Наявність технічної бази з усіх точок
2.	Потреби споживачів	Потреби споживачів впливають на вдосконалення застосунку
3.	Результативність	Кінцевий результат
4.	Відсутність конкурентів	Відсутність варіантів вибору у цільовій аудиторії
5.	Ціна та собівартість продукції	Конкурентоспроможна ціна

Таблиця 5.27 – Порівняльний аналіз сильних та слабких сторін

№ п/п	Фактор конкурентоспроможності	Бали 1-20	Рейтинг товарів-замінників у порівнянні зі стартапом						
			-3	-2	-1	0	+1	+2	+3
1	Технічне обслуговування	13					+		
2	Потреби споживачів	19							+
3	Результативність	17			+				
4	Відсутність конкурентів	16							+
5	Ціна та собівартість продукції	18				+			

Таблиця 5.28 – SWOT - аналіз стартап-проекту

Сильні сторони: даний застосунок дозволяє значно пришвидшити процес обрахунку параметрів геліоустановки та визначення строку її окупності.	Слабкі сторони: компанія є «новачком» на ринку, тому не має репутації. Необхідність залучення інвесторів через недолік фінансування.
Можливості: Ексклюзивна модель Легке налаштування Можливість використання з різних платформ	Загрози: Не є універсальним продуктом, використовується лише в області розробки чат-ботів

Таблиця 5.29 – Альтернативи ринкового впровадження стартап-проекту

№ п/п	Альтернатива (орієнтовний комплекс заходів) ринкової поведінки	Ймовірність отримання ресурсів	Строки реалізації
	Соціальні мережі «Facebook», «Telegram», «Instagram».	51%	4-6 місяців

5.7 Виробничий план

Мета: довести можливості ефективного організаційного та ресурсного забезпечення проекту; продемонструвати, що запропоновані організаційні рішення дозволять ефективно реалізувати стартап-проект.

Таблиця 5.30 – Календарний план-графік реалізації стартап-проекту

№ з/п	Етапи реалізації	Період реалізації проекту						
		0-й рік				1-й рік	2-й рік	3-й рік
		1-й кв.	2-й кв.	3-й кв.	4-й кв.			
1.	Проведення науково-дослідницьких робіт	+						
2.	Розробка ТЗ та ТЕО	+						
3.	Робоче проектування та тестування	+	+	+	+			
4.	Створення компанії			+	+			
5.	Придбання нематеріальних активів, отримання дозвільних документів тощо				+	+		
6.	Придбання й оренда земельних ділянок, будівель, приміщень, споруд		+	+				
7.	Придбання обладнання	+	+					
8.	Перед виробничі маркетингові дослідження	+	+	+				
9.	Впровадження системи				+			
10.	Придбання матеріальних ресурсів			+	+	+		
11.	Рекламна компанія				+	+		
12.	Продаж продукції					+	+	+

Таблиця 5.31 – Планова потреба у виробничих площах

<i>№ з/п</i>	<i>Тип приміщення (будівлі, ділянки, споруди)</i>	<i>Кількість одиниць</i>	<i>Площа, кв. м</i>	<i>Вимоги до приміщення (будівлі, ділянки, споруди)</i>	<i>Умови надання</i>	<i>Вартість, тис. грн.</i>
1.	Офісне приміщення	1	60	Освітлення, температурний режим відповідно до норм ДСТУ	Оренда	16 000 грн
<i>Разом</i>				—	—	16 000 грн

Таблиця 5.32 – Планова потреба у виробничому обладнанні та устаткуванні

<i>№ з/п</i>	<i>Вид обладнання (устаткування, пристрою)</i>	<i>Тип (модель)</i>	<i>Виробник обладнання (устаткування, пристрою)</i>	<i>Терміни постачання</i>	<i>Вартість, тис. грн.</i>
1.	Ноутбук	Aspire 7	Acer	1 тиждень	40 000 грн
2.	Ноутбук	VivoBook S14	Asus	1 тиждень	22 000 грн
<i>Разом:</i>		—	—	—	62 000 грн

Таблиця 5.33 – Планова вартість нематеріальних активів

<i>№ з/п</i>	<i>Вид активів</i>	<i>Активи, що можуть бути віднесені до даного виду</i>	<i>Вартість, тис. грн.</i>
1.	Права користування природними ресурсами	право користування ресурсами природного середовища, геологічною та іншою інформацією про природне середовище	
2.	Права користування майном	право на оренду приміщень	16 000 грн
3.	Права на комерційні позначення	права на торговельну марку (логотип компанії)	10 000 грн
4.	Права на об'єкти промислової власності	право на комерційні таємниці, у тому числі ноу-хау, захист від недобросовісної конкуренції	
5.	Авторське право та суміжні з ним права	право на комп'ютерні програми	6000 грн

Таблиця 5.34 – Плановий обсяг виробництва продукції стартап-проекту

<i>Вид продукції</i>	<i>Одиниця виміру</i>	<i>Обсяги виробництва за період</i>		
		<i>1-й рік</i>	<i>2-й рік</i>	<i>3-й рік</i>
Програмне забезпечення	1 пакет ПО	50 000	100 000	150 000

Таблиця 5.35 – Планова потреба у матеріальних ресурсах та комплектуючих

№ з/п	Вид ресурсу	Одиниця виміру	Витрати на одиницю продукції в натуральних показниках	Вартість на одиницю продукції, тис. грн.	Вартість за плановим обсягом виробництва за період, тис грн.		
					1-й рік	2-й рік	3-й рік
1.	Матеріали						
1.1	Папір	1 упаковка	420	420 грн	10уп*420 грн =4200	15уп*420грн=6300	18уп*420 грн =7560
Всього матеріалів		—	—				
Разом:		—	—		4200 грн	6300 грн	7560грн

Таблиця 5.36 – Планова потреба та витрати на персонал

№ з/п	Категорія персоналу	Чисельність	Заробітна плата, тис грн. на місяць	Відрахування на соціальні заходи, тис грн. на місяць	Витрати на оплату праці за період, тис. грн.		
					1-й рік	2-й рік	3-й рік
1.	Генератор ідей	1	11 000 грн	2970 грн	132 000	264 000	396 000
2.	Маркетолог	1	8 000 грн	2160 грн	96 000	192 000	288 000

Продовження таблиці 5.37

3.	Реалізатор	1	14 000 грн	3780 грн	168 000	336 000	504 000
4.	Організатор	1	16 000 грн	4320 грн	192 000	384 000	576 000
<i>Разом:</i>			49 000 грн	13230 грн	588 000	1176 000	1764 000

Таблиця 5.38 – Загальні початкові витрати проекту

<i>№ з/п</i>	<i>Стаття витрат</i>	<i>Обсяги витрат в 0-й рік, тис. грн.</i>
1.	Проведення науково-дослідницьких робіт	2200
2.	Розробка ТЗ і ТЕО	-
3.	Робоче проектування та тестування	49
4.	Витрати на оренду приміщень	192
5.	Витрати на придбання обладнання	62
6.	Витрати на впровадження системи	7
7.	Витрати на придбання нематеріальних активів	32
8.	Витрати на оплату інтернету	5
9.	Витрати на перед виробничі маркетингові дослідження і створення збутової мережі	-
10.	Витрати, пов'язані з діяльністю персоналу	-
<i>Разом</i>		<i>349 200 грн</i>

Таблиця 5.39 – Планові загальногосподарські витрати

№ з/п	Стаття витрат	Витрати за період, тис. грн.		
		1-й рік	2-й рік	3-й рік
1.	Витрати на оренду земельних ділянок, будівель, приміщень, споруд	192	220	260
2.	Витрати на обладнання	62	80	100
3.	Витрати на придбання нематеріальних активів	32	42	52
4.	Витрати на персонал (на відрядження, соціальні заходи тощо)	588	1176	1764
5.	Витрати на зв'язок	9	10	11
6.	Витрати на паливо та електроенергію	15,2	20	25
7.	Витрати на водопостачання	5	10	15
8.	Витрати на утримання обладнання та приміщень	10	15	20
9.	Витрати на збут	20	30	40
10.	Витрати на просування та рекламу	30	50	80
11.	Оплата юридичних послуг	15	20	30
12.	Податкові платежі (земельний, комунальний податки, інші)	20	37	45
...				
	<i>Разом:</i>	988 200 грн	1710 000 грн	2442 000 грн

Витрати на електроенергію: 800кВт*год на місяць

$100 * 0,9 = 90$ грн

$700 * 1,68 = 1176$ грн

Разом 1266 грн за місяць

$1266 * 12 = 15192$ грн за рік

5.8 Організаційний план

Таблиця 5.40 – Генератор ідей (ІТ- спеціаліст)

Критерій	Зміст
Основна освіта	Технічна в галузі комп'ютерних наук
Додаткова освіта, спеціалізація	Веб-дизайн, архітектура систем
Необхідний досвід роботи	Не потрібен досвід роботи
Завдання	Відповідає за пошук нових рішень та розробку
Знання	Мова Python
Навички, вміння, ділові якості	Аналітичний склад розуму, вміння доносити ідею
Особистісні якості	Впевненість, креативність, відповідальність
Мотивація (що можемо запропонувати)	Реалізація власних ідей, отримання прибутку

Таблиця 5.41 – Маркетолог (Менеджер)

Критерій	Зміст
Основна освіта	Менеджмент та Маркетинг
Додаткова освіта, спеціалізація	Бізнес, Менеджмент, Маркетинг
Необхідний досвід роботи	Від 2 років
Завдання	Відповідає за рекламну кампанію
Знання	Знання про новинки в сфері технологій
Навички, вміння, ділові якості	Стратегії менеджменту, аналіз, дизайн, якість спілкування
Особистісні якості	Креативність, самостійність, вміння працювати на результат, комунікативність, ініціативність, наполегливість, відповідальність, уважність
Мотивація (що можемо запропонувати)	Отримання прибутку

Таблиця 5.42 – Організатор (Проектний менеджер)

Критерій	Зміст
Основна освіта	Технічна в галузі комп'ютерних наук
Додаткова освіта, спеціалізація	Бізнес, Менеджмент
Необхідний досвід роботи	Від 2 років ведення проектів
Завдання	Відповідає за ведення та організацію команди

Закінчення таблиці 5.41

Знання	Технології Agile, Scrum, Kanban
Навички, вміння, ділові якості	Організація, відповідальність, управління ризиками
Особистісні якості	Впевненість, комунікабельність, лідерські задатки, вміння контролювати процеси
Мотивація (що можемо запропонувати)	Створення власного продукту, отримання прибутку

Таблиця 5.42 – Реалізатор (Розробник)

Критерій	Зміст
Основна освіта	Вища освіта з комп'ютерних технологій
Додаткова освіта, спеціалізація	Програмування
Необхідний досвід роботи	Від 1 року
Завдання	Розробка програмного коду, тестування продукту
Знання	Python
Навички, вміння, ділові якості	Аналітичний склад розуму
Особистісні якості	Відповідальність, розуміння постановки завдань
Мотивація (що можемо запропонувати)	Вдосконалення практичних навиків, отримання прибутку

Опишемо організаційну структуру ТОВ UkrSpeak у перспективі на 6-12 місяців. На даний період планується, що компанія розшириться близько до 25 чоловік.

Планується, що у компанії на заданий період буде один керівник, а також 5 підрозділів. Підрозділи будуть поділені за спеціалізацією та включають в собі:

- відділ маркетингу;
- ІТ-відділ;
- відділ продажів;
- HR відділ;
- відділ фінансів.

Кожен відділ буде мати керівника, що буде контролювати роботу всередині відділу та звітувати перед керівником компанії. Також, у свою чергу керівник компанії буде напряму взаємодіяти з керівниками відділів для постановки завдань, прийнятих рішень та оголошення політик та правил компанії.

Опишемо відповідальність та вид діяльності кожного підрозділу.

Відділ маркетингу: займається вивченням ринку та пошуком існуючих рішень у напрямку роботи компанії. Займається розробкою рекламної компанії підтримання іміджу компанії, участю у піар-компаніях.

ІТ-відділ: займається розробкою програмного продукту. Відділ відповідає за розробку та тестування продукту.

Відділ продажів: відповідає з продаж продукту, пошук клієнтів, підтримання актуальності на ринку.

Відділ HR: відповідає за підбір персоналу, налагодження внутрішніх відносин, комфорту кожного працівника, покращення умов.

Відділ фінансів: займається веденням фінансів всередині компанії. Відповідає за заробітну плату працівників.

Переваги обраної структури полягають у:

- чіткості і простоті взаємодії;
- надійному контролі та дисципліні всередині компанії;
- економічності за умов невеликих розмірів організації.

Основним видом матеріальної мотивації працівників у компанії буде підвищення заробітної плати два рази на рік.

До нематеріальної видів мотивації будуть відноситись організація тим-білдінгів для компанії та окремих відділів, а також навчальні центри з можливістю саморозвитку та вдосконалення професійних здібностей.

5.9 Оформлення авторського права на програму

- заповнити заяву на реєстрацію авторського права.

(Заява наведена нижче)

- зареєструвати заяву. Для цього необхідно подати заяву, довіреність та примірник програми у матеріальній формі(наприклад, на флешці) за адресою м Київ, бульвар Дружби Народів, 28 до **Українського агентства з авторських і суміжних прав;**

- сплатити державний збір за реєстрацію та надати квитанцію разом з заявою.

Реквізити для сплати:

Таблиця 5.43 - Реквізити для сплати

Отримувач коштів	УК у Печер. р-ні / Печерс. р-н / 22011900	
ЗКПО	38004897	
Код банку отримувача (МФО)	820019	
Банк отримувача	ГУ ДКСУ у м. Києві	
Рахунок отримувача	31116025700007	
Призначення платежу	22011900; код ЄДРПОУ (для юридичних осіб) або ідентифікаційний код (для фізичних осіб); збір за:	
за підготовку до держреєстрації авторського права від фізичних осіб		55,25 грн.
за оформлення та видачу свідоцтва про реєстрацію авторського права на твір від фізичних осіб		34 грн.

- дочекатися розгляду заяви;
- сплатити державний збір за свідоцтво;
- отримати свідоцтво про реєстрацію авторського права.

Висновки по розділу

Як результат було створено ринкову (маркетингову) програму, що включає в себе визначення ключових переваг концепції потенційного товару, опис моделі товару, визначення меж встановлення ціни, формування системи збуту та концепцію маркетингових комунікацій.

ВИСНОВКИ

Отже в сучасному світі все активніше розвивається надання послуг онлайн. Для отримання послуг люди заходять на сайт у зручний для них час, і іноді цей час не співпадає з робочими годинами консультантів. У таких випадках дуже доречним стає використання програмного забезпечення яке дозволяє імітувати діалог та відповідати на питання покупців.

Проте більшість такого програмного забезпечення має обмежений функціонал, оскільки містить у собі набір приготовлених заздалегідь питань та відповідей. Такий функціонал дозволяє дати відповіді на розповсюдженні питання, проте мінімальне відхилення від стандартних питань вже потребує уваги консультанта магазину.

Тому рішенням даної проблеми є розробка програмного та математичного забезпечення для генерації діалогів, яке одночасно використовує данні магазину, та будь-які тексти що відповідають необхідній тематиці. Це рішення дозволяє підтримувати природний діалог та дає змогу відповідати на більшу кількість питань покупців.

Для розробки такого програмного забезпечення У ході виконання магістерської дисертації було розглянуто питання пов'язані з обробкою української мови. Розглянуті різні задачі обробки природної мови та здійснений пошук існуючих рішень, що або є конкурентами, або можуть допомогти у розробці системи.

Розроблено алгоритм для генерації діалогів пристосований саме до української мови з такими її особливостями як різні форми слів та довільний порядок речень. Алгоритм виключає в себе як попередню обробку даних так і власне модель що імітує діалог.

Розроблено програмне забезпечення, яке одночасно використовує розроблену загальну модель та є адаптованим до конкретної бази даних користувача. Такий підхід

дозволяє комбінувати чіткі відповіді про ціну та популярність товару та широкі відповіді на загальні питання.

Проведений маркетинговий аналіз стартап-проєкту. Проведено опис ідеї проєкту, технологічний аудит ідеї проєкту, аналіз ринкових можливостей запуску стартап-проєкту. Розроблено ринкову стратегію проєкту та маркетингову програму стартап-проєкту.

Результати роботи над магістерською дисертацією опубліковані у одній статті.

Наукова новизна одержаних результатів магістерської дисертації:

вперше:

- запропоновано алгоритм для генерації діалогів українською мовою, що використовує остові дерева як основу та кластеризацію як функцію оцінки результатів;
- надана можливість використовувати даний алгоритм у поєднанні з базою даних товарів.

удосконалено:

- алгоритм генерації діалогу з статичними відповідями на запитання.

Практична значимість одержаних результатів полягає у розробці рішення адаптованого до даних про товари яке з одного боку дозволяє бути впевненим у чітких відповідях що стосуються даних про товар, а з іншого надає більш широку відповідь покупцю у темах, які його цікавлять.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

- 1) Py morphology2 [Електронний ресурс]. – Режим доступу: <https://py morphology2.readthedocs.io/en/stable/#>
- 2) GitHub - brown-uk/nlp_uk: this is a project to demonstrate NLP API from languagetool for ukrainian language. [Електронний ресурс] // GitHub. – Режим доступу: https://github.com/brown-uk/nlp_uk.
- 3) UA-GEC: перший анотований GEC-корпус української мови вже у вільному доступі! [Електронний ресурс] // UA-GEC: перший анотований GEC-корпус української мови вже у вільному доступі!. – Режим доступу: <https://ua-gec-dataset.grammarly.ai/>
- 4) Гнатюк Г. Галактика слова / Галина Гнатюк. – [Б. м.] : дім Дмитра Бураго, 2020.
- 5) Language Models are Few-Shot Learners [Electronic resource] / Tom Brown [et al.] // arXiv.org. – Mode of access: <https://arxiv.org/abs/2005.14165>
- 6) Бісікало О. В. Аналіз морфологічної структури слова на основі асоціативно-статистичному підходу / О. В. Бісікало, І. А. Кравчук // Вісник Вінницького політехнічного інституту. – 2011. – № 4. – С. 134–136.
- 7) Бісікало О. В. Концептуальні алгоритми виокремлення морфем для реалізації інформаційної технології обробки природномовних текстів / О. В. Бісікало, І. А. Кравчук // Інформаційні технології в технічних системах. – 2011. – С. 7–9.
- 8) Klein D. Natural language grammar induction using a constituent-context model / Dan Klein, Christopher Manning // Advances in Neural Information Processing Systems 14. – [S. l.], 2002.
- 9) Turchin A. Using Natural Language Processing to Measure and Improve Quality of Diabetes Care: A Systematic Review [Electronic resource] / Alexander Turchin, Luisa F. Florez Builes // Journal of Diabetes Science and Technology. – 2021. – Vol. 15, no. 3. – P. 553–560. – Mode of access: <https://doi.org/10.1177/19322968211000831>

- 10) Cortés Rodríguez F. Building an RRG computational grammar [Electronic resource] / Francisco Cortés Rodríguez, Ricardo Mairal-Usón // *Onomázein*. – 2016. – No. 34. – P. 86–117. – Mode of access: <https://doi.org/10.7764/onomazein.34.22>.
- 11) Phillips L. The promise of dialogue: The dialogic turn in the production and communication of knowledge / Louise Phillips. – Amsterdam : John Benjamins Pub. Co., 2011.
- 12) Дале Д. Создание простого разговорного чатбота в python [Электронный ресурс] / Давид Дале // Хабр. – Режим доступа: <https://habr.com/ru/post/462333/>
- 13) Гмурман В. Е. Теория вероятностей и математическая статистика / В. Е. Гмурман. – Москва : Высшая школа, 2003.
- 14) Pettie S. An optimal minimum spanning tree algorithm [Electronic resource] / Seth Pettie, Vijaya Ramachandran // *Journal of the ACM*. – 2002. – Vol. 49, no. 1. – P. 16–34. – Mode of access: <https://doi.org/10.1145/505241.505243>.
- 15) Djauhari M. A. Optimality problem of network topology in stocks market analysis [Electronic resource] / Maman Abdurachman Djauhari, Siew Lee Gan // *Physica A: Statistical Mechanics and its Applications*. – 2015. – Vol. 419. – P. 108–114. – Mode of access: <https://doi.org/10.1016/j.physa.2014.09.060>.
- 16) A Hub-Based Labeling Algorithm for Shortest Paths on Road Networks / Ittai Abraham [et al.] // *Symposium on Experimental Algorithms*. – 2011. – P. 230–241.
- 17) SEBASTIAN S. MULTI-FUZZY EXTENSIONS OF FUNCTIONS [Electronic resource] / SABU SEBASTIAN, T. V. RAMAKRISHNAN // *Advances in Adaptive Data Analysis*. – 2011. – Vol. 03, no. 03. – P. 339–350. – Mode of access: <https://doi.org/10.1142/s1793536911000714>.
- 18) Recommending Refactorings to Reverse Software Architecture Erosion [Electronic resource] / Ricardo Terra [et al.] // 2012 16th European Conference on Software

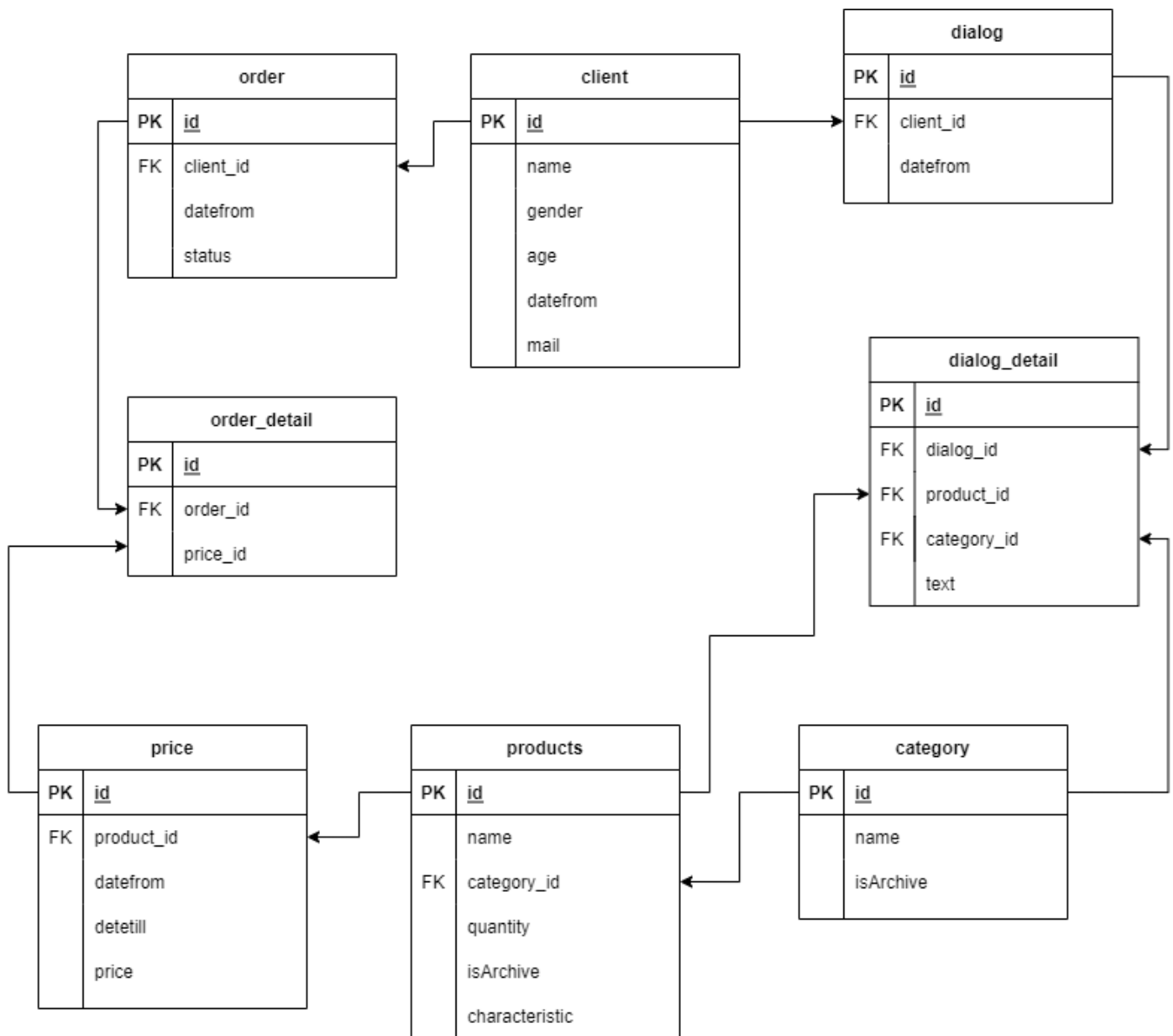
Maintenance and Reengineering (CSMR), Szeged, Hungary, 27–30 March 2012 – [S. l.], 2012. – Mode of access: <https://doi.org/10.1109/csmr.2012.40>.

- 19) Poort E. R. RCDA: Architecting as a risk- and cost management discipline [Electronic resource] / Eltjo R. Poort, Hans van Vliet // Journal of Systems and Software. – 2012. – Vol. 85, no. 9. – P. 1995–2013. – Mode of access: <https://doi.org/10.1016/j.jss.2012.03.071>. – Title from screen.

ДОДАТКИ

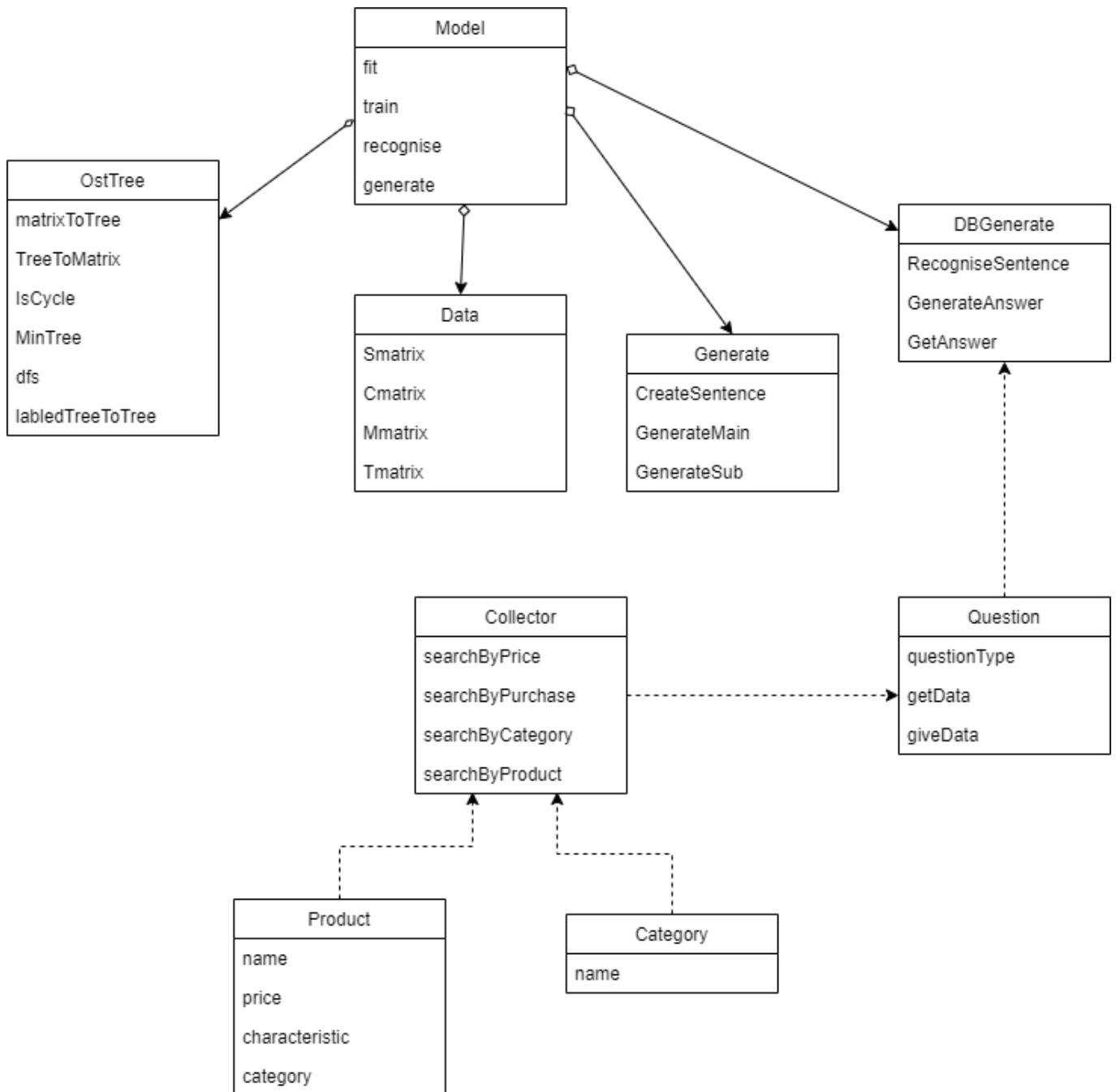
ДОДАТОК А

Схема бази даних



ДОДАТОК Б

UML діаграма класів



ДОДАТОК В

ЛІСТИНГ КОДУ

```
#!/usr/bin/env python
# coding: utf-8

# In[428]:

import numpy as np

class OstTree:
    def __init__(self,matrix):
        self.matrix = matrix
        self.tree = []
        self.temp_tree=[]
        self.v_list = []
        self.p=0

    def matrixToTree(self):
        for i in range(matrix.shape[0]):
            for j in range(i,matrix.shape[1]):
                if matrix[i][j]!=0:
```

```

        self.tree.append([i,j,matrix[i][j]])
self.tree = np.array(self.tree)

def treeToMatrix(self):
    n = max(self.tree[:,0])
    m = max(self.tree[:,1])
    #print(n)
    matrix = np.zeros([int(n)+1,int(m)+1])
    for i in self.tree:
        matrix[int(i[0])][int(i[1])]=i[2]
    return matrix

def minTree(self):
    labeled_tree = np.c_[self.tree,np.zeros([self.tree.shape[0],1])]
    used_v = np.zeros(matrix.shape[0])
    k=0
    while min(used_v)==0:
        #print(labeled_tree)
        #print("")
        min_e = 10000
        min_e_index = -1
        for i in range(labeled_tree.shape[0]):

```

```

if labled_tree[i][3]==0:
    if labled_tree[i][2]<min_e:
        min_e = labled_tree[i][2]
        min_e_index = i
labled_tree[min_e_index][3]=1
if self.isCicle(self.labledTreeToTree(labled_tree,1))==False:
    labled_tree[min_e_index][3]=-1
else:
    used_v[int(labled_tree[int(min_e_index)][0])]=1
    used_v[int(labled_tree[int(min_e_index)][1])]=1

return self.labledTreeToTree(labled_tree,1)

def isCicle(self,tree):
    self.v_list = np.zeros(int(max([max(tree[:,0]),max(tree[:,1])])+1))
    self.temp_tree = np.c_[tree,np.zeros([tree.shape[0],1])]
    self.temp_tree[0][3]=1
    self.v_list[int(self.temp_tree[0][0])]=1
    self.p=0
    if min(self.temp_tree[:,3])==-1:
        return False
    else:

```

```

return self.dfs(int(self.temp_tree[0][1]),int(self.temp_tree[0][0]))

def dfs(self,v,p):
    if self.p==-1:
        return False
    self.v_list[v]=1
    # print("")
    # print(self.temp_tree)

    if min(self.v_list)==1:
        return True

    for i in range(self.temp_tree.shape[0]):
        if self.temp_tree[i][3]==0:
            if self.temp_tree[i][0]==v and self.v_list[int(self.temp_tree[i][1])]==0:
                #print('1-1')
                self.temp_tree[i][3]=1
                self.dfs(int(self.temp_tree[i][1]),int(v))
            elif self.temp_tree[i][1]==v and self.v_list[int(self.temp_tree[i][0])]==0:
                #print('1-2')
                self.temp_tree[i][3]=1
                self.dfs(int(self.temp_tree[i][1]),int(v))
    for i in range(self.temp_tree.shape[0]):

```

```

    if self.temp_tree[i][3]==0:
        if self.temp_tree[i][0]==v and self.temp_tree[i][1]!=p and
self.v_list[int(self.temp_tree[i][1])]==1:
            #print('2-1')
            self.p=-1
            self.temp_tree[i][3]=-1
            return False

        elif self.temp_tree[i][1]==v and self.temp_tree[i][0]!=p and
self.v_list[int(self.temp_tree[i][0])]==1:
            #print('2-2')
            self.p=-1
            self.temp_tree[i][3]=-1
            return False

        elif self.temp_tree[i][0]==v or self.temp_tree[i][1]==v:
            #print('2-3')
            self.temp_tree[i][3]=-1
            self.dfs(v,p)

def labledTreeToTree(self,labled_tree,label):
    tree = []
    for i in labled_tree:
        if i[3]==label:
            tree.append(i[0:3])

```

```
return np.array(tree)
```

```
#!/usr/bin/env python
```

```
# coding: utf-8
```

```
# In[179]:
```

```
import psycopg2 as ps
```

```
from datetime import datetime
```

```
class dbdata:
```

```
    def __init__(self):
```

```
        self.conn = ps.connect(
```

```
            host="localhost",
```

```
            database="diplomadb",
```

```
            user="postgres",
```

```
            password="postgres")
```

```
        self.cur = conn.cursor()
```

```
    def get_product_name(self,product_id):
```

```

self.cur.execute("select name from ""products"" where id = "+str(product_id))
return self.cur.fetchone()[0]

def get_product_cheaper(self,product_id):
    self.cur.execute('select category_id from products where id ='+str(product_id))
    category_id = self.cur.fetchone()[0]
    self.cur.execute('select price from price where product_id ='+str(product_id)+' and
'+""""
                        +datetime.today().strftime('%Y-%m-%d')+""""+' between datefrom and
datetill')
    t= self.cur.fetchone()
    if t==None:
        price=-1
    else:
        price = t[0]
    self.cur.execute("""with t as (
        select prod.name, count(o.id) as q
        from order_detail o
        join price p on o.price_id = p.id
        join products prod on prod.id = p.product_id
        where prod.category_id = ""+str(category_id)+' and
'+""""+datetime.today().strftime('%Y-%m-%d')+""""+
        ""
        between p.datefrom and p.datetill

```

```

and prod.id <> '"+str(product_id)+' and p.price < '+str(price)+
'''

group by prod.name)

select name

from t

where q = (select max(q) from t)

''')

#print(self.cur.fetchone())

r=self.cur.fetchone()

if r == None:

    return "

else:

    return r[0]

def get_product_expensive(self,product_id):

    self.cur.execute('select category_id from products where id =' +str(product_id))

    category_id = self.cur.fetchone()[0]

    self.cur.execute('select price from price where product_id =' +str(product_id)+' and
'+''''

                        +datetime.today().strftime('%Y-%m-%d')+''''+' between datefrom and
datetill')

    t = self.cur.fetchone()

    if t==None:

        price=100000

```

```

else:
    price = t[0]
self.cur.execute("""with t as (
    select prod.name, count(o.id) as q
    from order_detail o
    join price p on o.price_id = p.id
    join products prod on prod.id = p.product_id
    where prod.category_id = '"+str(category_id)+' and
'+"""+datetime.today().strftime('%Y-%m-%d')+"""+
    ""
    between p.datefrom and p.datetill
    and prod.id <> '"+str(product_id)+' and p.price > '+str(price)+
    ""
    group by prod.name)
select name
from t
where q = (select max(q) from t)
""")
r=self.cur.fetchone()
if r == None:
    return "
else:
    return r[0]

```

```

def get_product_popular(self,category_id):
    self.cur.execute("""with t as (
        select prod.name, count(o.id) as q
        from order_detail o
        join price p on o.price_id = p.id
        join products prod on prod.id = p.product_id
        where prod.category_id = '"+str(category_id)+' and
'+"""+datetime.today().strftime('%Y-%m-%d')
        +"""+""between p.datefrom and p.datetill
        group by prod.name)
    select name
    from t
    where q = (select max(q) from t)
    """)
    r=self.cur.fetchone()
    if r == None:
        return "
    else:
        return r[0]

def get_product_id(self,product):
    self.cur.execute("select id from ""products"" where lower(name) =
"+product.lower())
    return self.cur.fetchone()[0]

```

```
def get_category(self,base):
    self.cur.execute("select name from ""category"" where lower(name) like
'% "+base.lower()+"% ' ")
    return self.cur.fetchone()[0]

def get_category_id(self,category):
    self.cur.execute("select id from ""category"" where lower(name) =
"+category.lower())
    return self.cur.fetchone()[0]

def get_category_by_id(self,id):
    self.cur.execute("select name from ""category"" where id = "+str(id))
    return self.cur.fetchone()[0]

def get_product_category(self,product_id):
    self.cur.execute('select category_id from products where id = '+str(product_id))
    return self.cur.fetchone()[0]
```

ДОДАТОК Г

Результати перевірки роботи на співпадіння



Имя пользователя:
Лісовиченко Олег Іванович

ID проверки:
1009451011

Дата проверки:
01.12.2021 12:14:53 EET

Тип проверки:
Doc vs Internet + Library

Дата отчета:
01.12.2021 12:15:24 EET

ID пользователя:
76913

Название файла: IP-301мл_Корольова

Количество страниц: 34 Количество слов: 5552 Количество символов: 45519 Размер файла: 74.53 KB ID файла: 1009466010

13.1% Совпадения

Наибольшее совпадение: 3.28% с источником из Библиотеки (ID файла: 5857472)



0% Цитат

Исключение цитат выключено

Исключение списка библиографических ссылок выключено

0% Исключений

Нет исключенных источников

Модификации

Обнаружены модификации текста. Подробная информация доступна в онлайн-отчете.

Замененные символы 7