Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)
Кафедра програмної інженерії
(повна назва кафедри)

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня

бакалавр
(назва освітнього ступеня)

на тему: Розробка семантичної гри на основі NLP: векторні представлення слів, аналіз та оптимізація алгоритмів

Виконав: студент IV курсу, групи СП-41
спеціальності 121 інженерія програмного забезпечення
(шифр і назва спеціальності)

_____ Кононучук А.В.
(підпис) (прізвище та ініціали)

Керівник _____ Мудрик І.Я.
(підпис) (прізвище та ініціали)

Нормоконтроль _____ _____
(підпис) (прізвище та ініціали)

Завідувач кафедри _____ _____
(підпис) (прізвище та ініціали)

Рецензент _____ _____
(підпис) (прізвище та ініціали)

Тернопіль 2025

Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)

Кафедра програмної інженерії
(повна назва кафедри)

ЗАТВЕРДЖУЮ
Завідувач кафедри

(підпис) (прізвище та ініціали)

« » 2025 р.

З А В Д А Н Н Я
НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття освітнього ступеня Бакалавр
(назва освітнього ступеня)

за спеціальністю 121 Інженерія програмного забезпечення
(шифр і назва спеціальності)

студенту Конончуку Андрію Володимировичу
(прізвище, ім'я, по батькові)

1. Тема роботи Розробка семантичної гри на основі NLP: векторні представлення слів, аналіз та оптимізація алгоритмів

Керівник роботи Мудрик Іван Ярославович
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Затверджені наказом ректора від «___» _____ 20___ року № _____

2. Термін подання студентом завершеної роботи

3. Вихідні дані до роботи корпус з ~14 000 українських іменників , моделі OpenAI (text-embedding-3-large , gpt-3.5-turbo) та метод косинусної схожості для розрахунку семантичних рейтингів.

4. Зміст роботи (перелік питань, які потрібно розробити)
Вступ. 1. Аналіз предметної області семантичних ігор та технологій NLP, постановка задачі розробки гри "Словозв'яз" на основі векторних представлень слів. 2. Проєктна частина розробки гри "Словозв'яз": обґрунтування вибору моделей та алгоритмів, проєктування архітектури та програмних модулів. 3. Практична частина: реалізація системи, розробка підсистеми генерації семантичних рейтингів, аналіз ефективності та оптимізація алгоритмів. 4. Безпека життєдіяльності, основи охорони праці. Висновки. Перелік джерел. Додатки.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, слайдів)

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Безпека життєдіяльності, основи охорони праці	Лазарюк В.В.		

7. Дата видачі завдання _____

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1	Ознайомлення з завданням до кваліфікаційної роботи	29.01.2025	*Виконано*
2	Аналіз існуючих семантичних ігор та моделей векторних представлень слів	30.01.2025–02.02.2025	*Виконано*
3	Розробка підсистеми генерації ембеддінгів та семантичних рейтингів	03.02.2025–07.02.2025	*Виконано*
4	Реалізація архітектури клієнт-сервер гри «Словозв'яз» (frontend/backend)	10.02.2025–16.02.2025	*Виконано*
5	Оформлення розділу «Аналіз предметної області та постановка задачі»	17.02.2025–19.02.2025	*Виконано*
6	Оформлення розділу «Теоретичні основи NLP для реалізації гри»	20.02.2025–23.02.2025	*Виконано*
7	Оформлення розділу «Проєктування та реалізація гри “Словозв'яз”»	24.02.2025–28.02.2025	*Виконано*
8	Оформлення розділу «Порівняльний аналіз та результати»	03.03.2025–06.03.2025	*Виконано*
9	Оформлення розділу «Безпека життєдіяльності, основи охорони праці»	07.04.2025–09.04.2025	*Виконано*
10	Оформлення повної кваліфікаційної роботи		
11	Нормоконтроль		
12	Перевірка на плагіат		
13	Попередній захист кваліфікаційної роботи Захист кваліфікаційної роботи		
14	Захист кваліфікаційної роботи		

Студент _____ Конончук А.В.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Мудрик І.Я.
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Розробка семантичної гри на основі NLP: векторні представлення слів, аналіз та оптимізація алгоритмів. Кваліфікаційна робота бакалавра, спец. 121 Інженерія програмного забезпечення. ТНТУ ім. І. Пулюя, Тернопіль, 2025.

Робота: 63 с., 12 рис., 2 додат., 23 джерела.

Ключові слова: семантична гра, NLP, векторні представлення слів, OpenAI, text-embedding-3-large, gpt-3.5-turbo, косинусна схожість, веб-додаток, українська мова, Словозв'яз.

Мета – розробка україномовної семантичної гри «Словозв'яз» на основі сучасних NLP-моделей (OpenAI text-embedding-3-large, gpt-3.5-turbo) та семантичних визначень, з порівняльним аналізом ефективності підходу.

Об’єкт: програмно-алгоритмічні засоби та процес розробки україномовної семантичної гри.

Предмет: методи отримання векторних представлень слів у поєднанні з генерованими семантичними визначеннями, алгоритми семантичної схожості, архітектура та реалізація системи «Словозв'яз», аналіз якості та ефективності.

У роботі проаналізовано існуючі семантичні ігри, обґрунтовано актуальність створення україномовного аналога, розглянуто теоретичні основи NLP. Спроєктовано та розроблено клієнт-серверний веб-додаток «Словозв'яз» з підсистемою підготовки даних. Проведено аналіз якості семантичних рейтингів, порівняно підхід з Word2Vec/GloVe, оцінено оптимізації.

Наукова новизна: Вперше розроблено україномовну семантичну гру «Словозв'яз» на базі OpenAI text-embedding-3-large та генерованих визначень; запропоновано підхід до генерації семантичних рейтингів, що фокусується на основному значенні слова.

Практичне значення: Розроблено функціональний веб-додаток, що сприяє розширенню україномовного цифрового контенту.

ANNOTATION

Development of a semantic game based on NLP: word embeddings, analysis, and algorithm optimization. Bachelor's Qualification Thesis, specialty 121 Software Engineering. Ternopil Ivan Puluj National Technical University, Ternopil, 2025.
Thesis: 62 p., 12 fig., 2 append., 23 ref.

Keywords: semantic game, NLP, word embeddings, OpenAI, text-embedding-3-large, gpt-3.5-turbo, cosine similarity, web application, Ukrainian language, Slovozviaz.

The purpose is to develop a Ukrainian-language semantic game "Slovozviaz" based on modern NLP models (OpenAI text-embedding-3-large, gpt-3.5-turbo) and semantic definitions, with a comparative analysis of the approach's effectiveness.

Object: software-algorithmic tools and the development process of a Ukrainian-language semantic game.

Subject: methods for obtaining word embeddings combined with generated semantic definitions, semantic similarity algorithms, architecture and implementation of the "Slovozviaz" system, quality and efficiency analysis.

The thesis analyzes existing semantic games, substantiates the relevance of creating a Ukrainian-language analogue, and reviews theoretical NLP foundations. A client-server web application "Slovozviaz" with a data preparation subsystem was designed and developed. The quality of semantic ratings was analyzed, the approach was compared with Word2Vec/GloVe, and optimizations were evaluated.

Scientific novelty: For the first time, a Ukrainian-language semantic game "Slovozviaz" based on OpenAI text-embedding-3-large and generated definitions has been developed; an approach for generating semantic ratings focusing on the primary word meaning is proposed.

Practical significance: A functional web application contributing to the expansion of Ukrainian-language digital content has been developed.

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, СКОРОЧЕНЬ І ТЕРМІНІВ

API – Інтерфейс прикладного програмування (Application Programming Interface) BERT – Двонаправлене представлення кодувальника від трансформерів (Bidirectional Encoder Representations from Transformers).

CBOW – Безперервний мішок слів (Continuous Bag-of-Words).

ELMo – Векторні представлення з мовних моделей (Embeddings from Language Models).

Flask – Мікрофреймворк для веб-додатків на мові Python GloVe – Глобальні вектори для представлення слів (Global Vectors for Word Representation).

GPT – Генеративний попередньо навчений трансформер (Generative Pre-trained Transformer).

Gunicorn – WSGI HTTP-сервер для Python

JSON – Нотація об'єктів JavaScript (JavaScript Object Notation) LLM – Велика мовна модель (Large Language Model).

MIRACL – Багатомовний інформаційний пошук у континуумі мов (Multilingual Information Retrieval Across a Continuum of Languages), бенчмарк

MTEB – Масовий бенчмарк для текстових ембеддінгів (Massive Text Embedding Benchmark) NLP – Обробка природної мови (Natural Language Processing).

ORM – Об'єктно-реляційне відображення (Object-Relational Mapper)
SQLAlchemy – Бібліотека Python, що надає інструментарій SQL та ORM SQLite Вбудована реляційна система керування базами даних SPA – Односторінковий додаток (Single Page Application).

Word2Vec – Алгоритм для отримання векторних представлень слів WSGI – Інтерфейс шлюзу веб-сервера (Web Server Gateway Interface).

ЗМІСТ

ВСТУП

Сучасний етап розвитку інформаційних технологій характеризується стрімким зростанням обсягів текстових даних та прогресом у галузі обробки природної мови (Natural Language Processing, NLP). Одним із перспективних напрямків застосування NLP є створення інтелектуальних ігор, які використовують семантичні властивості мови для формування унікального ігрового досвіду. Ігри, що базуються на вгадуванні слів за їхнім значенням чи семантичною близькістю, набувають значної популярності у світі (наприклад, Contexto.me, Semantle.com).

Актуальність теми. Аналіз існуючих популярних семантичних ігор показує, що вони переважно використовують алгоритми векторних представлень слів (Word2Vec, GloVe), розроблені близько десятиліття тому. Ці алгоритми, хоча й ефективні, мають відомі обмеження у точному моделюванні семантики та обробці полісемії. Водночас, поява нових потужних моделей ембеддінгів на базі архітектури Transformer, таких як моделі OpenAI, відкриває можливості для створення семантичних ігор з потенційно вищою точністю ранжування. Разом з тим, спостерігається зростання попиту на якісний україномовний цифровий контент, зокрема ігровий. На даний момент відчувається брак україномовних аналогів семантичних ігор, які б використовували сучасні досягнення NLP. Таким чином, розробка україномовної семантичної гри «Словозв'яз» із застосуванням новітніх моделей ембеддінгів та дослідженням підходу, що використовує семантичні визначення для фокусування значення, є актуальним науково-технічним завданням.

Мета і задачі дослідження. Метою даної дипломної роботи є розробка україномовної семантичної гри-головоломки «Словозв'яз» з використанням сучасної моделі векторних представлень слів OpenAI text-embedding-3-large та семантичних визначень, а також проведення порівняльного аналізу ефективності та характеру семантичної близькості, що генерується цим підходом, у зіставленні з методами Word2Vec та GloVe. Для досягнення поставленої мети було вирішено наступні задачі:

- Провести аналіз теоретичних основ векторних представлень слів та методів вимірювання семантичної схожості.
- Сформувати корпус українських іменників та розробити методику генерації семантичних визначень.
- Реалізувати підсистему генерації семантичних рейтингів на основі моделі text-embedding-3-large та косинусної схожості.
- Спроектувати та розробити програмну систему гри «Словозв'яз» (backend та frontend).
- Провести порівняльний аналіз характеру семантичної близькості різних підходів (Word2Vec, GloVe, OpenAI+визначення).
- Проаналізувати ефективність застосованих оптимізаційних рішень.

Об'єкт дослідження. Процес розробки та функціонування семантичної гри, що базується на векторних представленнях слів, отриманих за допомогою сучасних NLP-моделей та семантичних визначень.

Предмет дослідження. Методи отримання векторних представлень слів (Word2Vec, GloVe, OpenAI text-embedding-3-large + визначення), алгоритми розрахунку семантичної схожості (косинусна міра), архітектура та програмна реалізація системи «Словозв'яз», аналіз якості та ефективності застосованого підходу у порівнянні з аналогами.

Методи дослідження. У роботі використано наступні методи: аналіз науково-технічної літератури (для дослідження існуючих підходів до векторних представлень та семантичних ігор), порівняльний аналіз (для зіставлення характеристик алгоритмів Word2Vec, GloVe та розробленого підходу), системний аналіз (для проектування архітектури системи), об'єктно-орієнтоване проєктування та програмування (для реалізації програмних компонентів), експериментальне дослідження (для генерації семантичних рейтингів та оцінки їх якості), методи оптимізації (кешування, батчінг, прекомпіляція), аналіз результатів.

Наукова новизна одержаних результатів.

- Вперше розроблено україномовну семантичну гру «Словозв'яз», що використовує сучасну модель векторних представлень OpenAI text-embedding-3-large.
- Запропоновано та апробовано підхід до генерації семантичних рейтингів, що поєднує потужну модель ембеддінгів із зовнішньо згенерованими семантичними визначеннями для фокусування на основному значенні слова.
- Проведено порівняльний аналіз характеру семантичної схожості, що генерується підходами Word2Vec, GloVe та запропонованим методом, в контексті їх застосування для семантичних ігор.
- Отримано практичні дані щодо якості рейтингів та ефективності оптимізаційних технік при роботі з сучасними NLP-моделями для української мови.

Практичне значення одержаних результатів. Практичне значення роботи полягає у створенні готового до використання веб-додатку – україномовної гри «Словозв'яз», яка може бути використана як інтелектуальна розвага та інструмент для дослідження семантичних зв'язків мови. Розроблений програмний каркас та результати аналізу можуть бути використані при створенні інших NLP-орієнтованих додатків українською мовою. Робота робить внесок у розширення україномовного цифрового контенту.

Публікації.

Основні положення та результати даної кваліфікаційної роботи було опубліковано у вигляді тез у збірнику матеріалів Міжнародної науково-технічної конференції "Фундаментальні та прикладні проблеми сучасних технологій", присвяченій 180-річчю з дня народження Івана Пулюя та 65-річчю з дня заснування Тернопільського національного технічного університету імені Івана Пулюя.

РОЗДІЛ 1. АНАЛІЗ СЕМАНТИЧНИХ ІГОР ТА ПОСТАНОВКА ЗАДАЧІ РОЗРОБКИ ГРИ "СЛОВОЗВ'ЯЗ"

1.1 Аналіз існуючих семантичних ігор на основі векторних представлень слів

Останніми роками значної популярності набули інтелектуальні ігри, які використовують методи обробки природної мови (NLP) для створення унікального ігрового досвіду, заснованого на семантичній близькості слів. Замість традиційних завдань на орфографію чи визначення, ці ігри пропонують гравцям вгадувати слова, орієнтуючись на їхнє значення або контекст використання відносно секретного слова. Аналіз двох провідних представників цього жанру, Contexto.me та Semantle.com, дозволяє зрозуміти поточні підходи та виявити потенційні напрямки для вдосконалення.

1.1.1 Contexto.me: Схожість на основі контексту використання (GloVe)

Гра Contexto.me пропонує гравцям знайти секретне слово дня, маючи необмежену кількість спроб. Ключовою особливістю є механізм зворотного зв'язку: після введення слова-здогадки гра показує його позицію (ранг) у списку, відсортованому за ступенем схожості до секретного слова, яке завжди має ранг 1 (рис. 1.1).

Згідно з офіційною інформацією гри, порядок слів визначається за допомогою алгоритму штучного інтелекту, який аналізує тисячі текстів. В основі цього алгоритму лежить модель GloVe (Global Vectors for Word Representation) [1]. Цей підхід використовує глобальну статистику спільного вживання слів у великому текстовому корпусі для побудови векторних представлень. Дані для розрахунку близькості слів, як зазначено на сайті гри, були отримані від Stanford NLP Group [1].

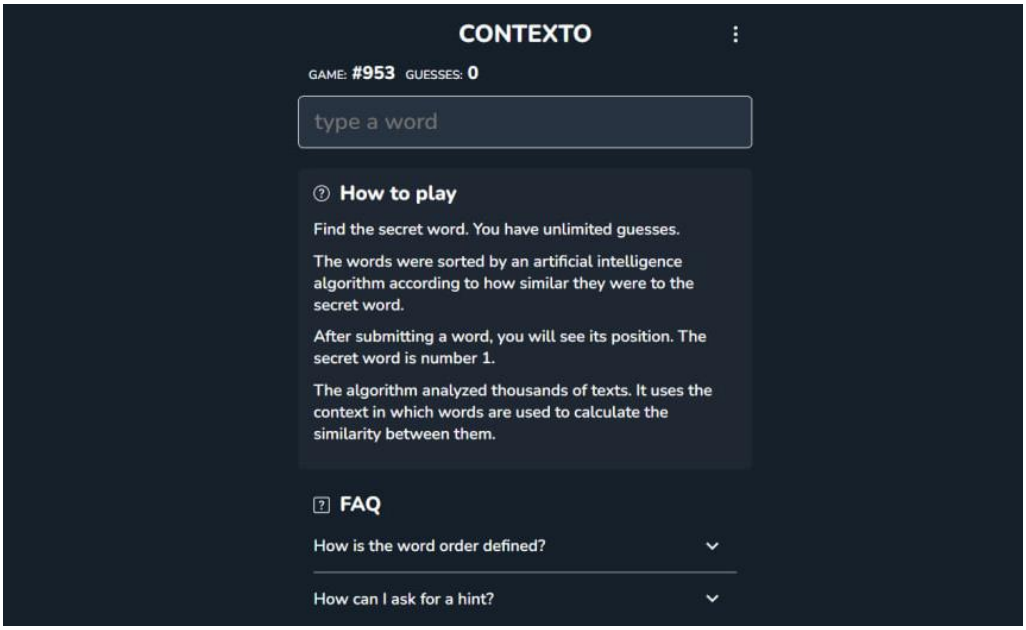

Рисунок 1.1 – Головний екран гри Contexto.me

Важливою характеристикою Contexto є те, що алгоритм визначає схожість слів не стільки за їх прямим лексичним значенням, скільки за близькістю контекстів, у яких ці слова використовуються в проаналізованих текстах. Як приклад, розробники наводять ситуацію, коли для секретного слова "нескінченний" близькими можуть виявитися як слова, пов'язані з "космосом", так і слова, пов'язані з "коханням", оскільки слово "нескінченний" часто вживається в обох цих різних контекстах. Це означає, що навіть синоніми (як "tv" та "television") можуть мати дуже різні ранги, якщо вони типово вживаються у різних контекстах відносно секретного слова дня. Такий підхід вимагає від гравця не тільки знання синонімів, але й інтуїтивного розуміння того, як слова функціонують разом у мові.

1.1.2 Semantle.com: Схожість на основі значення (Word2Vec)

Semantle.com, створена Девідом Тернером, також є грою на вгадування секретного слова з необмеженою кількістю спроб. На відміну від рангу в Contexto, Semantle надає гравцеві числову оцінку семантичної схожості здогадки до секретного слова (від -100 до 100), а також повідомляє, коли здогадка потрапляє до 1000 найближчих слів (рис. 1.2).

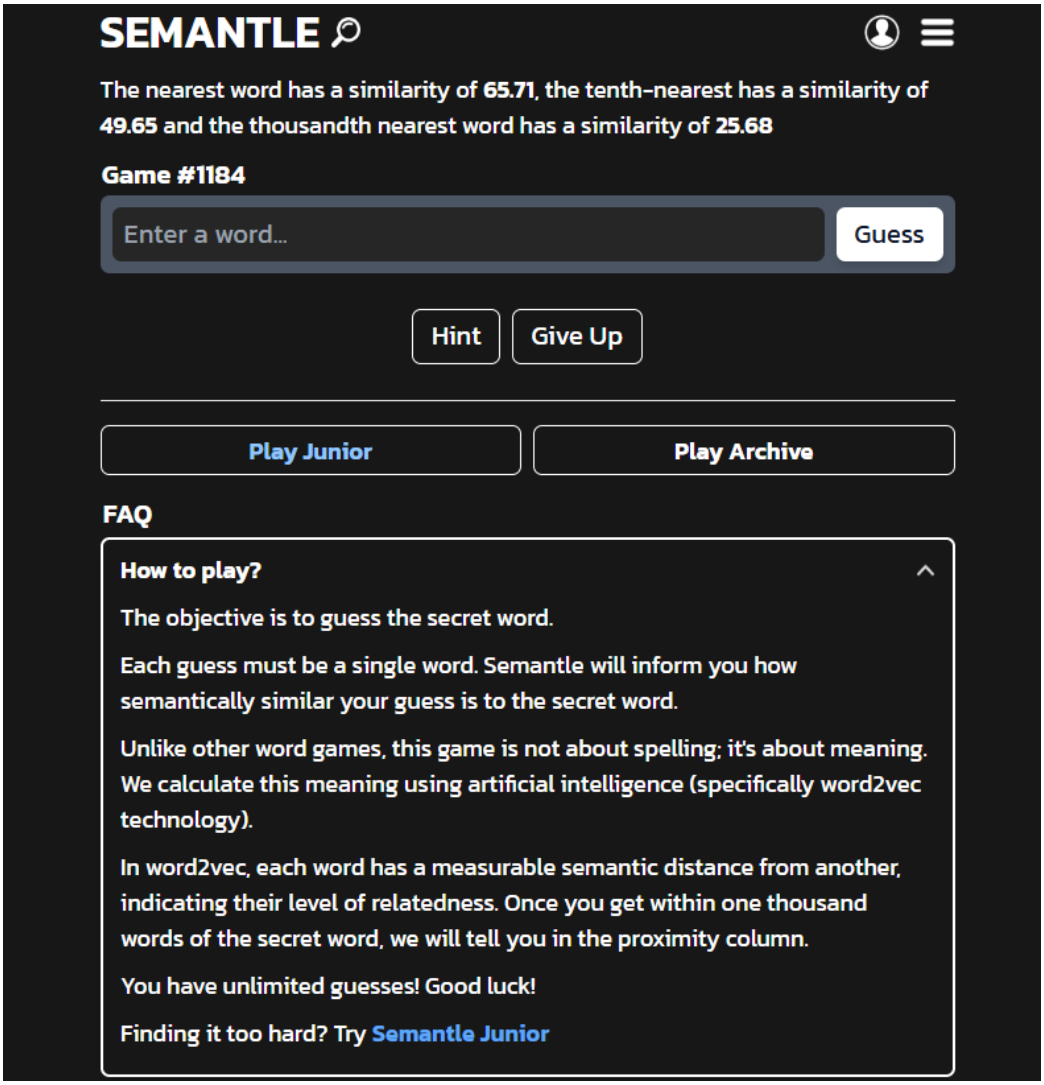

Рисунок 1.1 – Головний екран гри Semantle.com

Гра позиціонує себе як така, що фокусується саме на значенні (семантиці) слів, а не на орфографії. Для розрахунку схожості використовується технологія штучного інтелекту, а саме алгоритм Word2Vec [2]. Цей алгоритм, згідно з описом гри, визначає значення слова на основі слів, з якими воно часто зустрічається разом ("друзів" слова), перетворюючи кожне слово на векторне представлення (серію чисел) для подальшого порівняння [2].

Хоча Semantle наголошує на значенні, особливості алгоритму Word2Vec, який навчається на локальних контекстах (слова, що знаходяться поруч), призводять до певних ефектів. Наприклад, як зазначається у підказках до гри, антоніми (як "гарячий" та "холодний") можуть отримати високу оцінку схожості, оскільки вони часто зустрічаються у схожих контекстах (наприклад, "Чай занадто _"). Також

згадується, що алгоритм навчався на текстах газет, тому типові для цього стилю слова можуть бути вдалими здогадками, що вказує на вплив навчального корпусу на результати ранжування.

Аналіз цих двох популярних ігор показує, що хоча обидві використовують векторні представлення слів для моделювання семантичної близькості, вибір конкретного алгоритму (GloVe проти Word2Vec) та, ймовірно, навчальних даних призводить до різних інтерпретацій "схожості" (контекст використання проти схожість оточення), що безпосередньо впливає на ігровий процес та стратегії гравців. Це створює підґрунтя для дослідження альтернативних, потенційно більш точних підходів до семантичного ранжування в рамках розробки гри "Словозв'яз".

1.2 Актуальність розробки україномовної гри "Словозв'яз" з використанням сучасних NLP-моделей та аналізу впливу семантичних визначень

Аналіз популярних семантичних ігор Contexto.me та Semantle.com (п. 1.1), що базуються на моделях GloVe [1] та Word2Vec [2] відповідно, виявляє їхню залежність від технологій векторних представлень слів, розроблених майже десятиліття тому. Як показують дослідження, зокрема робота Малиги та Шматкова [4], незважаючи на ефективність цих "традиційних" методів для певних завдань, проблеми точного відтворення семантики та контексту при обробці текстів залишаються актуальними і потребують подальшого вивчення. Статична природа векторів у Word2Vec та GloVe (один вектор на слово) та їхня орієнтація на контексти вживання [1, 2] обмежують здатність цих моделей повною мірою враховувати багатозначність слів та тонкі семантичні відтінки, що особливо помітно у складних чи багатозначних сценаріях [4].

З огляду на ці обмеження та стрімкий розвиток технологій NLP [4], використання сучасних моделей ембеддінгів, таких як text-embedding-3-large від OpenAI [5], стає актуальним напрямком для дослідження. Ці моделі, побудовані на

новітніх архітектурах, потенційно здатні генерувати векторні представлення, що краще відображають семантичну суть слів та їх зв'язки у порівнянні з підходами попереднього покоління [5].

Водночас, проблема полісемії залишається викликом. У даній роботі застосовується підхід, що поєднує сучасну модель ембеддінгів із використанням одного семантично насиченого визначення для кожного слова. Актуальність розробки гри "Словозв'яз" із таким підходом полягає в наступному:

1. Задоволення попиту на україномовний контент: Створення сучасної інтелектуальної гри українською мовою заповнює нішу на ринку цифрових продуктів [3].

2. Дослідження сучасних моделей ембеддінгів: Робота дозволяє на практиці оцінити ефективність та характер семантичної схожості, що генерується моделлю text-embedding-3-large [5], у порівнянні з Word2Vec та GloVe [1, 2], які мають відомі обмеження у роботі з семантикою та контекстом [4].

3. Аналіз впливу сфокусованих визначень: Дослідження того, як використання одного визначення для слова впливає на результати ранжування (особливо для полісемічних слів), порівняно з методами, що усереднюють всі значення слова на основі контекстів вживання.

4. Потенціал для специфічного ігрового досвіду: Аналіз того, наскільки ігровий процес, базований на близькості до одного, конкретизованого значення слова, є інтуїтивним та цікавим для користувача.

Таким чином, розробка "Словозв'язу" є актуальною не тільки як створення затребуваного україномовного продукту, але і як дослідження застосування сучасних NLP-технологій для вирішення завдань, де традиційні методи стикаються з труднощами у точному моделюванні семантики та контексту [4], а також як аналіз впливу специфічного методу (додавання визначень) на кінцевий результат у семантичній грі.

РОЗДІЛ 2. ТЕОРЕТИЧНІ ОСНОВИ ТА МЕТОДИ NLP ДЛЯ РЕАЛІЗАЦІЇ СЕМАНТИЧНОЇ ГРИ

2.1 Векторні представлення слів

Основою для аналізу семантичної близькості слів та функціонування ігор на кшталт «Словозв'яз» є технологія векторних представлень слів, або ж ембеддінгів (word embeddings). Це сукупність методів моделювання мови та навчання ознак в обробці природної мови (NLP), де слова або фрази зі словника відображаються у вигляді векторів дійсних чисел у багатовимірному просторі [6]. Головна ідея полягає в тому, щоб представити слова у такій формі, яка була б зрозумілою для комп'ютерних алгоритмів, і при цьому зберегти або навіть явно закодувати семантичні та синтаксичні відношення між ними. У такому векторному просторі слова зі схожим значенням або схожим контекстом використання розташовуються близько одне до одного, що дозволяє кількісно вимірювати семантичну близькість за допомогою метрик відстані, таких як косинусна схожість [6]. Цей підхід став фундаментальним для багатьох сучасних завдань NLP, від машинного перекладу та аналізу тональності до інформаційного пошуку та, як у даній роботі, створення семантичних ігор.

2.1.1 Концепція семантичних векторних просторів

В основі методів векторних представлень лежить концепція семантичного векторного простору (semantic vector space). Ідея полягає у тому, що значення слова можна представити не ізольовано, а через його відношення з іншими словами у мові. Замість того, щоб працювати з текстовими рядками безпосередньо, слова відображаються як вектори (точки) у багатовимірному математичному просторі таким чином, що їхнє взаємне розташування в цьому просторі відображає семантичні відношення між ними [6].

Фундаментальним принципом, що лежить в основі багатьох алгоритмів побудови таких просторів, є гіпотеза дистрибутивної семантики. Ця гіпотеза, яку часто формулюють як "слово характеризується своїм оточенням" ("a word is

characterized by the company it keeps"), була запропонована ще Дж. Р. Фертом (J. R. Firth) у 1957 році [7]. Вона стверджує, що слова, які зустрічаються у схожих контекстах у великих обсягах тексту, ймовірно, мають схожі значення. Алгоритми навчання векторних представлень, такі як Word2Vec [1] та GloVe [2], використовують цю гіпотезу, аналізуючи статистику вживання слів у великих текстових корпусах для побудови векторів [6].

Таким чином, створюється простір, де кожне слово представлене вектором фіксованої розмірності (наприклад, 100, 300 або більше вимірів), а відстань (або кут) між векторами корелює зі ступенем семантичної близькості відповідних слів [6]. Слова-синоніми ("автомобіль", "машина") матимуть близькі вектори, тоді як вектори слів з різними значеннями ("автомобіль", "яблуко") будуть розташовані далі один від одного. Ця властивість семантичних векторних просторів дозволяє застосовувати математичні операції до векторів для виявлення аналогій (наприклад, "король" - "чоловік" + "жінка" ≈ "королева") та кількісного вимірювання схожості слів, що є ключовим для реалізації семантичних ігор [6] (рис. 2.1).

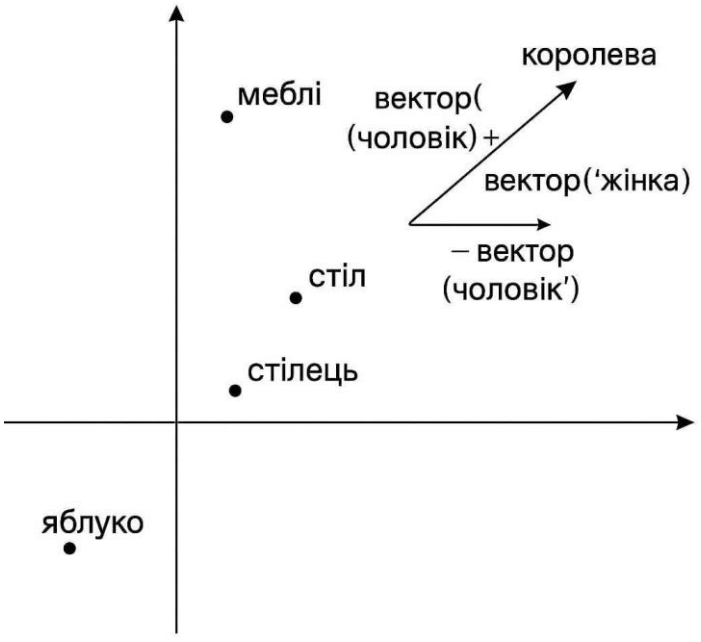

Рисунок 2.1 – Ілюстрація концепції семантичного векторного простору та векторної аналогії

2.1.2 Моделі статичних ембеддінгів

Ранні та фундаментальні підходи до створення векторних представлень слів належать до категорії статичних ембеддінгів. Ключовою особливістю цих моделей є те, що кожному унікальному слову зі словника присвоюється один єдиний вектор фіксованої розмірності, незалежно від контексту, в якому це слово може з'являтися в тексті [6]. Цей вектор намагається усереднити або агрегувати всі можливі значення та варіанти використання слова. Найбільш відомими та впливовими моделями статичних ембеддінгів є Word2Vec та GloVe.

Word2Vec

Модель Word2Vec, представлена Міколовим та колегами у 2013 році [1], стала однією з перших широко використовуваних нейромережевих моделей для ефективного навчання високоякісних векторних представлень слів. Вона базується на дистрибутивній гіпотезі [7] і навчається на великих текстових корпусах, аналізуючи локальний контекст слова – тобто слова, що безпосередньо його оточують [4, с. 53]. Word2Vec існує у двох основних архітектурах [1, 6]:

- CBOW (Continuous Bag-of-Words): Модель намагається передбачити поточне слово на основі його контексту (оточуючих слів).
- Skip-gram: Модель, навпаки, намагається передбачити контекст (оточуючі слова) на основі поточного слова. Цей підхід зазвичай вважається ефективнішим для рідкісних слів.

Word2Vec продемонстрував здатність вловлювати тонкі семантичні та синтаксичні відношення між словами, включаючи аналогії [4, с. 53]. Однак, як статична модель, він має обмеження: один вектор для слова не дозволяє розрізняти його різні значення (полісемію), а орієнтація на локальний контекст може призводити до семантичного зближення слів, що часто зустрічаються у схожому оточенні, навіть якщо їхні значення протилежні (наприклад, антоніми) [4, с. 50, 53]. Саме цей алгоритм використовується у грі Semantle [2].

GloVe (Global Vectors for Word Representation)

Модель GloVe, розроблена Пеннінгтоном, Сокером та Меннінгом у Стенфордському університеті у 2014 році [2], пропонує альтернативний підхід до

навчання статичних ембеддінгів. На відміну від Word2Vec, який фокусується на локальних контекстних вікнах, GloVe використовує глобальну статистику спільного вживання слів у всьому корпусі [2, 4, с. 52]. Модель навчається безпосередньо на агрегованій матриці спільної зустрічальності слів, намагаючись відтворити співвідношення ймовірностей їхнього спільного вживання.

Основна ідея GloVe полягає в тому, що співвідношення у статистиці спільного вживання слів можуть нести більше семантичної інформації, ніж прості частоти локальних контекстів [2]. У експериментах, проведених Малигою та Шматковим [4, с. 53], GloVe показав вищу точність на задачі класифікації сентименту порівняно з Word2Vec для використаного датасету, що автори пов'язали саме зі здатністю GloVe краще агрегувати глобальну статистичну інформацію [4, с. 53]. Цей підхід лежить в основі гри Contexto [1].

Хоча GloVe також є моделлю статичних ембеддінгів і має притаманні їм обмеження (один вектор на слово, труднощі з полісемією), його опора на глобальну статистику може давати переваги у моделюванні певних типів семантичних зв'язків порівняно з Word2Vec [4, с. 53].

2.1.3 Сучасні моделі великих ембеддінгів: OpenAI text-embedding-3-large

Обмеження статичних моделей ембеддінгів, таких як Word2Vec [1] та GloVe [2], зокрема їхня нездатність повною мірою обробляти полісемію (оскільки кожному слову відповідає лише один вектор, що усереднює всі його значення) [4, с. 50, 53], стимулювали розробку нових підходів. Сучасні моделі векторних представлень, часто побудовані на базі архітектури Transformer [8], намагаються адресувати цю проблему шляхом генерації контекстуальних ембеддінгів, які можуть змінювати вектор слова залежно від його оточення. Однак, ефективність вирішення проблеми полісемії залежить не тільки від моделі, але й від способу її застосування.

На відміну від статичних, контекстуальні моделі створюють динамічні векторні представлення для кожного екземпляра слова в тексті, враховуючи його оточення. Наприклад, слово "банк" матиме різний вектор у реченнях "Я поклав гроші в банк" та "Я сидів на березі річки біля банку". Це досягається завдяки

механізмам уваги (attention), зокрема self-attention у Transformer-архітектурі, які дозволяють моделі зважувати важливість різних слів у контексті при формуванні вектора для цільового слова. Такі моделі, як BERT [9] та ELMo [10], продемонстрували значне покращення результатів у багатьох задачах NLP саме завдяки врахуванню контексту.

Одними з найсучасніших та найпотужніших представників моделей для генерації ембеддінгів є моделі, розроблені компанією OpenAI. У даній роботі використовується модель text-embedding-3-large. Це третє покоління ембеддінг-моделей OpenAI, представлене як найбільш продуктивне у лінійці. Ключові характеристики та потенціал цієї моделі включають:

- Висока продуктивність: Модель демонструє значно кращі результати на стандартних бенчмарках порівняно з попередньою популярною моделлю text-embedding-ada-002. Наприклад, на бенчмарку MTEB середня оцінка зросла з 61.0% до 64.6%, а на багатомовному MIRACL – з 31.4% до 54.9%.
- Велика розмірність: text-embedding-3-large генерує вектори з розмірністю до 3072, що потенційно дозволяє кодувати більше семантичних нюансів порівняно з моделями меншої розмірності (наприклад, 1536 у ada-002).
- Гнучкість розмірності: Важливою особливістю моделей третього покоління є підтримка скорочення розмірності вектора без значної втрати його концептуальних властивостей. Наприклад, вектор text-embedding-3-large можна скоротити до 1536, 1024 або навіть 256 вимірів, при цьому версія на 256 вимірів може перевершувати ada-002 на 1536 вимірів. Це дає гнучкість у виборі балансу між точністю, обчислювальними ресурсами та вимогами систем зберігання векторів.
- Широкий контекст: Модель може обробляти вхідні послідовності довжиною до 8191 токенів.
- Потенціал застосування: Завдяки кращому розумінню семантики та контексту, такі потужні ембеддінги мають високий потенціал для покращення якості семантичного пошуку, кластеризації текстів, систем рекомендацій,

класифікації, виявлення аномалій та інших завдань NLP, включаючи реалізацію семантичних ігор.

Використання text-embedding-3-large у грі «Словозв'яз» дозволяє дослідити, наскільки переваги сучасних контекстуальних моделей можуть трансформуватися у більш точне та інтуїтивно зрозуміле семантичне ранжування порівняно з підходами на базі статичних моделей Word2Vec та GloVe.

2.1.4 Використаний підхід: Посилення семантики за допомогою генерованих визначень

Як було зазначено раніше, і статичні моделі (Word2Vec [1], GloVe [2]), і навіть сучасні контекстуальні моделі [8, 9, 10] стикаються з викликами при роботі з полісемією та точним моделюванням семантики [4]. Крім того, під час попередніх експериментів з моделлю text-embedding-3-large [5, 11] для української мови було виявлено, що при поданні на вхід лише окремих слів, модель іноді надавала високу схожість не тільки семантично близьким словам, але й словам зі схожою морфологічною структурою, навіть якщо їхні значення суттєво відрізнялися (наприклад, "рік" та "рик"). Така поведінка є небажаною для семантичної гри, де гравці очікують на близькість саме за значенням.

З огляду на ці обмеження та спостереження, у рамках розробки гри «Словозв'яз» було застосовано специфічний підхід, спрямований на фокусування саме семантичного представлення слова та зменшення впливу морфологічної схожості. Замість того, щоб покладатися лише на векторне представлення самого слова, отримане за допомогою сучасної моделі text-embedding-3-large [5, 11], було вирішено доповнити або замінити слово його коротким, семантично насиченим визначенням. Ці визначення були згенеровані для кожного слова з підготовленого корпусу іменників за допомогою великої мовної моделі gpt-3.5-turbo, з інструкціями надавати лаконічне пояснення основного значення слова, уникаючи його повторення у самому визначені.

Основна гіпотеза цього підходу полягає в тому, що передача на вхід моделі text-embedding-3-large не самого слова (яке може мати морфологічних "омонімів" або різні значення), а його визначення, допоможе скерувати модель на генерацію

вектора, що відображає саме лексичне значення. Розрахунок полягав у тому, що векторне представлення, згенероване на основі тексту визначення, буде більш стабільно відображати семантику слова, ніж вектор, отриманий для ізольованого слова (де могла домінувати морфологія) або з усередненням усіх його значень, як у статичних моделях [1, 2].

Важливо зазначити, що такий підхід, хоча й допомагає сфокусуватися на значенні та зменшити небажану морфологічну близькість, не вирішує проблему полісемії повністю, оскільки для кожного слова використовується лише одне визначення, яке може не охоплювати всі можливі значення. Проте, це дозволяє створити векторний простір, де близькість розраховується відносно конкретизованого значення слова. У процесі генерації семантичних рейтингів для гри «Словозв'яз» саме текст визначення (якщо він був успішно згенерований для слова) передавався моделі text-embedding-3-large [5, 11] для отримання відповідного вектора. Детальний аналіз якості та особливостей рейтингів, отриманих за допомогою цього підходу, буде представлено у Розділі 4 даної роботи.

2.2 Косинусна схожість як міра близькості у векторному просторі

Після того, як слова або їхні визначення представлені у вигляді векторів у багатовимірному семантичному просторі (як описано в п. 2.1), виникає потреба у кількісній оцінці ступеня їхньої близькості. Однією з найпоширеніших та ефективних метрик для вимірювання схожості між векторами у високорозмірних просторах, особливо в контексті обробки природної мови та векторних представлень слів, є косинусна схожість (cosine similarity) [6].

Косинусна схожість вимірює косинус кута між двома ненульовими векторами [13]. Вона визначає орієнтацію векторів незалежно від їхньої довжини (магнитуди). Значення косинусної схожості лежить у діапазоні від -1 до 1:

- Значення 1 означає, що вектори мають однакову орієнтацію (кут між ними 0°), що відповідає максимальній схожості.
- Значення 0 означає, що вектори ортогональні (кут 90°), що вказує на відсутність схожості (в контексті орієнтації).
- Значення -1 означає, що вектори мають протилежну орієнтацію (кут 180°), що відповідає максимальній несхожості.

У задачах NLP, де вектори представляють семантику, саме орієнтація вектора часто вважається важливішою за його довжину, оскільки вона відображає "напрямок" значення у семантичному просторі. Тому косинусна схожість є зручною метрикою: чим ближче значення до 1, тим менший кут між векторами і тим більш схожими вважаються відповідні слова або тексти [6].

Математично косинусна схожість між двома векторами A та B розраховується за формулою [13]:

$$\cos(\theta) = \frac{\mathrm{A} \cdot \mathrm{B}}{\| \mathrm{A} \| \| \mathrm{B} \|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \times \sqrt{\sum_{i=1}^{n} B_i^2}}$$

де:

- $\mathrm{A} \cdot \mathrm{B}$ – скалярний добуток векторів A та B.
- $\|A\|$ та $\|B\|$ – евклідові норми (довжини) векторів A та B відповідно.
- A_i та B_i – компоненти векторів A та B.
- n – розмірність векторного простору.

У проєкті «Словозв'яз» саме косинусна схожість використовується для обчислення близькості між вектором цільового (секретного) слова/визначення та векторами всіх інших слів/визначень зі словника, отриманими за допомогою моделі text-embedding-3-large [5]. Результати цих обчислень потім сортуються для створення фінального рейтингу слів, який бачить гравець, де слово з найвищою косинусною схожістю (найближчою до 1) отримує ранг 1.

2.3 Технології генерації тексту: gpt-3.5-turbo для створення визначень

Підхід, описаний у підрозділі 2.1.4, що передбачає використання семантичних визначень для фокусування векторних представлень слів, потребував інструменту для автоматизованої генерації цих визначень для всього корпусу слів (близько 14 000 іменників). Ручне створення такої кількості визначень було б надзвичайно трудомістким процесом. Тому для вирішення цього завдання було використано можливості сучасних великих мовних моделей (Large Language Models, LLMs), здатних генерувати людиноподібний текст на основі вхідного запиту (промпту).

Зокрема, для генерації визначень у проєкті «Словозв'яз» була обрана модель gpt-3.5-turbo від OpenAI [14]. Ця модель належить до сімейства GPT-3.5, яке є вдосконаленням попередніх моделей GPT-3 і здатне розуміти та генерувати природну мову або код. Моделі gpt-3.5-turbo оптимізовані для завдань чату, але також добре справляються із завданнями, що вимагають чіткого слідування інструкціям [14]. Вони пропонують хороший баланс між можливостями, швидкістю та вартістю, що робить їх популярним вибором для багатьох практичних завдань [14].

У рамках даної роботи модель gpt-3.5-turbo використовувалася для генерації короткого, лаконічного визначення для кожного українського іменника з підготовленого списку. Сформульований для моделі промпт містив інструкцію надати визначення в одному реченні, точно описати основне семантичне значення слова, уникати зайвих деталей та не повторювати саме слово на початку відповіді. Результатом роботи моделі став JSON-файл, де кожному слову зі списку (якщо генерація була успішною) відповідало згенероване визначення, яке потім використовувалося на етапі створення векторних представлень за допомогою моделі text-embedding-3-large (п. 2.1.4).

РОЗДІЛ 3 ПРОЄКТУВАННЯ, РОЗРОБКА ТА ОПТИМІЗАЦІЯ СИСТЕМИ "СЛОВОЗВ'ЯЗ"

3.1 Архітектура системи

Розроблена семантична гра «Словозв'яз» є веб-додатком, побудованим за класичною архітектурою клієнт-сервер. Система включає клієнтську частину (frontend), що працює у браузері користувача, та серверну частину (backend), розгорнуту на хостингу. Важливим етапом життєвого циклу додатку є попередня підготовка даних, яка включає генерацію семантичних рейтингів та наповнення бази даних (рис. 3.1).

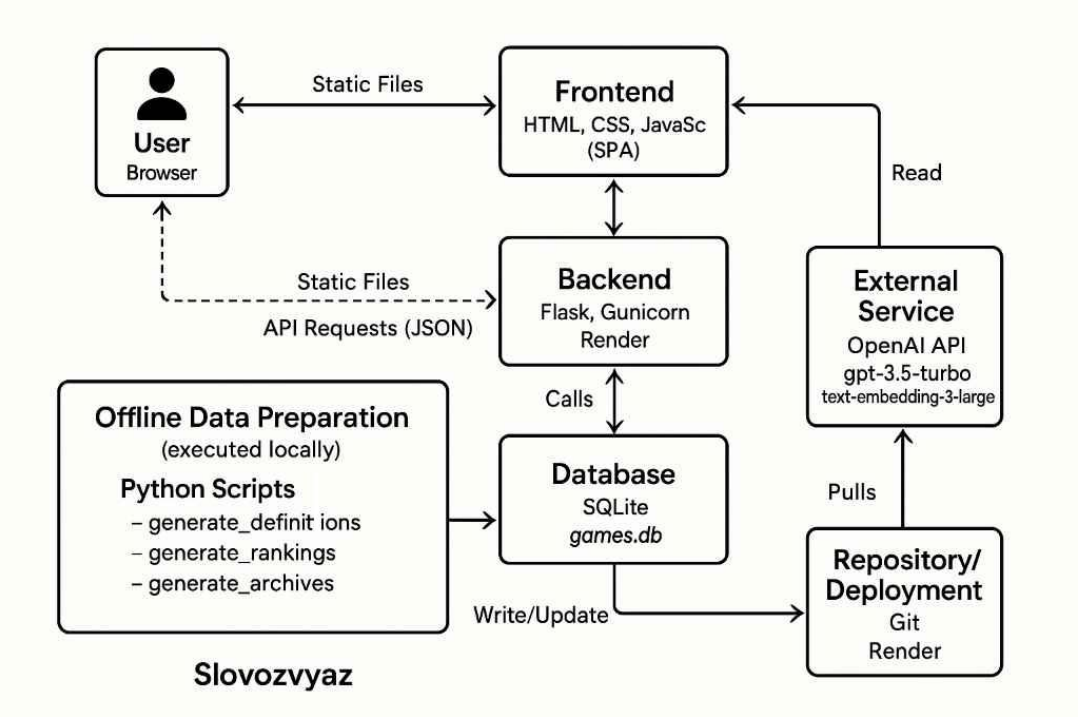

Рисунок 3.1 – Архітектурна діаграма системи «Словозв'яз»

Для візуалізації взаємодії користувача із системою та визначення її основного функціоналу було розроблено діаграму варіантів використання (Use Case Diagram), яка представлена на рисунку 3.2.

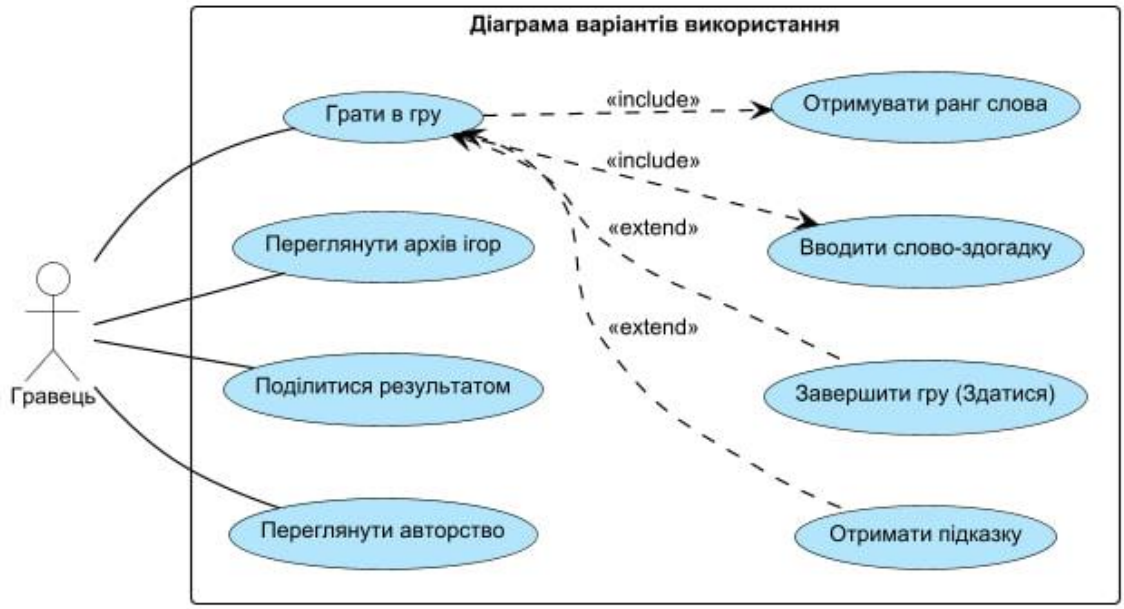

Рисунок 3.2 – Діаграма варіантів використання системи

На діаграмі (рис. 3.2) показано головну дійову особу – Гравця – та його основні можливості. Центральним варіантом використання є «Грати в гру», який обов'язково включає (<<include>>) процеси «Вводити слово-здогадку» та «Отримувати ранг слова». Також гравець може розширити (<<extend>>) ігровий процес, скориставшись підказкою або здавшись. Окремо виділено такі функції, як перегляд архіву ігор, можливість поділитися результатом та перегляд інформації про авторство.

Статичну структуру системи, що включає як онлайн-компоненти, так і елементи офлайн-пайплайну підготовки даних, ілюструє діаграма класів та компонентів (рис. 3.3).

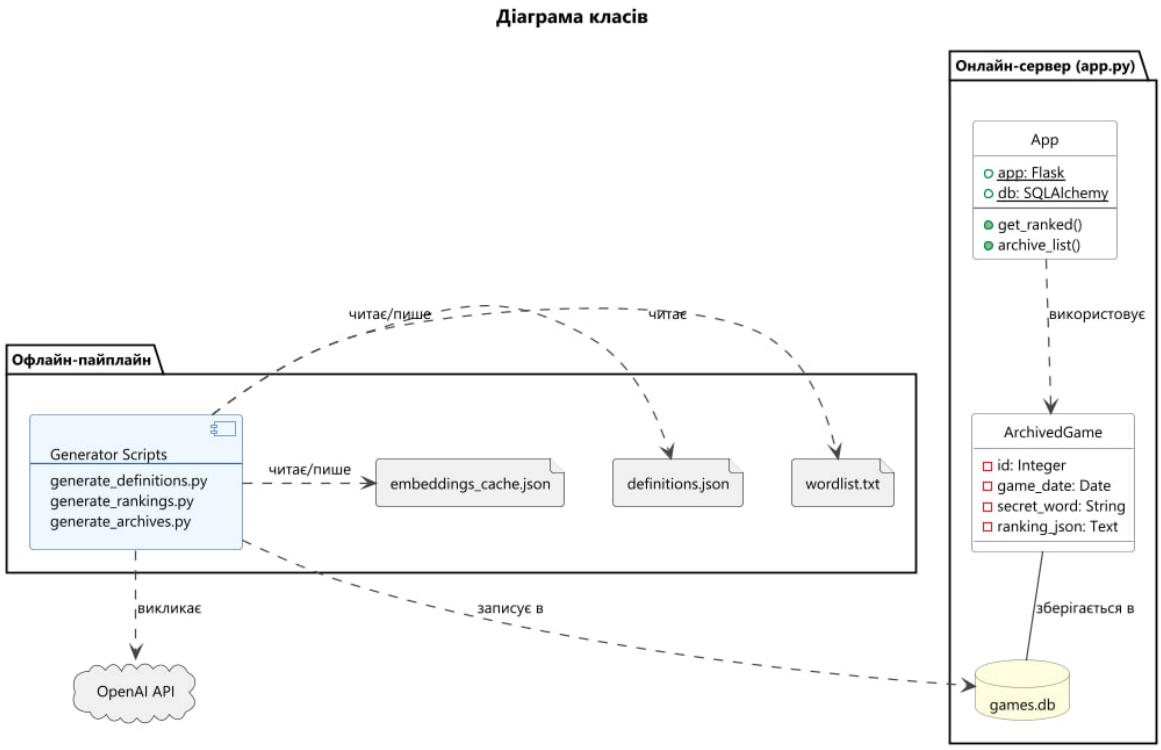

Рисунок 3.3 – Діаграма класів та компонентів системи»

На даній діаграмі (рис. 3.3) деталізовано архітектуру, представлену раніше на рис. 3.1. Вона візуалізує два головні пакети: «Онлайн-сервер» та «Офлайн-пайплайн». Пакет сервера включає клас App, що відповідає за логіку Flask-додатку, та модель ArchivedGame, що є представленням запису в базі даних games.db. Пакет пайплайну показує компонент Generator Scripts, який використовує файли даних (wordlist.txt, definitions.json, embeddings_cache.json), викликає зовнішній OpenAI API та наповнює базу даних. Ця діаграма демонструє ключові залежності та потік даних у всій екосистемі проєкту.

Для деталізації динамічної поведінки системи було розроблено діаграму діяльності, що описує ключовий процес – обробку здогадки користувача на клієнтській частині (рис. 3.4).

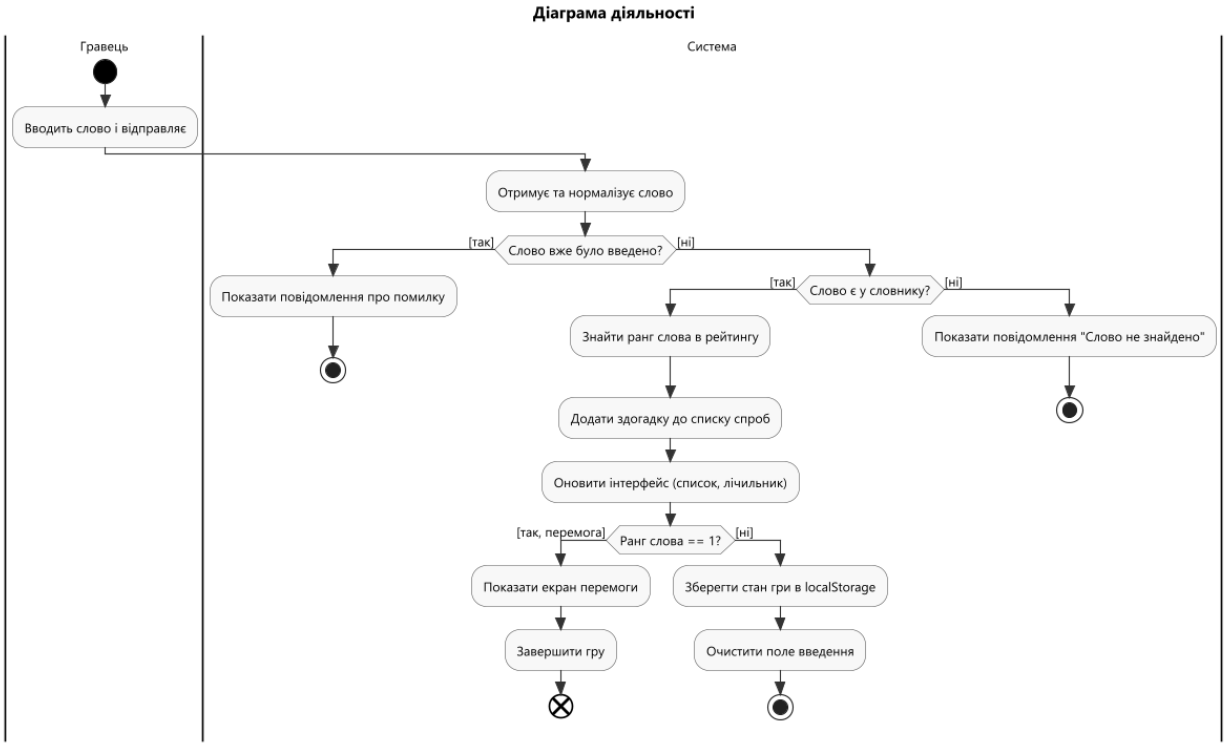

Рисунок 3.4 – Діаграма діяльності

Діаграма (рис. 3.4) покроково ілюструє логіку, реалізовану у функції handleSubmit на фронтенді. Вона розділена на дві доріжки (partitions), що показують зони відповідальності Гравця та Системи. Процес починається з дії гравця, після чого система виконує низку перевірок (чи вводилося слово раніше, чи є воно у словнику), приймає рішення і, залежно від результату, оновлює інтерфейс, завершує гру або очікує на наступну дію.

1. Клієнтська частина (Frontend): Реалізована за допомогою HTML, CSS та JavaScript. Відповідає за:

- Відображення ігрового інтерфейсу та взаємодію з користувачем.
- Надсилання запитів до API серверної частини для отримання ігрових даних.
- Реалізацію ігрової логіки на стороні клієнта.
- Збереження локального стану гри користувача в localStorage.

2. Серверна частина (Backend): Розроблена на Python з використанням Flask та розгорнута на Render (сервер Gunicorn). Основні функції:

- Обслуговування статичних файлів frontend.
- Надання API для клієнта (отримання рейтингу для гри за датою, списку слів, архівних дат тощо), переважно у форматі JSON.
- Читання даних з файлу бази даних SQLite (games.db), який є частиною розгорнутого репозиторію та містить архів ігор з попередньо розрахованими рейтингами (модель ArchivedGame).

3. Підготовка даних (Офлайн етап): Для функціонування гри необхідний набір даних, підготовка якого відбувається локально перед розгортанням або оновленням додатку. Цей процес включає кілька кроків:

- Одноразове формування базових ресурсів:
- Створення основного списку слів (wordlist.txt) ~15 000 найуживаніших українських іменників з частотного словника UberText (lang.org.ua) та їх подальшого очищення (~14 000 слів).
- Генерація семантичних визначень для слів зі списку за допомогою моделі OpenAI API gpt-3.5-turbo [14] та збереження результатів у файл (definitions.json).
- Генерація ігрових даних (виконується перед розгортанням/оновленням):
- Отримання векторних представлень для слів (використовуючи визначення з definitions.json або саме слово за його відсутності) за допомогою моделі text-embedding-3-large [5, 11], з використанням кешування ембеддінгів (embeddings_cache.json).
- Розрахунок косинусної схожості [13] між векторами цільових слів (для щоденних ігор) та всіма іншими словами/визначеннями.
- Формування та збереження попередньо розрахованих рейтингів слів.
- Створення та наповнення/оновлення бази даних SQLite (games.db) архівними записами ігор на основі згенерованих рейтингів за допомогою окремого скрипта (generate_archives.py).

Цей багатоетапний процес підготовки даних, особливо кроки, що включають отримання ембеддінгів та розрахунок рейтингів для всіх слів відносно цільових,

потребує значних обчислювальних ресурсів та доступу до OpenAI API, тому він виконується локально. Готовий файл бази даних games.db, разом зі статичними wordlist.txt та definitions.json (якщо він використовується безпосередньо розгорнутим додатком, а не лише для генерації БД), включається до репозиторію для розгортання.

3.2 Етап підготовки даних

Підготовка даних для гри є багатоетапним процесом, що включає як одноразове формування базових ресурсів (списку слів, визначень), так і ресурсоємні кроки генерації рейтингів та бази даних, які виконуються локально перед розгортанням чи оновленням додатку. Якість та функціональність семантичної гри «Словозв'яз» безпосередньо залежать від цих даних, на основі яких розраховуються рейтинги схожості слів.

3.2.1 Формування корпусу слів

Основою для словника гри слугував великий корпус сучасних українських текстів UberText 2.0, розроблений у рамках проєкту lang-uk [15]. Цей корпус агрегує тексти з різних джерел (новини, художня література, Вікіпедія тощо) і надає дослідникам доступ до значного обсягу мовних даних [15]. Для гри «Словозв'яз» було вирішено сфокусуватися на іменниках як основній частині мови для вгадування. З корпусу UberText було відібрано приблизно 15 000 найуживаніших (найбільш частотних) українських іменників. Цей початковий список пройшов етап очищення, який включав видалення дублікатів, слів з невідповідними символами, потенційних помилок та, можливо, занадто специфічних або рідковживаних термінів. В результаті було сформовано фінальний список з приблизно 14 000 унікальних іменників, який був збережений у файл wordlist.txt і використовувався на подальших етапах підготовки даних. Цей етап є одноразовим – сформований wordlist.txt є статичним ресурсом для проєкту.

3.2.2 Генерація семантичних визначень

Як було обґрунтовано в п. 2.1.4, для фокусування на семантиці та зменшення впливу морфологічної схожості було вирішено використовувати визначення слів при генерації векторних представлень. Цей процес було автоматизовано за допомогою великої мовної моделі OpenAI gpt-3.5-turbo [14]. Був розроблений скрипт generate_definitions.py, який ітерував по кожному слову з wordlist.txt і надсилав запит до OpenAI API з промптом, що інструктував модель надати коротке, лаконічне визначення основного семантичного значення слова українською мовою в одному реченні, уникаючи зайвих прикметників, деталей та повторення самого слова на початку відповіді. Для обробки можливих помилок API була реалізована логіка повторних спроб. Результати генерації – пари "слово": "визначення" – зберігалися у форматі JSON у файл definitions.json. Цей етап є одноразовим, а отриманий файл definitions.json – статичним ресурсом.

3.2.3 Створення векторних представлень

Наступним кроком є отримання векторних представлень (ембеддінгів) для кожного слова зі списку wordlist.txt. Для цього використовується сучасна модель OpenAI text-embedding-3-large [5, 11]. Процес реалізовано у скрипті generate_rankings.py. На вхід моделі text-embedding-3-large подається текст визначення слова з файлу definitions.json (якщо таке визначення існує). Для оптимізації взаємодії з OpenAI API та зменшення витрат часу і коштів було реалізовано кешування ембеддінгів (збереження у embeddings_cache.json) та пакетну обробку запитів (групування текстів у батчі).

3.2.4 Розрахунок рейтингів та формування бази даних

Фінальним кроком етапу підготовки даних є формування бази даних SQLite (games.db), яка буде використовуватися розгорнутим веб-додатком. Для цього використовується окремий скрипт generate_archives.py. Цей скрипт читає попередньо розраховані семантичні рейтинги для кожної ігрової дати (згенеровані підсистемою, описаною в п. 3.3, і збережені, наприклад, у JSON-файлах у папці precomputed/) та створює/оновлює відповідні записи у таблиці ArchivedGame бази даних games.db за допомогою SQLAlchemy. Кожен запис містить дату гри, секретне

слово та повний JSON-рейтинг для цієї гри. Цей крок виконується локально, а готовий файл games.db включається до репозиторію для розгортання.

3.3 Розробка підсистеми генерації рейтингів

Підсистема генерації рейтингів є центральною частиною етапу підготовки даних (п. 3.2) і відповідає за обчислення та впорядкування слів за ступенем їхньої семантичної схожості до заданого цільового (секретного) слова для кожної ігрової дати. Ця підсистема реалізована у вигляді Python-скрипту (generate_rankings.py) і виконується локально перед розгортанням чи оновленням веб-додатку.

Основними завданнями підсистеми є:

1. Отримання векторних представлень:
   - Для заданого цільового слова дня скрипт визначає відповідний текст для генерації ембеддінгу: якщо для цього слова існує визначення у файлі definitions.json, використовується текст визначення, в іншому випадку – саме слово.
   - Аналогічно готується список текстів (визначень або слів) для всіх слів з основного корпусу (wordlist.txt).
   - За допомогою функції get_embeddings_batch, яка взаємодіє з OpenAI API та моделлю text-embedding-3-large [5, 11], отримуються векторні представлення для цільового тексту та всіх текстів корпусу. Ця функція ефективно використовує кешування (через embeddings_cache.json) та пакетну обробку запитів для оптимізації процесу (як описано в п. 3.2.3).
2. Обчислення косинусної схожості:
   - Маючи вектор цільового слова/визначення та вектори всіх слів/визначень корпусу, підсистема ітерує по всіх векторах корпусу.
   - Для кожної пари (вектор цільового слова, вектор слова з корпусу) обчислюється косинусна схожість [13] за допомогою функції

cosine_similarity. Ця функція використовує бібліотеку numpy для ефективних векторних операцій (скалярний добуток, розрахунок норми).

3. Ранжування та збереження результатів:
   - Результати обчислення схожості (пари "слово": схожість) зберігаються у списку.
   - Цей список сортується за значенням схожості у порядку спадання (від максимальної схожості, близької до 1, до мінімальної).
   - Кожному слову у відсортованому списку присвоюється ранг – порядковий номер, починаючи з 1 (де ранг 1 відповідає найвищій схожості – це і є саме цільове слово).
   - Фінальний результат – список об'єктів, кожен з яких містить слово, його косинусну схожість до цільового слова та присвоєний ранг – зберігається у форматі JSON у файл. Назва файлу відповідає даті гри (наприклад, YYYY-MM-DD.json) і розміщується у папці precomputed/.

Ці JSON-файли з попередньо розрахованими рейтингами є основним продуктом даної підсистеми. Вони потім використовуються на наступному етапі підготовки даних (скриптом generate_archives.py, п. 3.2.4) для наповнення бази даних SQLite, яка вже безпосередньо використовується розгорнутим веб-додатком для відображення рейтингів гравцям. Така архітектура дозволяє винести складні та тривалі обчислення за межі роботи веб-сервера, забезпечуючи швидку відповідь API на запити клієнта.

3.4 Реалізація серверної частини (Backend)

Серверна частина (backend) веб-додатку «Словозв'яз» реалізована на мові програмування Python з використанням легковагового веб-фреймворку Flask [16]. Як було зазначено в описі архітектури (п. 3.1), основне завдання розгорнутого на хостингу Render бекенду – це обслуговування клієнтської частини та надання

доступу через API до попередньо підготовлених ігрових даних. Для роботи у production-середовищі використовується WSGI-сервер Gunicorn [18], що зазначено у файлі requirements.txt.

Ключовими компонентами та технологіями бекенду є:

- Веб-фреймворк Flask [16]: Обраний через свою простоту та гнучкість. Файл app.py містить основну логіку додатку, включаючи ініціалізацію Flask, конфігурацію бази даних та визначення маршрутів (ендпоінтів).
- База даних SQLite та SQLAlchemy [17]: Для зберігання даних архівних ігор використовується SQLite (games.db). Взаємодія з базою даних здійснюється через ORM SQLAlchemy за допомогою розширення Flask-SQLAlchemy. У коді визначено модель ArchivedGame, що відповідає таблиці в БД і містить необхідні поля (ідентифікатор, дата, секретне слово, JSON-рейтинг, мітка часу створення).
- API Ендпоінти: Бекенд надає набір HTTP-ендпоінтів для взаємодії з клієнтською частиною:
- /: Обслуговує головну HTML-сторінку (index.html).
- /ranked (або /api/ranked): Приймає дату, здійснює запит до ArchivedGame та повертає відповідний JSON-рейтинг.
- /api/wordlist: Повертає список дозволених слів (з wordlist.txt).
- /api/daily-index: Повертає номер поточної гри.
- /archive: Повертає список дат архівних ігор з БД.
- /archive/<game_date>: Повертає повні дані для конкретної архівної гри з БД.
- /robots.txt, /sitemap.xml: Ендпоінти для пошукових систем.
- Обробка даних: Бекенд читає дані з файлу games.db, що є частиною розгорнутого додатку, та серіалізує їх у JSON. Функція ensure_games_archived у розгорнутому додатку, виконує перевірку наявності таблиць БД. Важливою частиною підготовки цих даних є попереднє генерування визначень для слів та створення щоденних рейтингів близькості, що реалізовано окремими Python-скриптами (generate_definitions.py та generate_rankings.py), код яких наведено у Додатку Б.

Таким чином, реалізація бекенду зосереджена на ефективному наданні доступу до попередньо згенерованих даних, використовуючи стандартні інструменти екосистеми Python/Flask [16, 17].

3.5 Реалізація клієнтської частини (Frontend)

Клієнтська частина (frontend) гри «Словозв'яз» – це односторінковий веб-додаток (SPA), що працює у браузері користувача. Створена на основі HTML, CSS та JavaScript, вона забезпечує відображення інтерфейсу, взаємодію з користувачем та реалізацію ігрової логіки, отримуючи дані від серверної частини (п. 3.4) через API.

Структура інтерфейсу, визначена в index.html, включає ключові елементи керування, інформаційні блоки, меню та модальні вікна. Візуальне оформлення (файл static/css/style.css) задає загальний стиль, темну тему та адаптивність. Динамічне стильове оформлення списку спроб, що залежить від рангу слова, реалізовано через JavaScript-маніпуляції CSS-класами для візуалізації близькості здогадки.

Клієнтська логіка на JavaScript (Vanilla JS) структурована модульно:

- api.js: асинхронна взаємодія з backend API (Fetch API) для завантаження даних (рейтинг, список слів, архів).
- ui.js: генерація та оновлення DOM, рендеринг списку спроб (renderGuesses, createGuessItem) та розрахунок їх стилю (calculateProgressBarStyle).
- main.js: координація клієнтської частини, ініціалізація гри при завантаженні через api.js (отримання списку слів, номера гри, рейтингу). Керує станом гри (спроби, ранг, статус перемоги/здачі), зберігаючи його в localStorage для кожної дати. Ключовим елементом ігрової логіки є функція handleSubmit (код наведено на рисунку 3.5), яка обробляє введення користувача, валідує слово, визначає його ранг, оновлює стан та викликає функції з ui.js для оновлення

UI. Також main.js реалізує додаткові функції: підказки, здача гри, архівні ігри (loadArchive), показ найближчих слів та "Поділитися".

Отже, frontend є інтерактивним компонентом, який реалізує ігрову логіку на стороні клієнта, взаємодіє з API та використовує стандартні веб-технології для інтерфейсу й збереження стану.

```
async function handleSubmit() {
    if (didWin || didGiveUp) return;

    const word = guessInput.value.trim().toLowerCase();
    if (!word) return;

    hideInitialInfoBlocks();

    if (guesses.some(g => g.word === word)) {
        displayDynamicMessage(`Слово "${word}" вже було введено`, true);
        guessInput.value = "";
        guessInput.focus();
        return;
    }

    if (!allowedWords.has(word)) {
        displayDynamicMessage("Вибачте, я не знаю цього слова", true);
        guessInput.focus();
        return;
    }

    const match = rankedWords.find(item => item.word === word);
    let data = match
        ? { rank: match.rank }
        : { rank: Infinity, error: true, errorMessage: "Цього слова немає у рейтингу цього дня." };

    guessCount++;
    if (guessCountElem) guessCountElem.textContent = guessCount;
    lastWord = word;
    guesses.push({ word, rank: data.rank, error: data.error || false, errorMessage: data.errorMessage
});

    guesses.sort((a, b) => {
        if (a.error && !b.error) return -1;
        if (!a.error && b.error) return 1;
        if (a.error && b.error) return 0;
        return a.rank - b.rank;
    });

    if (!data.error) {
        if (data.rank < bestRank) bestRank = data.rank;
        if (data.rank === 1) {
            renderGuesses(guesses, lastWord, MAX_RANK, guessesContainer, lastGuessWrapper,
lastGuessDisplay);
            endGameAsWin();
            return;
        }
    }

    renderGuesses(guesses, lastWord, MAX_RANK, guessesContainer, lastGuessWrapper, lastGuessDisplay);
    guessInput.value = "";
    guessInput.focus();
    saveGameState();
}
```

Рисунок 3.5 – Реалізація функції handleSubmit для обробки припущення користувача»

РОЗДІЛ 4. ПОРІВНЯЛЬНИЙ АНАЛІЗ АЛГОРИТМІВ ТА РЕЗУЛЬТАТІВ РОБОТИ ГРИ "СЛОВОЗВ'ЯЗ"

4.1 Функціонал та інтерфейс розробленої гри "Словозв'яз"

Розроблений веб-додаток «Словозв'яз» є україномовною семантичною грою-головоломкою, доступною через веб-браузер. Мета гри – вгадати секретне слово дня, спираючись на семантичну близькість слів-здогадок до нього. Нижче описано основний функціонал та ключові елементи інтерфейсу реалізованої системи.

Основний ігровий процес:

- Мета: Відгадати щоденне секретне слово (іменник української мови).
- Введення здогадок: Гравець вводить слова-здогадки у спеціальне текстове поле. Система перевіряє, чи є слово валідним (присутнім у завантаженому словнику wordlist.txt) та чи не було воно введене раніше.
- Візуалізація близькості: Кожен рядок у списку спроб містить візуальний індикатор (прогрес-бар), ширина та колір якого динамічно змінюються залежно від рангу слова, надаючи гравцеві інтуїтивне уявлення про близькість здогадки. Список спроб сортується за рангом (рис. 4.4).

Рисунок 4.4 – Приклад ігрового процесу «Словозв'яз»: список спроб з рангами та візуалізацією близькості

- Зворотний зв'язок: Після кожної валідної здогадки система відображає введене слово та його ранг – позицію у списку слів, відсортованому за косинусною схожістю до секретного слова (де ранг 1 – це саме секретне слово). Чим нижчий ранг, тим семантично ближче слово до цілі.
- Необмежені спроби: Гравець може робити необмежену кількість спроб для знаходження секретного слова.
- Стан гри: Відстежується кількість зроблених спроб та найкращий досягнутий ранг. Прогрес гравця для кожної ігрової дати зберігається у localStorage браузера, дозволяючи продовжити гру пізніше.

Інтерфейс користувача: Інтерфейс гри (реалізований за допомогою HTML, CSS та JavaScript, як описано в п. 3.5) включає наступні елементи:

- Головний екран: Містить поле для введення слова, кнопку відправки, лічильник спроб, номер/дату поточної гри, динамічний список зроблених спроб з рангами та візуальними індикаторами, а також окремий блок для відображення останньої вдалої спроби (рис. 4.1).

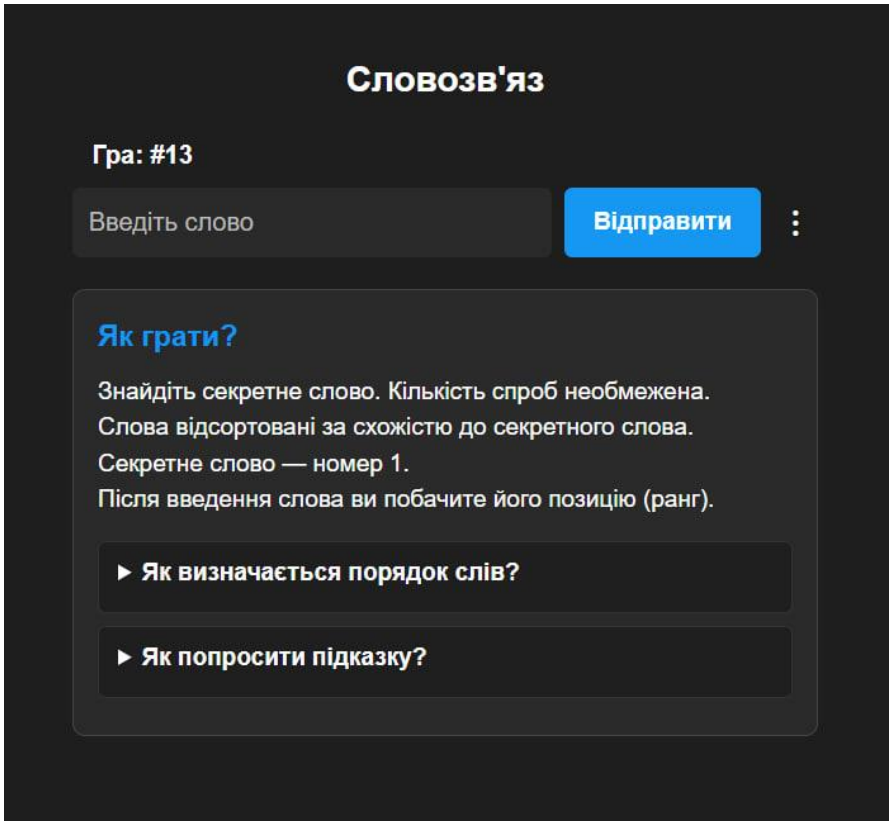

Рисунок 4.1 – Головний екран гри Словозв’яз

- Меню: Випадаюче меню надає доступ до додаткових функцій: "Підказка", "Здатися", "Попередні ігри", "Авторство".
- Інформаційні блоки: Блок "Як грати?" пояснює правила; блок результатів гри з'являється після перемоги або здачі.
- Модальні вікна: Використовуються для:
    - Вибору попередньої гри з архіву (зі списком дат та кнопкою "Випадкова").
    - Відображення списку N найближчих слів до секретного після завершення гри (рис. 4.3).
    - Показу інформації про авторство та використані технології.

Додаткові функції:

- Підказка: Надає гравцеві одне з ще не введених слів з рейтингу (логіка вибору слова базується на поточному найкращому рангу). Використання підказки зараховується як спроба.
- Здатися: Дозволяє гравцеві завершити поточну гру достроково, після чого система показує секретне слово.
- Архів ігор: Гравець може переглядати список попередніх ігрових днів та грати в будь-яку з архівних ігор (рис. 4.2).

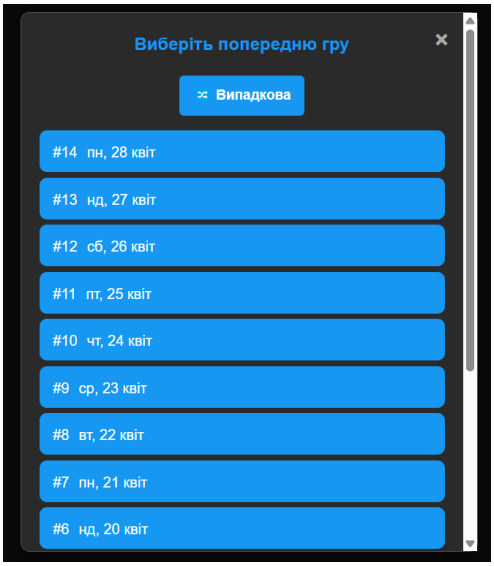

Рисунок 4.2 – Вікно вибору попередньої гри з архіву

- Найближчі слова: Після завершення гри (перемоги або здачі) гравець може переглянути список слів, які були семантично найближчими до секретного слова (рис. 4.3).

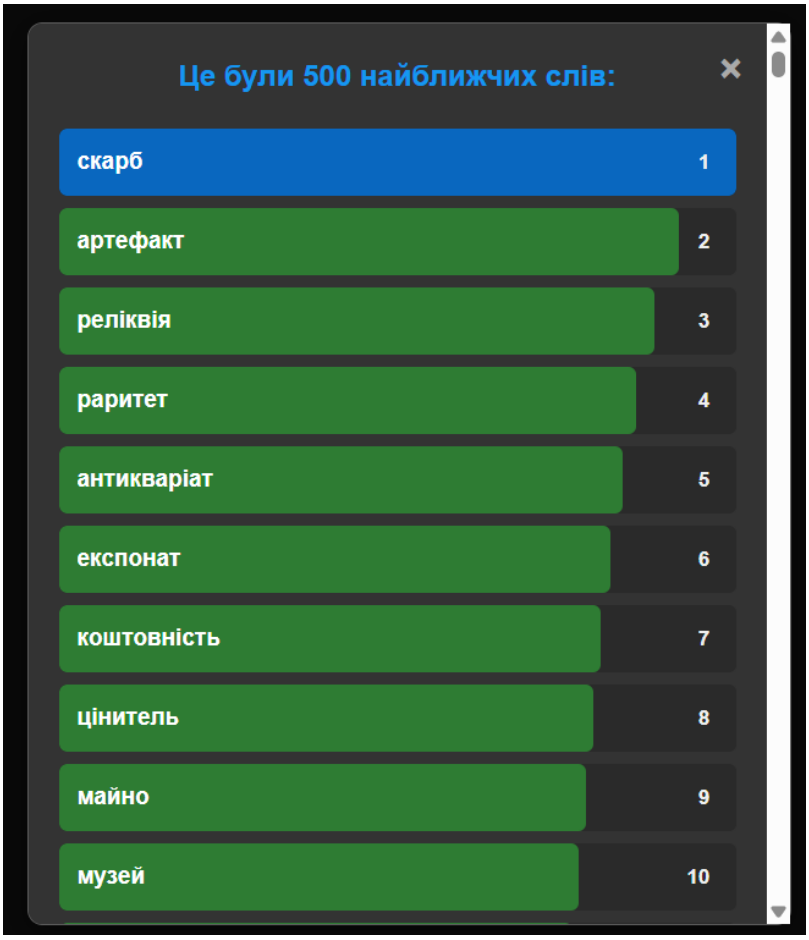

Рисунок 4.3 – Вікно найближчих слів

- Поділитися результатом: Надає можливість скопіювати або поділитися (через Web Share API) коротким резюме результату гри (номер гри, кількість спроб, статус).

Інтерфейс має темну тему оформлення та реалізований з урахуванням адаптивності для коректного відображення на різних пристроях. Загалом, функціонал та інтерфейс гри «Словозв'яз» реалізують основну ідею семантичного пошуку слова, надаючи гравцеві необхідні інструменти та зворотний зв'язок для ігрового процесу.

4.2 Аналіз якості семантичних рейтингів у "Словозв'яз"

Якість семантичних рейтингів є ключовим фактором, що визначає ігровий досвід та відповідність гри заявленій меті – вгадуванню слів за їхнім значенням. У цьому підрозділі проводиться аналіз рейтингів, згенерованих за допомогою розробленого підходу (використання визначень та моделі ембеддінгів text-embedding-3-large, як описано в п. 2.1.4 та п. 3.3), на прикладі кількох слів різного типу: конкретних іменників ("стіл", "дерево"), абстрактних понять ("свобода", "щастя") та полісемічних слів ("ключ", "замок", "коса").

4.2.1 Приклади рейтингів та їх аналіз

- Приклад 1: Секретне слово "стіл" (конкретне поняття)

Перші позиції рейтингу для слова "стіл" займають слова, що безпосередньо пов'язані з ним за функцією або типом: "столик" (ранг 2), "парта" (ранг 6). Високі позиції також займають інші предмети меблів ("поличка" - ранг 3, "буфет" - 9, "тапчан" - 10, "шафка" - 11, "лавка" - 12, "тумба" - 16, "комод" - 20, "табуретка" - 27, "ліжко" - 28, "крісло" - 33, "шафа" - 34, "диван" - 40) або узагальнення "меблі" (ранг 23). Це свідчить про добре вловлювання системою категорії "предмети інтер'єру/меблі". Також близько розташовані предмети, що часто використовуються разом зі столом або мають схожу функціональну поверхню: "таця" (ранг 4), "посуд" (ранг 8), "мисочка" (ранг 13), "таріль" (ранг 14), "тарілка" (ранг 21), "блюдо" (ранг 41), "підставка" (ранг 36). Цікавими є зв'язки з локаціями, де столи є центральним елементом: "трапезна" (ранг 25), "ресторан" (ранг 29), "кафе" (ранг 42), "кабінет" (ранг 49). Загалом, топ-50 для слова "стіл" виглядає семантично обґрунтованим, групуючи пов'язані об'єкти, поняття та локації. Однак, присутнє слово "ковила" (ранг 37), яке здається семантично дуже далеким і може бути артефактом або помилкою на етапі генерації визначення чи ембеддінгу.

- Приклад 2: Секретне слово "дерево" (конкретне поняття, природа)

Рейтинг для слова "дерево" демонструє чітке групування навколо ботанічної тематики. Найближчими є прямі синоніми та зменшувальні форми: "древо" (ранг 2), "деревце" (ранг 3). Далі йдуть інші типи рослин: "кущ" (ранг 7), "пальма" (ранг 8), а також конкретні види дерев та рослин: "бамбук" (4), "кедр" (6), "тополя" (9), "платан" (10), "липа" (19), "бук" (20), "береза" (23), "клен" (28), "граб" (32), "кипарис" (35), "акація" (42), "магнолія" (49) та інші трави, квіти, частини рослин ("лопух", "нарцис", "осока", "тростина", "очерет", "трава", "мох", "кактус" тощо). Це показує здатність системи добре охоплювати семантичне поле "рослини". Викликають питання слова "панцерник" (ранг 22) та "канчук" (ранг 37), які виглядають як явні аномалії у цьому контексті.

- Приклад 3: Секретне слово "свобода" (абстрактне поняття)

Рейтинг для "свобода" є дуже показовим. Найближчим є прямий синонім "воля" (ранг 2), а також близькі за значенням "вольність" (3), "незалежність" (4), "вільність" (5), "автономія" (7), "самостійність" (8), "автономність" (10). Цікаво, що антонім "несвобода" знаходиться досить високо (ранг 6), а інший антонім "неволя" – далі (ранг 46). Це може свідчити про те, що навіть із визначеннями, модель ембеддінгів вловлює близькість контекстів вживання протилежних понять (як це характерно для Word2Vec). Рейтинг також включає пов'язані абстрактні поняття: "рівноправність" (12), "приватність" (13), "вседозволеність" (14, як крайній прояв свободи), "рівність" (23), "недоторканість" (24), "безпека" (32), "щастя" (33), "процвітання" (27), "благополуччя" (34), "змога" (48), "прозорість" (47). Це демонструє здатність системи працювати з абстрактними категоріями та їхніми зв'язками.

- Приклад 4: Секретне слово "щастя" (абстрактне поняття, емоція)

Аналогічно до "свободи", рейтинг для "щастя" добре групує синонімічні та близькі поняття: "блаженство" (2), "насолода" (6), "кайф" (5), "веселість" (7), "ейфорія" (19), "втіха" (30), "позитив" (31), "задоволення" (45). Також присутні стани та умови, що асоціюються зі щастям: "злагода" (3), "процвітання" (4), "комфорт" (8), "затишок" (9), "спокій" (11), "благополуччя" (12), "достаток" (13), "ідилія" (14), "рай" (15), "розкіш" (17), "мир" (23), "безпека" (28), "добробут" (29),

"рівновага" (34). Слово "свобода" тут також присутнє (ранг 35). Аномаліями виглядають слова "нужда" (ранг 43) та "хворий" (ранг 49).

- Приклад 5: Аналіз полісемії ("ключ", "замок", "коса")

Ці приклади чітко ілюструють ефект використання одного визначення для багатозначних слів.

- Для "ключ": Рейтинг (топ-10: ключка, ключик, засув, запор, клямка, викрутка, засувка, звірка, кувалда, кайло) явно сфокусований на значенні "інструмент для замикання/відмикання" або "інструмент/знаряддя" загалом. Інші значення (музичний, джерело, до шифру) у топ-50 відсутні.
- Для "замок": Рейтинг (топ-10: оплот, твердиня, форт, острог, фортеця, острога, цитадель, загорожа, кремль) однозначно вказує на значення "укріплена споруда, фортеця". Значення "механізм для замикання" повністю ігнорується у топ-50.
- Для "коса": Рейтинг (топ-10: чуприна, джгут, грива, завивання, зачіска, китиця, клубок, плетіння, чуб) концентрується на значенні "заплетене волосся". Значення "сільськогосподарське знаряддя" або "географічний об'єкт" відсутні.

Загальний аналіз якості: Наведені приклади свідчать, що розроблений підхід із використанням визначень та моделі text-embedding-3-large дозволяє генерувати семантично зв'язні та здебільшого інтуїтивно зрозумілі рейтинги слів як для конкретних, так і для абстрактних понять. Ключовою особливістю є ефект фокусування: використання одного визначення ефективно спрямовує модель на одне конкретне значення слова, що особливо помітно на прикладах полісемічних слів ("ключ", "замок", "коса"). Це може бути перевагою для гри, роблячи її більш передбачуваною. Водночас, це є і обмеженням, оскільки інші значення слова ігноруються. Також варто відзначити присутність антонімів на високих позиціях ("несвобода" біля "свободи"), що може бути спадком від навчання базової моделі ембеддінгів. Наявність поодиноких семантичних аномалій (наприклад, "панцерник" біля "дерево", "нужда" біля "щастя") підкреслює важливість якості

даних на всіх етапах. Зокрема, аномально високий ранг слова "ковила" (ранг 37) у рейтингу для "стіл" пояснюється помилкою на етапі генерації визначень: модель gpt-3.5-turbo надала для "ковила" некоректне визначення ("місце для відпочинку або сидіння..."), що призвело до його помилкового семантичного зближення зі словами категорії "меблі". Це вказує на чутливість даного підходу до якості автоматично генерованих визначень. Незважаючи на ці окремі випадки, загалом якість рейтингів є достатньо високою для реалізації ігрової механіки, а виявлені особливості є важливими для порівняльного аналізу.

Це призводить до групування слів, що часто зустрічаються в подібних контекстах або є функціонально взаємозамінними [4]. Аналіз рейтингу для слова "table" в Semantle підтверджує це: високі позиції займають предмети, що безпосередньо використовуються разом зі столом (tray, napkin, tablecloth, trivet, serviettes, platter), інші меблі, дії (seated, sit) та пов'язані локації (kitchen). Це свідчить про фокус на функціональній та ситуативній близькості. Цікавою особливістю, що випливає з опори на локальний контекст, є потенційна близькість антонімів, які часто вживаються в однакових граматичних конструкціях [4], що також згадується у підказках до Semantle [2]. Такий тип схожості апелює до інтуїції гравця щодо того, які слова використовуються *разом* або *замість* цільового слова у типових фразах та ситуаціях.

4.3 Порівняльний аналіз алгоритмів семантичної схожості та їх впливу

Одним із ключових завдань даної роботи є не лише розробка гри, але й аналіз ефективності та характеру семантичної близькості, що генерується обраним підходом (OpenAI text-embedding-3-large + визначення), у порівнянні з алгоритмами, що використовуються в популярних іграх-аналогах – Word2Vec (гра Semantle) та GloVe (гра Contexto). Розуміння відмінностей у типах

схожості, які вловлюють ці алгоритми, дозволяє оцінити потенційний вплив на ігровий досвід користувача.

4.3.1 Word2Vec (Semantle): Схожість за локальним контекстом

Алгоритм Word2Vec [1], що використовується у грі Semantle [2], визначає схожість слів на основі їхнього локального оточення (п. 2.1.2). Це призводить до групування слів, що часто зустрічаються в подібних контекстах або є функціонально взаємозамінними [4]. Аналіз рейтингу для слова "table" в Semantle підтверджує це: високі позиції займають предмети, що безпосередньо використовуються разом зі столом (tray, napkin, tablecloth, trivet, serviettes, platter), інші меблі, дії (seated, sit) та пов'язані локації (kitchen). Це свідчить про фокус на функціональній та ситуативній близькості. Цікавою особливістю, що випливає з опори на локальний контекст, є потенційна близькість антонімів, які часто вживаються в однакових граматичних конструкціях [4], що також згадується у підказках до Semantle [2]. Такий тип схожості апелює до інтуїції гравця щодо того, які слова використовуються *разом* або *замість* цільового слова у типових фразах та ситуаціях.

4.3.2 GloVe (Contexto): Схожість за глобальним контекстом

На відміну від Word2Vec, модель GloVe [2], що лежить в основі Contexto [1], використовує глобальну статистику спільного вживання слів у всьому корпусі (п. 2.1.2). Це дозволяє вловлювати ширші тематичні та асоціативні зв'язки, які не завжди очевидні з локального контексту [4]. Рейтинг для слова "table" в Contexto це ілюструє: поруч зі столом опиняються не лише меблі ("chair", "desk"), але й дії ("dine", "set", "sit"), локації ("room", "bar", "kitchen"), матеріал ("wood") та навіть слова, пов'язані з таблицями як структурами даних ("example", "figure", "data", "row", "type"). Багато слів у рейтингу Contexto ("different", "include", "provide", "instead") відображають те, як слово "table" використовується в текстах загалом, а не лише його предметне значення. Це створює більш "асоціативний" тип схожості, який вимагає від гравця розуміння ширших контекстів та тем, у яких фігурує цільове слово.

4.3.3 OpenAI + визначення: Схожість за сфокусованим значенням

Підхід, реалізований у грі «Словозв'яз», використовує сучасну модель ембеддінгів text-embedding-3-large [5, 11] у поєднанні з попередньо згенерованими визначеннями слів [14] (п. 2.1.4). Аналіз рейтингів, наведений у п. 4.2, показує, що цей підхід генерує сфокусовану семантичну близькість.

- Фокус на одному значенні: Приклади полісемічних слів ("ключ", "замок", "коса") демонструють, що використання одного визначення спрямовує модель на конкретний сенс слова, ігноруючи інші. Рейтинги для цих слів чітко групуються навколо значення, ймовірно, заданого визначенням (інструмент, фортеця, зачіска відповідно), що відрізняється від потенційного усереднення значень у статичних моделях.
- Семантична когерентність: Для слів з відносно однозначною семантикою ("стіл", "дерево", "свобода", "щастя") рейтинги виглядають логічними, об'єднуючи синоніми, гіпоніми/гіпероніми та інші семантично пов'язані поняття.
- Зменшення морфологічного впливу: Хоча прямий експеримент не проводився, очікується, що використання визначень зменшує ймовірність високої схожості між словами лише через подібність форми (як потенційна проблема "рік"/"рик").
- Залишкові ефекти та чутливість: Спостерігається близькість антонімів ("свобода"/"несвобода"), що може бути властивістю базової моделі ембеддінгів. Також підхід чутливий до якості згенерованих визначень, що може призводити до аномалій ("ковила"), якщо визначення було некоректним.

Таким чином, використаний підхід намагається наблизитись до "чистої" семантичної схожості, фокусуючись на значенні, заданому визначенням, завдяки можливостям сучасної моделі [11, 12] та додатковому семантичному сигналу від визначення.

4.3.4 Зіставлення підходів та вплив на гравця

Порівняння трьох підходів виявляє різні типи семантичної схожості, що впливає на ігровий досвід:

- Word2Vec (Semantle): Базується на локальному контексті, добре вловлює функціональні зв'язки та взаємозамінність слів. Може зближувати антоніми. Гравець інтуїтивно шукає слова, що використовуються *поруч* або *замість*.
- GloVe (Contexto): Базується на глобальній статистиці, виявляє ширші тематичні та асоціативні зв'язки. Рейтинги можуть бути менш передбачуваними, включати слова з різних категорій, що вживаються в схожих дискурсах. Гравець шукає слова, пов'язані з *темою* або *контекстами використання*.
- OpenAI+визначення (Словозв'яз): Намагається сфокусуватися на конкретному значенні слова завдяки визначенню та потужності сучасної моделі. Потенційно дає більш "чисту" семантичну близькість за значенням, але ігнорує інші сенси полісемічних слів та залежить від якості визначень. Гравець шукає слова, близькі за *основним значенням* до цільового.

Для гри, що позиціонується як семантична головоломка, підхід, реалізований у "Словозв'яз", видається перспективним. Фокусування на значенні, заданому визначенням, може зробити процес вгадування більш логічним та таким, що відповідає очікуванням гравця від поняття "семантична близькість", порівняно з широкими асоціаціями GloVe або функціональною близькістю Word2Vec (яка іноді зближує протилежності). Незважаючи на обмеження (ігнорування полісемії, чутливість до помилок визначень), цей підхід демонструє потенціал використання сучасних великих моделей ембеддінгів у поєднанні з додатковою семантичною інформацією для створення більш точного ігрового досвіду.

4.4 Аналіз ефективності застосованих оптимізацій

Під час розробки етапу підготовки даних (п. 3.2) та підсистеми генерації рейтингів (п. 3.3) було застосовано низку оптимізаційних технік, спрямованих на зменшення часу виконання, зниження витрат на використання зовнішніх API та забезпечення високої продуктивності розгорнутого додатку. До ключових оптимізацій належать кешування ембеддінгів, пакетна обробка запитів до API та прекомпіляція (офлайн-генерація) рейтингів.

- Кешування ембеддінгів: Як описано в п. 3.2.3, для уникнення повторних запитів до OpenAI API для одних і тих самих слів або визначень, було реалізовано механізм кешування отриманих векторів у локальному файлі (embeddings_cache.json). Ефективність цього підходу підтверджується загальними витратами на OpenAI API за весь період розробки, які склали лише $2.66. Враховуючи необхідність отримання ембеддінгів для ~14 000 слів/визначень та багаторазові запуски скриптів генерації рейтингів під час розробки та тестування (наприклад, для 10 рейтингів, згенерованих для аналізу в п. 4.2, було витрачено $0.08), стає очевидним, що кешування дозволило значно скоротити загальну кількість токенів, що передавалися на обробку API (згідно з даними OpenAI, загальна кількість токенів склала ~4.85 млн), і, відповідно, суттєво зменшити фінансові витрати та час очікування. Без кешування вартість та час підготовки даних були б набагато вищими.
- Пакетна обробка запитів (Batching): При отриманні ембеддінгів за допомогою скрипта generate_rankings.py запити до OpenAI API надсилалися не індивідуально для кожного тексту, а пакетами (батчами) по 100 елементів (п. 3.2.3). Хоча пряме порівняння швидкості з індивідуальними запитами не проводилося, пакетна обробка є стандартною рекомендованою практикою для підвищення пропускної здатності при роботі з API [5]. Це дозволяє зменшити мережеві затримки та оптимізувати обробку на стороні

сервісу, що сприяло швидшій генерації векторів для всього корпусу слів за прийнятний час. Загальна кількість запитів (~30 тис.) також ефективніше обробляється завдяки пакетуванню.

- Прекомпіляція / Офлайн-генерація рейтингів: Найважливішою оптимізацією з точки зору кінцевого користувача та продуктивності розгорнутого додатку є повна попередня генерація всіх ігрових рейтингів та їх збереження у базі даних SQLite (games.db), яка розгортається разом із додатком (п. 3.1, 3.2.4). Альтернативний підхід – розрахунок рейтингу "на льоту" для кожного запиту до API – був би абсолютно неприйнятним. Він вимагав би отримання ~14 000 ембеддінгів та обчислення ~14 000 значень косинусної схожості при кожному завантаженні гри, що займало б хвилини часу та призводило б до значних витрат на API при кожному запиті. Натомість, завдяки прекомпіляції, API бекенду лише читає готовий JSON-рейтинг з локальної бази даних, що забезпечує відповідь за лічені мілісекунди та гарантує швидкий і плавний ігровий досвід. Навіть відносно невелика вартість офлайн-генерації 10 рейтингів ($0.08) показує непрактичність розрахунків у реальному часі для користувача.

Висновок: Застосовані оптимізаційні техніки – кешування ембеддінгів, пакетна обробка запитів та прекомпіляція рейтингів – були невід'ємною частиною процесу розробки. Вони дозволили ефективно керувати витратами на зовнішні API, скоротити час підготовки даних та, найголовніше, забезпечити високу швидкість роботи та позитивний користувацький досвід для фінального веб-додатку.

4.5 Практичне значення, висновки щодо ефективності та якості обраного підходу у порівнянні з аналогами та напрямки розвитку

У рамках даної дипломної роботи було розроблено та проаналізовано україномовну семантичну гру «Словозв'яз», що базується на сучасних методах

обробки природної мови. Проведений аналіз дозволяє зробити низку висновків щодо якості обраного підходу, його ефективності та практичного значення, а також окреслити напрямки подальшого розвитку.

Висновки щодо якості та ефективності:

1. Якість семантичних рейтингів: Обраний підхід, що поєднує використання семантичних визначень (згенерованих gpt-3.5-turbo [14]) та потужної моделі ембеддінгів (text-embedding-3-large [5, 11]), дозволив створити рейтинги слів, які здебільшого демонструють високу семантичну когерентність та інтуїтивну зрозумілість для носія мови (п. 4.2). Це підтверджується аналізом прикладів для слів різного типу.
2. Обробка полісемії: Використання одного визначення для слова ефективно фокусує векторне представлення на конкретному значенні, що дозволяє уникнути усереднення та отримати чіткі рейтинги для багатозначних слів (п. 4.2.1, 4.3.3). Водночас, це є і обмеженням, оскільки інші значення слова ігноруються.
3. Порівняння з аналогами: У порівнянні з Word2Vec (Semantle) [1], що базується на локальному контексті та функціональній схожості, та GloVe (Contexto) [2], що спирається на глобальну статистику та асоціативні зв'язки, підхід «Словозв'яз» пропонує тип схожості, більш сфокусований на лексичному значенні, що може бути більш відповідним для гри, орієнтованої саме на семантику (п. 4.3.4). Однак, спостерігається близькість деяких антонімів ("свобода"/"несвобода"), що може бути успадковано від базової моделі ембеддінгів.
4. Вплив якості визначень: Аналіз показав чутливість підходу до якості автоматично згенерованих визначень. Помилки на етапі генерації визначень (як у випадку зі словом "ковила") безпосередньо призводять до семантичних аномалій у фінальних рейтингах (п. 4.2.1).
5. Ефективність оптимізацій: Застосовані технічні оптимізації – кешування ембеддінгів, пакетна обробка запитів до API та прекомпіляція рейтингів – виявилися критично важливими для реалізації проєкту (п. 4.4). Вони

дозволили суттєво скоротити витрати на зовнішні API, зменшити час підготовки даних та забезпечити високу швидкість роботи розгорнутого веб-додатку.

Практичне значення:

- Розроблено та розгорнуто функціональний веб-додаток «Словозв'яз» – одну з перших україномовних семантичних ігор такого типу, що робить внесок у розвиток україномовного цифрового контенту [3].
- Продемонстровано практичне застосування сучасної моделі ембеддінгів text-embedding-3-large [5, 11] для української мови та проаналізовано її характеристики у специфічній задачі ранжування за семантичною схожістю.
- Запропоновано та досліджено підхід, що поєднує ембеддінги з LLM-згенерованими визначеннями, виявлено його сильні сторони (фокусування значення) та слабкі місця (чутливість до якості визначень, ігнорування полісемії).
- Створено програмний каркас (backend на Flask [16], frontend на JS), який може бути основою для подальшого розвитку подібних NLP-орієнтованих проєктів.

Напрямки розвитку:

- Покращення обробки полісемії: Дослідження методів генерації та використання кількох визначень для одного слова; інтеграція моделей, що безпосередньо працюють зі значеннями слів (word sense disambiguation/embeddings).
- Підвищення якості визначень: Використання досконаліших LLM (наприклад, GPT-4) для генерації, застосування технік промпт-інжинірингу, впровадження етапу валідації або пост-редагування визначень.
- Розширення словника: Включення інших частин мови (прикметників, дієслів), збільшення розміру словника іменників.
- Дослідження альтернативних моделей: Порівняння результатів з відкритими (open-source) моделями ембеддінгів, адаптованими для української мови, або іншими комерційними API.

- Покращення ігрового досвіду: Додавання нових режимів гри, системи підказок, що базуються на різних типах семантичних зв'язків, збір статистики та аналіз поведінки гравців для подальшої оптимізації рейтингів.
- Більш глибока оцінка: Проведення користувацьких досліджень для оцінки інтуїтивності та цікавості рейтингів, порівняно з аналогами; використання формальних метрик для оцінки якості семантичної схожості.

Таким чином, дана робота демонструє успішну реалізацію семантичної гри з використанням сучасних NLP-інструментів та закладає основу для подальших досліджень та вдосконалень у цій галузі.

РОЗДІЛ 5. БЕЗПЕКА ЖИТТЄДІЯЛЬНОСТІ, ОСНОВИ ОХОРОНИ ПРАЦІ

5.1 Працездатність людини – оператора

Працездатність людини визначається як її потенційна можливість ефективно виконувати певну роботу протягом заданого часу і залежить від комплексу зовнішніх умов та внутрішніх психофізіологічних властивостей. Її вивчення та оптимізація є завданням фізіології, гігієни та психології праці [19]. Робота оператора, зокрема в ІТ-сфері, є переважно розумовою, що характеризується навантаженням на нервову систему, можливим емоційним напруженням або монотонією [19, с. 94].

Фактори, що впливають на працездатність оператора

На працездатність оператора впливають індивідуально-психофізіологічні фактори та умови виробничого середовища. До перших належать стан здоров'я, функціональний стан організму, професійна підготовка, індивідуально-типологічні властивості нервової системи та мотивація [19]. Професійна придатність, що відображає відповідність якостей працівника вимогам професії, проявляється у швидкості, якості виконання завдань та стійкості до несприятливих умов [19, с. 93].

Ключовими факторами виробничого середовища, що впливають на працездатність оператора, є мікроклімат та освітлення.

- Метеорологічні умови (мікроклімат) виробничих приміщень визначаються температурою, вологістю, швидкістю руху повітря та іншими фізичними параметрами [20, п.6]. Ці фактори суттєво впливають на тепловий стан організму, самопочуття та працездатність людини [19, с. 23]. Несприятливі метеорологічні умови можуть порушувати терморегуляцію, викликати дискомфорт, знижувати увагу, прискорювати втому та сприяти розвитку захворювань [19, с. 87-88]. Згідно з ДСН 3.3.6.042-99, для операторської роботи (категорія Іа [20, п.13]), оптимальними є температура повітря 22-25 °С, відносна вологість 40-60 % та швидкість руху повітря ≤ 0,1 м/с [2, Таблиця 1].

- Освітлення робочих місць має забезпечувати комфортні умови для зорової роботи, запобігати втомі та підтримувати працездатність [21, розділ 5]. Нераціональне освітлення (недостатня освітленість, блискість, пульсація) негативно впливає на зоровий аналізатор і загальний стан оператора [21, п. 5.1]. ДБН В.2.5-28:2018 регламентує нормовані показники освітленості (наприклад, 300-500 лк для офісів з ВДТ), рівномірність розподілу яскравості, обмеження блискості та коефіцієнта пульсації (не більше 10% для приміщень з ПК) [3, Додаток Д, Таблиця Д.1]. Перевага надається природному освітленню, яке нормується коефіцієнтом природної освітленості (КПО) [21, розділ 6].

Динаміка працездатності та її порушення

Працездатність людини протягом робочого дня не є постійною і змінюється за фазами: впрацьовування, оптимальна працездатність, компенсація, зниження та можливий "кінцевий порив" [19, с. 94-95].

Тривала робота викликає стомлення – фізіологічний стан тимчасового зниження працездатності, що є нормальною реакцією організму. Воно проявляється суб'єктивним відчуттям втоми та об'єктивним погіршенням показників роботи [19, с. 93]. Якщо відпочинок не відновлює працездатність, може розвинутися перевтома – більш глибокий та стійкий стан, що негативно впливає на здоров'я [19, с. 88, 95]. Для операторів характерними є також стани нервово-психічної напруги (при високій відповідальності або дефіциті часу) та монотонії (при одноманітній роботі), які прискорюють розвиток втоми [19, с. 94]. Функціональний стан оператора, включаючи оптимальний рівень активації, стомлення, стрес (дистрес), безпосередньо впливає на надійність та ефективність його діяльності [19, с. 194].

Заходи щодо підтримання та підвищення працездатності оператора

Для забезпечення високої та стійкої працездатності оператора необхідний комплекс заходів:

1. Оптимізація умов праці: Дотримання нормативних вимог до мікроклімату [20] та освітлення [21].

2. Раціоналізація режимів праці та відпочинку: Впровадження регламентованих перерв, бажано активного характеру, для відновлення сил та профілактики втоми [19, с. 87].
3. Ергономічна організація робочого місця: Зручне розміщення обладнання та ВДТ з урахуванням антропометричних та психофізіологічних особливостей людини [19].
4. Професійний відбір та навчання: Врахування індивідуальних особливостей та належна підготовка операторів [19].
5. Психопрофілактичні заходи: Зниження нервово-психічної напруги, запобігання монотонії, створення сприятливого психологічного клімату [19].

Забезпечення належних умов праці та врахування психофізіологічних особливостей операторської діяльності є ключовими для підтримання здоров'я та високого рівня працездатності.

5.2 Гігієнічні вимоги до організації та обладнання робочих місць з ВДТ

Забезпечення належних гігієнічних умов та безпеки праці при роботі з відеодисплейними терміналами (ВДТ), або екранними пристроями, є фундаментальною вимогою для збереження здоров’я та підтримки оптимального рівня працездатності працівників. Ці вимоги спрямовані на мінімізацію ризиків, пов’язаних із погіршенням зору, розвитком захворювань опорно-рухового апарату, а також стресом та іншими психофізіологічними розладами [22]. Роботодавець несе відповідальність за створення безпечних умов праці, інформування працівників про потенційні ризики та забезпечення відповідності робочих місць чинним нормативним документам [22].

Загальні вимоги до робочих місць з ВДТ

Робочі місця працівників, які використовують екранні пристрої, мають бути спроектовані з урахуванням ергономічних принципів, забезпечуючи достатній

простір для зміни робочої пози та виконання рухів [22, розділ III, п.1]. Організація робочого місця повинна відповідати антропологічним, психофізіологічним характеристикам працівника та характеру виконуваних ним робіт [22, розділ III, п.3].

Вимоги до обладнання робочого місця

- Екранний пристрій (дисплей): Символи на екрані повинні бути чіткими, контрастними, відповідного розміру, з достатнім інтервалом між символами та рядками [22, розділ V, п.3]. Зображення має бути стабільним, без миготінь чи інших форм нестабільності [22, розділ V, п.4]. Яскравість та контрастність зображення повинні легко регулюватися оператором відповідно до умов навколишнього середовища [22, розділ V, п.5]. Конструкція дисплея має забезпечувати можливість його повороту та нахилу для зручності користувача [22, розділ V, п.6]. Поверхня екрана не повинна створювати відблисків або віддзеркалень, що можуть викликати дискомфорт [4, розділ V, п.8]. Важливо, щоб усе випромінювання від екранних пристроїв, за винятком видимої частини електромагнітного спектра, було зведене до незначного (гранично допустимого) рівня [22, розділ III, п.2; розділ V, п.2]. Інженерно-психологічні аспекти проектування засобів відображення інформації наголошують на важливості врахування психофізіологічних можливостей людини для забезпечення ефективного та безпечного сприйняття інформації [19].
- Клавіатура: Клавіатура повинна бути відокремленою від монітора, мати можливість нахилу та матову поверхню для уникнення відблисків [22, розділ V, п.9, 10]. Розташування та характеристики клавіш (форма, розмір, маркування) мають сприяти зручності та точності введення даних [22, розділ V, п.10].
- Робочий стіл або робоча поверхня: Повинні мати достатні розміри для раціонального розміщення основного та допоміжного обладнання (екран, клавіатура, документи тощо) та характеризуватися низькою відбивною здатністю поверхні [22, розділ III, п.6].

- Робоче крісло: Має бути стійким та забезпечувати працівнику можливість легко рухатися і займати зручну робочу позу. Конструкція крісла повинна передбачати регулювання висоти сидіння, висоти та кута нахилу спинки. За необхідності працівнику має надаватися підставка для ніг [22, розділ III, п.7]. Ергономічний дизайн крісла є важливим для профілактики втоми та захворювань опорно-рухового апарату [19].

Гігієнічні вимоги до виробничого середовища

- Освітлення: На робочих місцях з ВДТ необхідно забезпечити належні умови освітлення, що створюють відповідний контраст між екраном та навколишнім середовищем і відповідають характеру зорової роботи [22, розділ III, п.4]. Нормативні вимоги до освітлення встановлені ДБН В.2.5-28:2018 [21]. Це включає забезпечення нормованих рівнів освітленості (наприклад, для офісних приміщень з ВДТ – 300-500 лк при загальному освітленні [21, Додаток Д, Таблиця Д.1]), рівномірності розподілу яскравості, обмеження прямої та відбитої блискості (сліпучої дії), а також коефіцієнта пульсації освітленості (не більше 10 % для приміщень з ВДТ [21, Додаток Д, Таблиця Д.1]). Особливу увагу слід приділяти розташуванню світильників та робочих місць для уникнення відблисків на екрані [21, п. 8.2.5]. Природне освітлення є пріоритетним і нормується значеннями коефіцієнта природної освітленості (КПО) [21, розділ 6].
- Шум: Рівні шуму на робочих місцях не повинні перевищувати допустимих значень, встановлених ДСН 3.3.6.037-99 [22, розділ II, п.4]. Устаткування, що є джерелом шуму, слід розміщувати таким чином, щоб мінімізувати його вплив на працівників.
- Інші фактори: Устаткування, що входить до складу робочої станції, не повинно виділяти надлишкового тепла, яке може спричинити дискомфорт для працівників [22, розділ V, п.11].

Вимоги до взаємодії "оператор-ЕОМ" та режимів праці і відпочинку

Програмне забезпечення, що використовується оператором, повинно відповідати виконуваним завданням, бути легким у використанні та, за потреби,

адаптованим до рівня знань і досвіду працівника [22, розділ V, п.12]. Роботодавець зобов'язаний за рахунок тривалості робочої зміни організувати внутрішні регламентовані перерви для відпочинку працівників, відповідно до ДСанПІН 3.3.2.007-98 [22, розділ II, п.5]. Дотримання раціональних режимів праці та відпочинку є важливим заходом психогігієни праці, спрямованим на профілактику перевтоми та збереження високої працездатності [19, с. 87-88].

Ретельне дотримання вищезазначених гігієнічних вимог до організації та обладнання робочих місць з ВДТ є запорукою створення безпечних, комфортних та ефективних умов праці для операторів.

ВИСНОВКИ

У даній дипломній роботі було успішно досягнуто поставленої мети: розроблено україномовну семантичну гру-головоломку «Словозв'яз» та проведено аналіз ефективності застосованого підходу на основі сучасних NLP-моделей у порівнянні з відомими аналогами.

В результаті виконання роботи були отримані наступні основні результати та зроблені висновки:

1. Проведено аналіз предметної області семантичних ігор, розглянуто принципи роботи та алгоритмічні основи існуючих популярних ігор Contexto.me (на базі GloVe) та Semantle.com (на базі Word2Vec). Обґрунтовано актуальність розробки україномовного аналога з використанням новітніх технологій векторних представлень (Розділ 1).

2. Досліджено теоретичні основи методів NLP, релевантних для задачі: розглянуто концепцію семантичних векторних просторів, проаналізовано статичні моделі ембеддінгів Word2Vec та GloVe, сучасні контекстуальні моделі на базі Transformer (зокрема, OpenAI text-embedding-3-large), роль косинусної схожості як міри близькості та можливості використання LLM (gpt-3.5-turbo) для генерації текстових даних (Розділ 2).

3. Спроектовано та реалізовано програмну систему гри «Словозв'яз», що включає:

- Клієнтську частину (frontend) на HTML, CSS, JavaScript, яка забезпечує інтерфейс користувача та ігрову логіку.
- Серверну частину (backend) на Python/Flask з базою даних SQLite/SQLAlchemy, що надає API для доступу до ігрових даних.
- Підсистему підготовки даних, що включає формування корпусу з ~14 000 українських іменників (на основі UberText 2.0 [15]), генерацію визначень за допомогою gpt-3.5-turbo [14], отримання ембеддінгів через text-embedding-3-

large [5, 11] та розрахунок/збереження рейтингів на основі косинусної схожості [13] (Розділ 3).

4. Проаналізовано якість семантичних рейтингів, отриманих за допомогою розробленого підходу (OpenAI text-embedding-3-large + визначення). Встановлено, що підхід генерує здебільшого семантично когерентні результати, ефективно фокусуючи представлення полісемічних слів на одному значенні завдяки використанню визначень. Водночас виявлено чутливість методу до якості згенерованих визначень та можливий залишковий вплив базової моделі (наприклад, близькість антонімів) (п. 4.2).

5. Проведено порівняльний аналіз характеру семантичної схожості, що генерується підходом «Словозв'яз», з підходами Word2Vec (локальний контекст, функціональна близькість) та GloVe (глобальний контекст, асоціативна близькість). Зроблено висновок, що підхід, реалізований у «Словозв'яз», потенційно забезпечує більш інтуїтивну для гравця семантичну близькість, сфокусовану на значенні слова, хоча й ціною ігнорування інших значень полісемічних слів (п. 4.3).

6. Проаналізовано ефективність застосованих технічних оптимізацій: кешування ембеддінгів, пакетна обробка запитів до API та прекомпіляція рейтингів. Доведено їхню критичну важливість для управління витратами на зовнішні API, скорочення часу підготовки даних та забезпечення високої швидкодії розгорнутого веб-додатку (п. 4.4).

7. Підтверджено практичне значення роботи, яке полягає у створенні функціональної україномовної семантичної гри, практичній апробації сучасних NLP-інструментів для української мови та внеску в розвиток україномовного цифрового контенту. Окреслено напрямки подальшого розвитку проєкту, включаючи покращення обробки полісемії, розширення словника та функціоналу гри, дослідження альтернативних моделей (п. 4.5).

Таким чином, усі поставлені у п. 1.3 завдання дипломної роботи були виконані, а мета роботи – досягнута. Розроблений додаток «Словозв'яз» демонструє життєздатність застосованого підходу та є основою для подальших досліджень та вдосконалень.

ПЕРЕЛІК ДЖЕРЕЛ

1. Mikolov T., Chen K., Corrado G., Dean J. Efficient Estimation of Word Representations in Vector Space [Електронний ресурс]. – 2013. – Режим доступу: https://arxiv.org/pdf/1301.3781

2. Pennington J., Socher R., Manning C. D. GloVe: Global Vectors for Word Representation [Електронний ресурс]. – Stanford University, 2014. – Режим доступу: https://nlp.stanford.edu/projects/glove/

3. Як людство взаємодіє з цифровими технологіями: звіт Digital 2024 [Електронний ресурс]. – MediaMaker, 2024. – Режим доступу: https://mediamaker.me/yak-lyudstvo-vzayemodiye-z-czyfrovymy-tehnologiyamy-zvit-digital-2024-8566/

4. Малига І. Є., Шматков С. І. Аналіз впливу різних векторних представлень слів на точність класифікації текстових даних // Вісник Харківського національного університету імені В. Н. Каразіна. Серія «Математичне моделювання. Інформаційні технології. Автоматизовані системи управління» [Електронний ресурс]. – 2023. – Вип. 59. – С. 49–55. – Режим доступу: https://periodicals.karazin.ua/mia/article/download/23703/21588/

5. Embeddings [Електронний ресурс] // OpenAI API Documentation. – OpenAI. – Режим доступу: https://platform.openai.com/docs/guides/embeddings

6. Almeida F., Xexéo G. Word Embeddings: A Survey [Електронний ресурс]. – 2019. – arXiv:1901.09069. – Режим доступу: https://arxiv.org/abs/1901.09069

7. Firth J. R. A synopsis of linguistic theory, 1930-1955 // Studies in Linguistic Analysis. – Oxford : Blackwell, 1957. – P. 1–32.

8. Vaswani A. et al. Attention is all you need // Advances in neural information processing systems [Електронний ресурс]. – 2017. – Vol. 30. – Режим доступу: https://papers.nips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html

9. Devlin J. et al. Bert: Pre-training of deep bidirectional transformers for language understanding [Електронний ресурс]. – 2018. – arXiv:1810.04805. – Режим доступу: https://arxiv.org/abs/1810.04805

10. Peters M. E. et al. Deep contextualized word representations [Електронний ресурс]. – 2018. – arXiv:1802.05365. – Режим доступу: https://arxiv.org/abs/1802.05365

11. OpenAI's Text Embeddings v3 [Електронний ресурс] // Pinecone Learn. – Pinecone, 2024. – Режим доступу: https://www.pinecone.io/learn/openai-embeddings-v3/

12. What is the difference between static and contextual embeddings? [Електронний ресурс] // Zilliz AI FAQs. – Zilliz. – Режим доступу: https://zilliz.com/ai-faq/what-is-the-difference-between-static-and-contextual-embeddings

13. Cohen M. X. Practical Linear Algebra for Data Science. – O'Reilly Media, 2022. – 326 p.

14. GPT-3.5 [Електронний ресурс] // OpenAI API Documentation. – OpenAI. – Режим доступу: https://platform.openai.com/docs/models/gpt-3-5

15. Chaplynskyi D. Introducing UberText 2.0: A Corpus of Modern Ukrainian at Scale // Proceedings of the Second Ukrainian Natural Language Processing Workshop (UNLP) [Електронний ресурс]. – Dubrovnik, Croatia : Association for Computational Linguistics, 2023. – P. 1–10. – Режим доступу: https://aclanthology.org/2023.unlp-1.1/

16. Flask Documentation (3.1.x) [Електронний ресурс] // Pallets Projects. – Pallets. – Режим доступу: https://flask.palletsprojects.com/

17. SQLAlchemy Documentation (2.0) [Електронний ресурс] // SQLAlchemy Authors. – Режим доступу: https://docs.sqlalchemy.org/

18. Gunicorn Documentation (23.0.0) [Електронний ресурс] // Gunicorn Authors. – Режим доступу: https://docs.gunicorn.org/

19. Кириченко В.В. Психологія праці та інженерна психологія: навчальний посібник. – Житомир: Вид-во ЖДУ ім. І. Франка, 2022. – 240 с.

20. Санітарні норми мікроклімату виробничих приміщень ДСН 3.3.6.042-99 [Електронний ресурс] : затв. Постановою Головного державного санітарного лікаря України від 01.12.1999 р. № 42. – Режим доступу: https://zakon.rada.gov.ua/laws/show/va042282-99

21. ДБН В.2.5-28:2018. Природне і штучне освітлення [Електронний ресурс] : чинні від 2019-03-01. – Київ : Мінрегіон України, 2018. – 133 с. – Режим доступу: https://e-construction.gov.ua/laws_detail/3074958732556240833?doc_type=2

22. Про затвердження Вимог щодо безпеки та захисту здоров'я працівників під час роботи з екранними пристроями [Електронний ресурс] : Наказ Міністерства соціальної політики України від 14.02.2018 № 207 : зареєстр. в Міністерстві юстиції України 25.04.2018 за № 508/31960. – Режим доступу: https://zakon.rada.gov.ua/laws/show/z0508-18

23. Petryk, M. , Bachynskyi, M. , Brevus, V. , Mudryk, I. , Mykhalyk, D. Analysis technology of neurological movements considering cognitive feedback influences of cerebral cortex signals CEUR Workshop Proceedings, 2022, 3309, pp. 45–54.

ДОДАТКИ

Додаток А

Тези конференції

Міністерство освіти і науки України
Тернопільська обласна військова адміністрація
Тернопільський національний технічний університет імені Івана Пулюя
Фізико-механічний інститут ім. Г.В. Карпенка НАН України
Інститут прикладних проблем механіки і математики
ім. Я. С. Підстригача НАН України
Інститут електродинаміки НАН України
Інститут фізики напівпровідників ім. В.Є. Лашкарьова НАН України
Наукове товариство імені Шевченка
Vilnius Gediminas Technical University (Литва)
Warsaw University of Technology Lighting Technology Division (Польща)
Iskenderun Technical University (Туреччина);
Lublin University of Technology (Польща)
Technical University of Košice (Словаччина)
Асоціація випускників ТНТУ

МАТЕРІАЛИ

Міжнародної науково-технічної конференції

«ФУНДАМЕНТАЛЬНІ ТА ПРИКЛАДНІ ПРОБЛЕМИ СУЧАСНИХ ТЕХНОЛОГІЙ»

присвячена

180-річчю з дня народження Івана Пулюя та 65-річчю з дня заснування Тернопільського національного технічного університету імені Івана Пулюя

28-29 травня 2025 року
Тернопіль

УДК 004.8:81'322
А. Конончук – ст. гр. СП-41, док. філософії. І. Мудрик
(Тернопільський національний технічний університет імені Івана Пулюя, Україна)

СТВОРЕННЯ УКРАЇНСЬКОЇ МОВНОЇ ГРИ НА ОСНОВІ EMBEDDING-МОДЕЛЕЙ

A. Kononchuk, Ph.D I. Mudryk

DEVELOPMENT OF A UKRAINIAN LANGUAGE GAME BASED ON EMBEDDING MODELS

У цій роботі описано процес створення української мовної гри «Словозв'яз», яка базується на визначенні семантичної близькості між словами. Ігровий процес побудований навколо введення користувачем слів, які порівнюються з цільовим словом за допомогою попередньо згенерованих embedding-представлень. Кожне слово в грі має контекстуально збагачене визначення, сформоване за допомогою GPT-моделі, що дозволяє точніше обчислити семантичну відстань між словами.

Семантичні представлення (embedding-и) для слів генеруються за допомогою Embedding API від OpenAI, що забезпечує високий рівень якості та точності при порівнянні значень. Технічна реалізація включає бекенд на Flask (Python), фронтенд на HTML/CSS/JS та зберігання попередньо обчислених даних у форматі JSON. Словникова база гри складається з 15 000 найуживаніших українських іменників, відібраних з частотного лексичного ресурсу UberText.

Генерація визначень для кожного слова за допомогою GPT API дозволила надати embedding-представленням необхідного контексту, що значно підвищило точність ранжування слів за змістовною подібністю. Згенеровані JSON-файли з ранжуванням зберігаються не лише у файловій системі, але також структуровано зберігаються в базі даних у вигляді архівних ігор. Це дозволяє забезпечити швидкий доступ до історичних даних та ефективно реалізувати щоденні та архівні ігри.

Гру розгорнуто на хостинговій платформі Render, де вона доступна для користувачів у відкритому доступі. Результати роботи демонструють можливість ефективного використання embedding-моделей для реалізації україномовних інтерактивних систем на основі смислового аналізу.

Література:

1. O. Bryk, I. Mudryk, M. Holubovskyi, Y Stoianov. Machine learning models and methods aspects of processing unstructured data. Proceedings of the 1st International Workshop on Bioinformatics and Applied Information Technologies (BAIT 2024) Zboriv, Ukraine, 2024. pp. 64-74.
2. OpenAI. (2024). OpenAI API documentation – Text embeddings. URL: https://platform.openai.com/docs/guides/embeddings.
3. Pastukh O., Petryk M., Bachynskiy M., Mudryk I., Stefanyshyn V. Processing of Cerebral Cortex Neurosignals from EEG Sensors and Recognizin Specific Types of Mechanical Movements Elements of Pacient Limbs under the Cognitive Feedback Influenses // International Workshop on Computer Information Technologies in Industry 4.0. 2023. №3468. С. 61–70.
4. Render – Cloud Application Platform. URL: https://render.com.
5. UberText – частотний лексичний ресурс української мови. URL: https://lang.org.ua/en/ubertext/

Додаток Б

Код генерації визначень для слів та для генерації рейтингів

```
import openai
import json
import os
import re
import time
import concurrent.futures
import threading
from dotenv import load_dotenv

load_dotenv()
client = openai.OpenAI(api_key=os.environ.get("OPENAI_API_KEY"))

INPUT_FILE = "wordlist.txt"
OUTPUT_FILE = "definitions.json"
definitions = {}
lock = threading.Lock()

if os.path.exists(OUTPUT_FILE) and os.path.getsize(OUTPUT_FILE)
> 0:
    with open(OUTPUT_FILE, "r", encoding="utf-8") as f:
        try:
            definitions = json.load(f)
        except json.JSONDecodeError:
            pass  # Ігноруємо помилку парсингу, якщо файл
пошкоджений

with open(INPUT_FILE, "r", encoding="utf-8") as f:
    words = [line.strip() for line in f if line.strip()]

def remove_word_repetition(word, definition):
    pattern     =     re.compile(rf"^\s*{re.escape(word)}\s*(—|-
|:)?\s*", re.IGNORECASE)
    return pattern.sub("", definition, count=1)

def save_definition(word, definition_text):
    with lock:
        definitions[word] = definition_text
        with open(OUTPUT_FILE, "w", encoding="utf-8") as f:
            json.dump(definitions,    f,    ensure_ascii=False,
indent=2)

def  generate_definition(word,  max_retries=3):  #  Зменшено
max_retries для прикладу
    prompt = (
        f"Дай коротке, лаконічне визначення для слова '{word}'
українською мовою в одному реченні. "
```

```
        "Не повторюй саме слово на початку визначення."
    )
    attempt = 0
    while attempt < max_retries:
        try:
            response = client.chat.completions.create(
                model="gpt-3.5-turbo",
                messages=[
                    {"role": "system", "content": "Ти експерт зі
створення коротких визначень."},
                    {"role": "user", "content": prompt}
                ],
                temperature=0.5, max_tokens=100
            )
            raw_definition                                    =
response.choices[0].message.content.strip()
            return      word,      remove_word_repetition(word,
raw_definition)
        except Exception as e:
            if  "rate_limit_exceeded"  in  str(e).lower()  and
attempt < max_retries -1:
                time.sleep(5  *  (2  **  attempt))  #  Скорочений
backoff
                attempt += 1
            else:
                return word, None # Повертаємо None у разі інших
помилок або після всіх спроб
    return word, None

def main_definitions():
    to_process  =  [word  for  word  in  words  if  word  not  in
definitions]
    print(f"Генерація визначень для {len(to_process)} слів...")
    with concurrent.futures.ThreadPoolExecutor(max_workers=5) as
executor:
        futures  =  [executor.submit(generate_definition,  word)
for word in to_process]
        for future in concurrent.futures.as_completed(futures):
            word_res, definition_res = future.result()
            if definition_res:
                save_definition(word_res, definition_res)
    print("Генерація визначень завершена.")

if __name__ == "__main__":
    main_definitions()

import openai
import json
import os
import numpy as np
from tqdm import tqdm # Залишено для наочності прогресу
from dotenv import load_dotenv
```

```
    from datetime import timedelta, date

    load_dotenv()
    client = openai.OpenAI(api_key=os.environ.get("OPENAI_API_KEY"))

    CACHE_FILE = "embeddings_cache.json"
    BATCH_SIZE = 100 # Залишено для ілюстрації батчингу
    BASE_DATE = date(2025, 6, 2)

    os.makedirs("precomputed", exist_ok=True)

    embedding_cache = {}
    if os.path.exists(CACHE_FILE):
        with open(CACHE_FILE, "r", encoding="utf-8") as f:
            try:
                embedding_cache = json.load(f)
            except json.JSONDecodeError:
                pass # Ігноруємо, якщо кеш пошкоджений

    def get_embeddings_batch(phrases, model="text-embedding-3-
large"):
        uncached_phrases = [p for p in phrases if p not in
embedding_cache]
        if uncached_phrases:
            for i in range(0, len(uncached_phrases), BATCH_SIZE):
                batch = uncached_phrases[i:i + BATCH_SIZE]
                try:
                    response = client.embeddings.create(input=batch,
model=model)
                    for phrase_item, obj in zip(batch,
response.data):
                        embedding_cache[phrase_item] = obj.embedding
                except Exception:
                     pass # Пропускаємо помилковий batch для
скорочення
            with open(CACHE_FILE, "w", encoding="utf-8") as f:
                json.dump(embedding_cache, f, ensure_ascii=False)
        return [embedding_cache.get(p) for p in phrases] # .get()
поверне None, якщо ключа немає

    def cosine_similarity(a, b):
        if a is None or b is None: return 0.0
        vec_a, vec_b = np.array(a), np.array(b)
        norm_a, norm_b = np.linalg.norm(vec_a),
np.linalg.norm(vec_b)
        return np.dot(vec_a, vec_b) / (norm_a * norm_b) if norm_a and
norm_b else 0.0

    def generate_rankings_for_word(target_word, game_date,
definitions, all_words):
        target_phrase = definitions.get(target_word, target_word)
        target_emb_list = get_embeddings_batch([target_phrase])
```

```
        if not target_emb_list or target_emb_list[0] is None: return
[]
        target_emb = target_emb_list[0]

        phrases_to_embed = [definitions.get(w, w) for w in all_words]
        all_embeddings = get_embeddings_batch(phrases_to_embed)

        word_embedding_map = {word: emb for word, emb in
zip(all_words, all_embeddings) if emb is not None}

        scored = [(word, cosine_similarity(emb, target_emb))
                  for word, emb in tqdm(word_embedding_map.items(),
desc=f"Рейтинг для {target_word}", leave=False)]

        scored.sort(key=lambda x: x[1], reverse=True)
        ranked = [{"word": w, "similarity": float(s), "rank": r} for
r, (w, s) in enumerate(scored, start=1)]

        filename = f"precomputed/{game_date}.json"
        with open(filename, "w", encoding="utf-8") as f:
            json.dump(ranked, f, ensure_ascii=False, indent=2)
        print(f" {target_word} -> {filename}")
        return ranked

    def main_rankings():
        with open("daily_words.txt", "r", encoding="utf-8") as f:
            daily_words = [line.strip() for line in f if
line.strip()]
        with open("wordlist.txt", "r", encoding="utf-8") as f:
            words = [line.strip() for line in f if line.strip()]

        definitions_data = {}
        if os.path.exists("definitions.json"):
            with open("definitions.json", "r", encoding="utf-8") as
f:
                try:
                    definitions_data = json.load(f)
                except json.JSONDecodeError:
                    pass # Ігноруємо помилку

        for i, current_target_word in enumerate(daily_words):
            current_day = BASE_DATE + timedelta(days=i)
            if                                                  not
os.path.exists(f"precomputed/{current_day}.json"):
                 generate_rankings_for_word(current_target_word,
current_day, definitions_data, words)
            else:
                 print(f" Пропущено: {current_target_word} (файл
існує)")

    if __name__ == "__main__":
        main_rankings()
```

Додаток В

Диск