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

Факультет інформатики та обчислювальної техніки

Кафедра інформаційних систем та технологій

«На правах рукопису»
УДК _004.043_

До захисту допущено:
Завідувач кафедри
_____ Олександр РОЛІК
«___»_____2022 р.

Магістерська дисертація

на здобуття ступеня магістра
за освітньо-професійною програмою «Інформаційне забезпечення робототехнічних систем»
зі спеціальності 126 «Інформаційні системи та технології»
на тему: «Інтелектуальна система аналізу достовірності інформації новинних ресурсів»

Виконав (-ла):
студент (-ка) VI курсу, групи ІК-11мп
Кравченко Дмитро Олександрович _____

Керівник:
Доцент, кандидат технічних наук, доцент
Богданова Наталія Володимирівна _____

Рецензент:
професор, доктор технічних наук, професор
Мельник Ігор Віталійович _____

Засвідчую, що у цій магістерській дисертації немає запозичень з праць інших авторів без відповідних посилань.
Студент (-ка) _____

Київ – 2022 року

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра інформаційних систем та технологій

Рівень вищої освіти – другий (магістерський)

Спеціальність – 126 «Інформаційні системи та технології»

Освітньо-професійна програма «Інформаційне забезпечення робототехнічних систем»

ЗАТВЕРДЖУЮ
Завідувач кафедри
Олександр РОЛІК
«___»_____2022 р.

ЗАВДАННЯ
на магістерську дисертацію студенту
Кравченку Дмитру Олександровичу

1. Тема дисертації «Інтелектуальна система аналізу достовірності інформації новинних ресурсів», науковий керівник дисертації Богданова Наталія Володимирівна, кандидат наук, доцент, затверджені наказом по університету від «09» 11 2022 р. № 4115-с

2. Термін подання студентом дисертації 12.12.2022 р

3. Об'єкт дослідження веб-сервіс аналізу достовірності інформації.

4. Вихідні дані – результат перевірки інформації, факти, що її спростовують чи підтверджують.

5. 5. Перелік завдань, які потрібно розробити: аналіз задачі перевірки достовірності інформації, огляд існуючих рішень, створення математичної моделі, розробка багаторівневої системи збору пов'язаної інформації, її сегментування та опрацювання для встановлення істинності заданого твердження

6. Орієнтовний перелік графічного (ілюстративного) матеріалу - діаграми класів та послідовності, схема роботи використаних алгоритмів, діаграма компонентів.

7. Орієнтовний перелік публікацій

9. Дата видачі завдання 05.09.2022р.

Календарний план

№ з/п	Назва етапів виконання магістерської дисертації	Термін виконання етапів магістерської дисертації	Примітка
1	Огляд літературно-інформаційних джерел. Характеристика об’єкта.	27.10.2022-01.11.2022	
2	Постановка та формалізація задачі	02.11.2022-08.11.2022	
3	Розробка програмного забезпечення	09.11.2022-22.11.2022	
4	Обробка вхідних даних. Реалізація та застосування обраного алгоритму. Збір та аналіз результатів	23.11.2022-06.12.2022	
5	Налагодження програми	07.12.2022-09.12.2021	
6	Оформлення висновків	10.12.2022-12.12.2022	

Студент Дмитро КРАВЧЕНКО

Науковий керівник Наталія БОГДАНОВА

РЕФЕРАТ

Структура магістерської дисертації. Пояснювальна записка магістерської дисертації складається з шести розділів, містить 31 таблицю, 7 рисунків, 8 додатків, 13 джерел.

Об'єктом дослідження є веб-сервіс аналізу достовірності інформації.

Предметом дослідження є алгоритми отримання та обробки текстів, що виступають доказами або запереченнями твердження, що має бути перевіреним.

Метою роботи є дослідження, розробка та вдосконалення моделей пошуку уривків текстів та методів відбору речень.

Новизною роботи є покращення аналізу відношення тверджень при вирішені задачі NLI (Natural Language Inference) в умовах наявності відволікаючої інформації в контексті тверджень, що перевіряються на істинність.

В результаті виконання магістерської дисертації було розроблено програмний продукт, що дозволяє перевірити твердження навіть у великій кількості контекстуальної інформації.

Ключові слова: NLI, дата-сет, батч, аналіз природньої мови, факт-чекінг, токенізація, фейк.

SUMMARY

The structure of the master's thesis**.** The explanatory note of the master's thesis consists of six sections, contains 31 tables, 7 figures, 8 appendices, 13 sources.

The object of the study is a web service for analyzing the reliability of information.

The subject of research is algorithms for obtaining and processing texts that serve as proofs or refutations of statements that need to be verified.

The purpose of the work is to research, develop and improve models for searching text fragments and sentence selection methods.

The novelty of the work is the improvement of the analysis of the relation of statements when solving the NLI (Natural Language Inference) problem in the presence of distracting information in the context of statements that are checked for truth.

As a result of the master's thesis, a software product was developed that allows you to verify statements even with a large amount of contextual information.

Keywords: NLI, data set, batch, natural language analysis, fact-checking, tokenization, fake.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧЕНЬ, ТЕРМІНІВ

NLP (natural language processing) – загальний напрямок штучного інтелекту та математичної лінгвістики який вивчає проблеми комп'ютерного аналізу та синтезу текстів природними мовами.

NLI (natural language inference) – завдання автоматичного визначення логічного зв'язку між текстами.

BERT (bidirectional encoder representations from transformers) – метод машинного навчання на основі перетворювача для попередньої підготовки до обробки природної мови.

API (application programming interface) – програмний інтерфейс.

ВСТУП

В нинішньому світі інформація стала потужною зброєю. Фейки ж тепер є нічим інакшим як інноваційним інструментом управління інформаційними процесами у засобах масової інформації, що використовуються у сучасних гібридних війнах. Специфіка цих інструментів полягає у комплексній маніпуляції громадською думкою, що відбувається на основі психологічного тиску, використання привабливого контенту та ефективних методів поширення інформації.

Ретельне дослідження та практична розробка алгоритму та системи аналізу достовірності інформації, виявлення фейків, на сьогодні є актуальним завданням боротьби з великою кількістю недостовірної інформації в медіа просторі. В умовах гібридних протистоянь ця технологія є однією з ключових і затребуваною на теперішній момент.

Розроблювана система має на меті зробити факт-чекінг, головний інструмент боротьби з фейковою інформацією, більш простим та доступним для користувачів інтернету. Маючи можливість легко перевірити інформацію з будь-якої публікації в засобах масової інформації, посту в соціальній мережі чи просто перевірити будь-яке твердження люди стануть менш уразливими до ІПСО та краще обізнаними, тобто матимуть кращий інформаційний захист.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Обґрунтування доцільності розробки

З появою соціальних мереж і багатьох окремих джерел новин в Інтернеті поширення дезінформації стало серйозною проблемою з потенційно шкідливими соціальними наслідками. Фейкові новини можуть маніпулювати громадською думкою, створювати конфлікти та викликати необґрунтований страх і підозри. Величезна кількість неперевіреного онлайн-контенту призвела до створення зовнішніх пост-перевірочних організацій, таких як PolitiFact, FactCheck.org, Snopes, тощо, зі спеціальними ресурсами для перевірки інформації розміщеної в інтернеті. Однак ручна перевірка фактів займає багато часу, і її важко проводити у великих масштабах. Можливість автоматичної перевірки фактів має вирішальне значення для мінімізації негативного соціального впливу.

1.2 Опис предметної області

Протягом останніх кількох років фейкові новини стають все більш поширеними. Фальшиві новинні статті, як правило, надходять із сатиричних новинних веб-сайтів або окремих веб-сайтів із стимулом поширювати неправдиву інформацію, або як приманку для кліків, або для досягнення мети.

Оскільки ці статті зазвичай спрямовані на навмисне просування упередженої чи невірної інформації, їх важко виявити. Визначаючи джерело інформації, необхідно звернути увагу на багато атрибутів, включаючи, але не обмежуючись вмістом електронної пошти та взаємодії в соціальних мережах. Зокрема, у фейкових новинах ця мова зазвичай є більш підбурювальною, ніж у справжніх статтях, частково тому, що мета полягає в тому, щоб заплутати користувача. Більше того, методи моделювання, такі як кодування n-грамів і сумка слів, слугували іншими лінгвістичними техніками для визначення легітимності новин.

Крім того, дослідники визначили, що візуальні підказки також відіграють важливу роль у класифікації статті. Зокрема, деякі функції можуть бути розроблені,

щоб оцінити, чи зображення було легітимним, і надати нам більше ясності щодо новин. Існує також багато особливостей соціального контексту, які можуть зіграти свою роль, а також модель поширення новин. Веб-сайти, такі як "Snopes", намагаються виявити цю інформацію вручну, тоді як деякі університети намагаються побудувати математичні моделі для цього самостійно.

Адаптація соціальних медіа як законної та широко використовуваної платформи викликала широке занепокоєння щодо фейкових новин у цій сфері. Поширення фейкових новин через соціальні медіа-платформи, такі як Facebook, Twitter та Instagram, створює можливість надзвичайно негативного впливу на суспільство, тому нові галузі досліджень щодо виявлення фейкових новин у соціальних мережах набирають обертів.

Однак виявлення фейкових новин у соціальних мережах створює проблеми, які роблять попередні методи аналізу даних і виявлення неадекватними. Таким чином, дослідники закликають до додаткової роботи щодо фейкових новин, які характеризуються проти психології та соціальних теорій, а також адаптації існуючих алгоритмів інтелектуального аналізу даних для застосування в соціальних мережах. Крім того, було опубліковано кілька наукових статей, які закликають продовжити пошук автоматичних способів фільтрації фейкових новин із хронології соціальних мереж.

Після президентських виборів у Сполучених Штатах у 2016 році фейкові новини були популярною темою для обговорення президента Трампа та ЗМІ. Реальність фейкових новин стала всюдисущою, і багато досліджень було спрямовано на розуміння, ідентифікацію та боротьбу з фейковими новинами. Також низка дослідників почали з використання фейкових новин для впливу на президентську кампанію 2016 року.

Одне дослідження виявило докази того, що фейкові новини на підтримку Трампа вибірково націлювалися на консерваторів і прихильників Трампа в 2016 році. Дослідники виявили, що сайти соціальних мереж, зокрема Facebook, є потужними платформами для розповсюдження певних фейкових новин цільовим групам для звернення до них. їхні настрої під час президентських перегонів 2016 року. Крім того,

дослідники зі Стенфорда, Нью-Йоркського університету та NBER знайшли докази того, що протягом 2016 року було високо залучено до фейкових новин у Facebook і Twitter.

Автоматизована перевірка фактів – це складне завдання, яке включає отримання доказів з подальшим їх обґрунтуванням. Для пошуку відповідних доказів із тіл документів існуючі системи зазвичай використовують традиційний розріджений пошук, який може показувати не найкращу ефективність, особливо коли відповідні уривки містять кілька слів, що збігаються з твердженнями, які потрібно перевірити.

Моделі щільного пошуку довели свою ефективність у відповідях на питання, оскільки ці моделі можуть краще вловлювати прихований семантичний зміст тексту. Робота «Прихований пошук для широкомасштабної перевірки фактів і відповідей на запитання з навчанням NLI» є першою, у якій використовується щільний пошук для перевірки фактів. Автори створили новий набір даних під назвою Factual-NLI, який складається з пар твердження-докази з набору даних FEVER, а також синтетичних прикладів, згенерованих із контрольних наборів даних Question Answering. Вони продемонстрували, що використання Factual-NLI для навчання щільного ретривера може значно покращити пошук доказів.

Хоча набір даних FEVER уможливив систематичну оцінку автоматизованих систем перевірки фактів, він погано відображає природу даних реального світу, з безліччю зайвої інформації. Тому має сенс використати розширений набір даних Factual NLI+, як розширення набору даних FEVER із синтетичними прикладами з наборів даних, з відповідями на питання та фрагментами шуму з результатів веб-пошуку. Далі в даній роботі буде перевірено, як щільні репрезентації можуть покращити отримання уривків на першому етапі перевірки фактів у контексті з «шумами» та зробити пошук відповідних доказів більш зручним у великому масштабі.

1.3 Цілі та задачі розробки

Цілями розробки є:

- розробка веб-сервісу перевірки достовірності інформації;
- забезпечення зручного інтуїтивно-зрозумілого інтерфейсу;
- забезпечення можливості отримання доказів результатів аналізу інформації;
- забезпечення можливості отримати перевірені відповіді на поставлені питання.

Для досягнення поставлених цілей необхідно вирішити наступні задачі:

- створити систему збору та аналізу пов’язаної інформації;
- створити веб додаток з API для забезпечення доступу до ядра системи;
- створити веб інтерфейс.

Висновки до розділу

В першому розділі дипломного проєкту:

- обґрунтовано доцільності розробки застосунку;
- було розкрито обрану предметну область, описано основні цілі та задачі розробки;
- охарактеризовано функціональну модель системи.

2 АНАЛІЗ ІСНУЮЧИХ РІШЕНЬ

2.1 Критерії для аналізу аналогів

На ринку існують системи аналоги, подібні до розроблювальної. Для формування вимог до системи, вибору оптимального підходу до розробки та врахування досвіду інших систем, потрібно проаналізувати зразки подібних систем.

Детальний аналіз допоможе виявити переваги та недоліки таких систем. На основі проведеного аналізу можна буде сформувати вимоги до функціоналу власної майбутньої системи.

З критеріїв аналізу варто виділити наступні:

- простота використання;
- багатофункціональність;
- існуючі інтерфейси;
- точність аналізу інформації;
- доступність.

2.2 The Factual

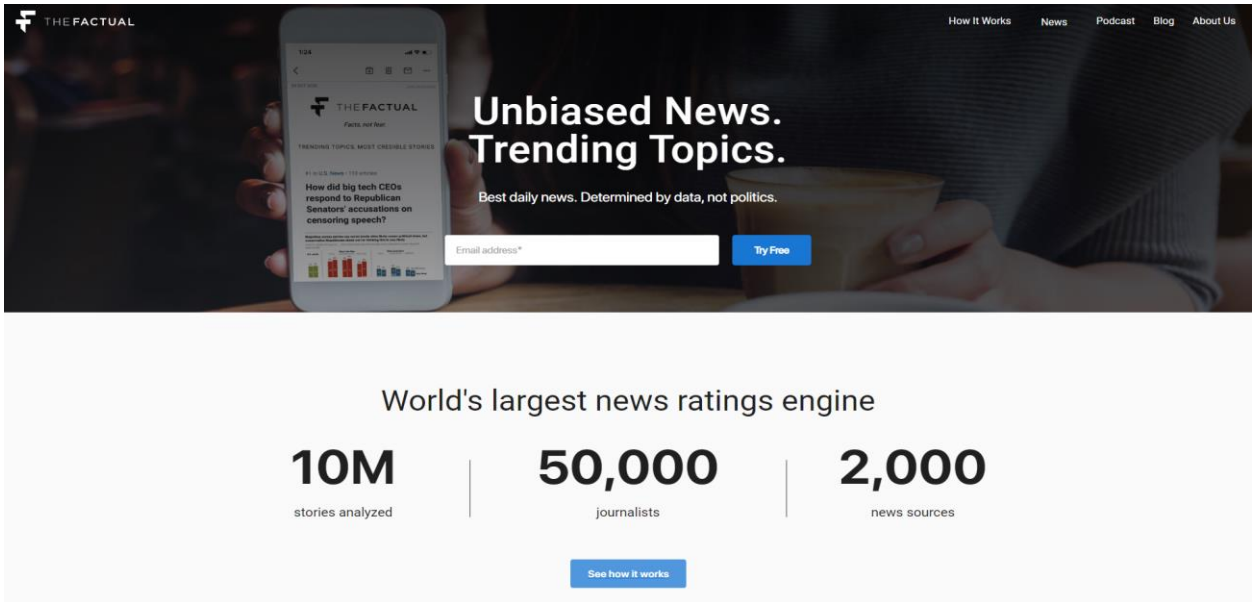

Рисунок 2.1 – Сервіс The Factual

The Factual (рисунок 2.1) – сервіс, що дозволяє отримати доступ до новин проаналізованих нейронною мережею розробника. Завдяки подібному аналізу, новини та статті отримують рейтинг довіри та фільтруються.

За своєю суттю The Factual – це алгоритм, який щоденно оцінює 10 000+ статей новин на предмет їх інформативності. На основі цих даних The Factual пропонує п’ять продуктів, щоб отримувати найбільш правдиві новини:

- інформаційний бюлетень. За допомогою редакторів-людей для перехресної перевірки алгоритму The Factual випускає щоденну інформаційну розсилку з найбільш інформативними статтями з усього політичного спектру на актуальні теми новин;
- веб-сайт – сайт, який показує актуальні теми в реальному часі, групуючи тисячі пов’язаних статей і дозволяючи фільтрувати за політичними пристрастями, оцінками тощо;
- додаток для iOS і Android – поєднує щоденні інформаційні бюлетені та наш веб-сайт у простій програмі, щоб ви могли швидко бути в курсі актуальних і важливих новин;
- розширення для Chrome – розширення для веб-переглядача, яке виконує дві функції: (1) миттєво оцінює інформативність будь-якої новинної статті та (2) додає рейтинги статей прямо у ваші канали Twitter і Facebook;
- IsThisCredible.com – простий веб-сайт, на якому можна отримати рейтинг будь-якої знайденої статті новин або знайти найактуальніші новини на будь-яку тему.

Кожна стаття отримує оцінку від 1 до 100% на основі чотирьох показників: якості сайту, досвіду автора, якості та різноманітності джерел і тону написання. Кожен із цих показників відповідає на ключові запитання щодо інформаційної цінності статті.

- якість сайту: чи має цей сайт історію створення інформативних статей із хорошими джерелами;

- експертиза автора: чи має автор досвід написання добре досліджених інформативних статей на цю тему? Чи автор зосереджується на темі і, отже, може мати певний досвід;
- якість і різноманітність джерел: скільки унікальних джерел і прямих цитат використано в статті? Який рейтинг сайту цих джерел;
- тон статті: чи була стаття написана в нейтральному, байдужому тоні чи вона була емоційною мовою.

Ці чотири метрики поєднуються, щоб отримати єдину відсоткову оцінку, яку ми інтерпретуємо як ймовірність того, що стаття є інформативною. Оцінки вище 75% вважаються інформативними, тоді як оцінки нижче 50% менш імовірно.

Переваги:

- цікава система рейтингу якості інформації
- зручні користувацькі інтерфейси

Недоліки:

- обов’язкова реєстрація акаунту;
- неможливість аналізу конкретної інформації, тільки надані новини;
- багато зайвої інформації;
- людське втручання в фінальний рейтинг достовірності інформації.

2.3 Check від Meedan

Check розроблений Meedan (рисунок 2.2) – набір інструментів для розробки ботів для популярних месенджерів, що можуть перевіряти достовірність інформації окремих повідомлень. Має великий та гнучкий набір інструментів для побудови потоків аналізу текстової інформації з налаштуванням під різні параметри та критерії.

Check є платформою для спільної перевірки цифрових носіїв. Проект Check створює онлайн-інструменти для підвищення якості розслідувань громадянської журналістики та допомагає обмежити швидке поширення чуток і дезінформації в Інтернеті. Використовуючи Check, ви можете швидко створити команду, додати

посилання на соціальні мережі для перевірки фактів, задати ключові запитання для дослідження новин, зібраних вашою командою.

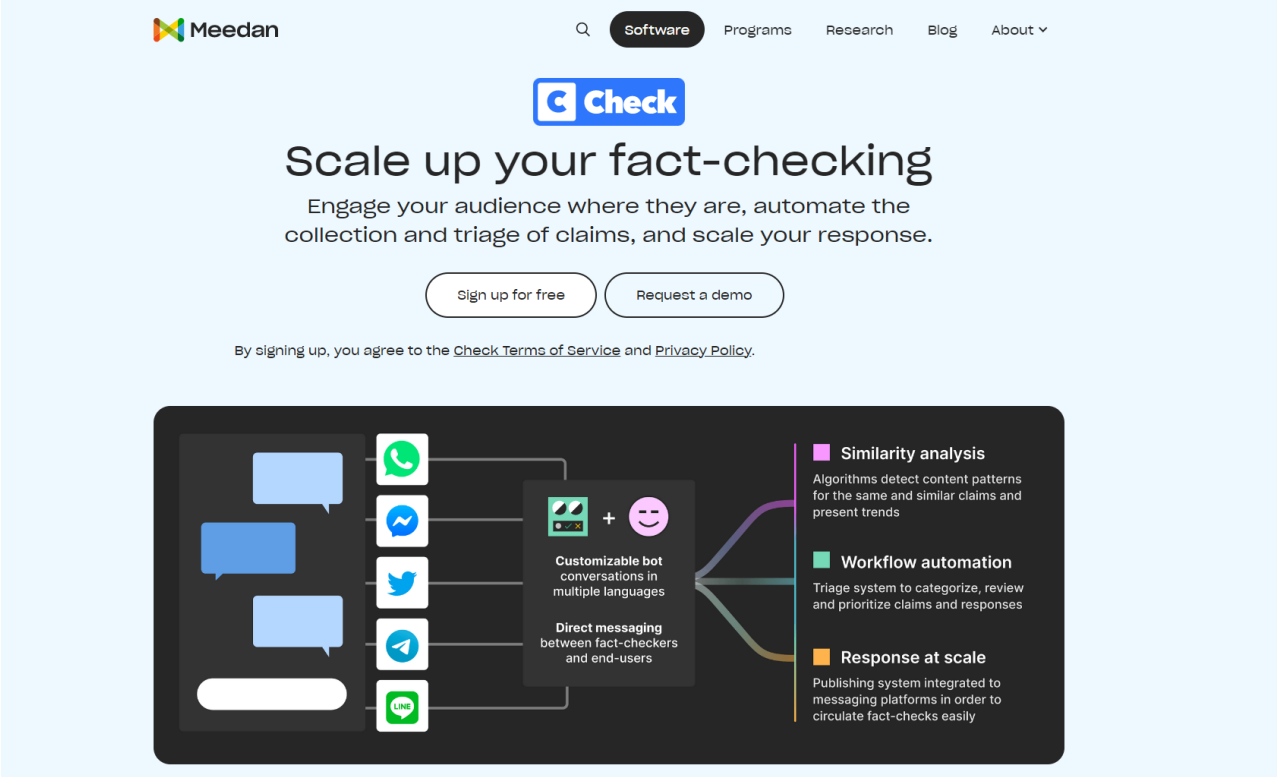

Рисунок 2.2 – Інструмент Check

Розширення для веб-переглядача Check дозволяє швидко додавати медіа-елементи, на які ви натрапляєте в Інтернеті, до вашої команди Check для дослідження та відповідати на анотації щодо цих елементів прямо з бічної панелі розширення.

Переваги:

- широкий функціонал;
- гнучкість налаштування процесу аналізу;
- працює з багатьма популярними месенджерами.

Недоліки:

- обов’язкова реєстрація акаунту;
- надає інструменти, але не є готовою системою;
- має досить високий поріг входження для використання користувачем ;
- не має готового додатку.

2.4. Logically

Logically (рисунок 2.3) – мобільний додаток з новинною стрічкою, що використовує нейронну мережу для перевірки достовірності інформації в постах які вони ретранслюють з популярних новинних джерел. Користувачі зазначають, що система має високі показники точності перевірки інформації.

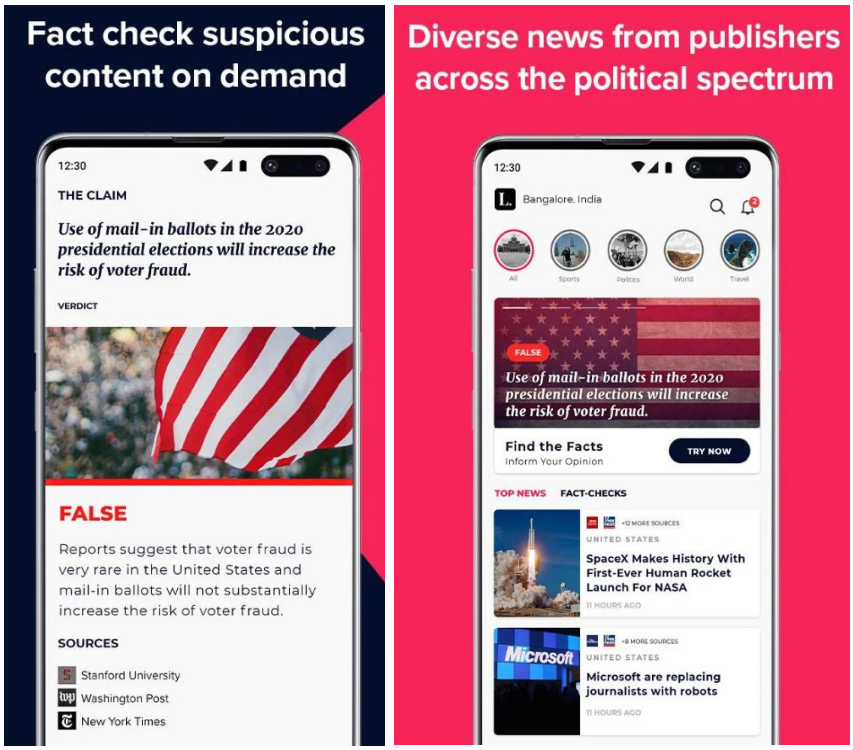

Рисунок 2.3 – Додаток Logically

Додаток використовує такі джерела:

- Yahoo News;
- The Hill;
- Jim Roberts Realty;
- The Boston Globe;
- Fox News;
- The Straits Times;
- POLITICO;
- Portland Press Herald;

- The Ledger;
- USA Today;
- Huffpost.

Переваги:

- висока якість аналізу достовірності інформації;
- безкоштовний мобільний додаток зі зручним інтерфейсом.

Недоліки:

- неможливість аналізу конкретної інформації, тільки надані новини;
- багато зайвої інформації;
- проблеми з продуктивністю на Android системах.

2.5 Аналіз розглянутих рішень

Як результат аналізу схожих ресурсів, було отримано їх переваги та недоліки, створено порівняльну таблицю, та визначено, які особливості повинен мати мій проєкт. Підсумковий аналіз наведено у порівняльній таблиці 2.1.

Таблиця 2.1 – Порівняльна таблиця

Критерій	The Factual	Check by Meedan	Logically	Власна реалізація
Обов’язкова реєстрація аккаунту	ТАК	ТАК	ТАК	НІ
Можливість перевірити будь-яку бажану інформацію	НІ	ТАК	НІ	ТАК
Доступність на різних платформах	ТАК	НІ	НІ	ТАК
Нагромадження зайвої інформації	ТАК	НІ	ТАК	НІ

Продовження таблиці 2.1

Критерій	The Factual	Check by Meedan	Logically	Власна реалізація
Повна автоматизація процесу факт-чекінгу	НІ	НІ	ТАК	ТАК

Висновки до розділу

В другому розділі дипломного проєкту було проаналізовано готові рішення, встановлено переваги та недоліки кожного з популярних аналогів, що значною мірою допомогло сформулювати вимоги до розроблюваного додатку.

3 АНАЛІЗ АЛГОРИТМІВ ОБРОБКИ ПРИРОДНЬОЇ МОВИ

3.1 Визначення NLP

Natural Language Processing (далі – NLP) – обробка природньої мови – підрозділ інформатики та AI, присвячений тому, як комп'ютери аналізують природні (людські) мови. NLP дозволяє застосовувати алгоритми машинного навчання для тексту та мовлення.

Наприклад, ми можемо використовувати NLP, щоб створювати системи на кшталт розпізнавання мови, узагальнення документів, машинного перекладу, виявлення спаму, розпізнавання іменованих сутностей, відповіді питання, автокомплита, предиктивного введення тексту тощо.

Сьогодні у багатьох з нас є смартфони з розпізнаванням мови – у них використовується NLP для того, щоб розуміти нашу мову. Також багато людей використовують ноутбуки із вбудованим у ОС розпізнаванням мови.

3.2 Токенізація за реченнями

Токенізація (іноді – сегментація) за реченнями – це процес поділу письмової мови на речення-компоненти. Ідея виглядає досить простою. В англійській та деяких інших мовах ми можемо виокремлювати речення кожного разу, коли знаходимо певний знак пунктуації – крапку.

Але навіть в англійській це завдання нетривіальне, тому що крапка використовується і в скороченнях. Таблиця скорочень може допомогти під час обробки тексту, щоб уникнути неправильної розстановки меж речень. У більшості випадків для цього використовуються бібліотеки, тому можна особливо не перейматися деталями реалізації.

3.3 Токенізація за словами

Токенізація (іноді – сегментація) за словами – це процес поділу речень на слова-компоненти. В англійській та багатьох інших мовах, які використовують ту чи іншу версію латинського алфавіту, пробіл – це непоганий роздільник слів.

Тим не менш, можуть виникнути проблеми, якщо ми будемо використовувати тільки прогалину – в англійській складові іменники пишуться по-різному і іноді через пропуск. І тут знову нам допомагають бібліотеки.

3.4 Лематизація та стемінг тексту

Зазвичай тексти містять різні граматичні форми однієї й тієї ж слова, і навіть можуть зустрічатися однокореневі слова. Лематизація і стемінг мають на меті привести всі зустрічаються словоформи до однієї, нормальної словникової форми.

Лематизація та стемінг – це окремі випадки нормалізації і вони відрізняються.

Стеммінг – це грубий евристичний процес, який відрізає "зайве" від кореня слів, часто це призводить до втрати словотвірних суфіксів.

Лематизація – це тонший процес, який використовує словник і морфологічний аналіз, щоб у результаті призвести слово до його канонічної формі – лемі.

Відмінність у тому, що стеммер (конкретна реалізація алгоритму стеммінгу – прим. перекладача) діє без знання контексту і, відповідно, не розуміє різницю між словами, які мають різний зміст залежно від частини мови. Однак у стеммерів є свої переваги: їх простіше впровадити і вони працюють швидше. Плюс, нижча «акуратність» може мати значення у деяких випадках.

Таким чином, лемматизація та стемінг є техніками попередньої обробки, що означає, що ми можемо застосувати один із двох алгоритмів НЛП на основі наших потреб перед тим, як рухатися вперед із проектом НЛП, щоб звільнити простір даних і підготувати базу даних.

І лемматизація, і стікування є надзвичайно різноманітними процедурами, які можна виконувати різними способами, але кінцевий ефект однаковий для обох: зменшена область пошуку для проблеми, з якою ми маємо справу.

Приклади:

- слово good – це лема для слова better. Стеммер не побачить цей зв'язок, тому що тут потрібно звірятися зі словником;
- слово play – базова форма слова playing. Тут упораються і стемінг, і лематизація;
- слово meeting може бути як нормальною формою іменника, так і формою дієслова to meet, залежно від контексту. На відміну від стеммінгу, лематизація спробує вибрати правильну лему, спираючись на контекст.

3.5 Стоп-слова

Стоп-слова – це слова будь-якої мови, які не додають особливого значення реченню. Їх можна сміливо ігнорувати, не жертвуючи змістом речення. Для деяких пошукових систем це одні з найпоширеніших коротких службових слів, наприклад the, is, at, which і on. У цьому випадку стоп-слова можуть спричинити проблеми під час пошуку фраз, які їх містять, зокрема в таких назвах, як «The Who» або «Take That» Якщо у нас є завдання класифікації тексту або аналізу настроїв, тоді нам слід видалити стоп-слова, оскільки вони не надають жодної інформації для нашої моделі, тобто не допускати небажаних слів у наш корпус, але якщо у нас є завдання мовного перекладу, то стоп-слова є корисні, оскільки їх потрібно перекладати разом з іншими словами.

Немає жорсткого правила щодо того, коли прибирати стоп-слова. Але краще видалити стоп-слова, якщо наше завдання стосується класифікації мови, фільтрації спаму, генерації субтитрів, автоматичної генерації тегів, аналізу настрою чи щось, що пов’язано з класифікацією тексту.

З іншого боку, якщо нашим завданням є машинний переклад, проблеми з відповідями на запитання, узагальнення тексту, моделювання мови, краще не видаляти стоп-слова, оскільки вони є важливою частиною цих програм.

Видалити стоп-слова за допомогою бібліотек Python досить легко, і це можна зробити різними способами.

Набір інструментів природної мови, або частіше NLTK, – це набір бібліотек і програм для символічної та статистичної обробки природної мови для англійської мови, написаної мовою програмування Python. Він містить бібліотеки для обробки тексту для токенізації, синтаксичного аналізу, класифікації, формування основи, тегування та семантичного обґрунтування.

3.5 Вилучення ключових слів

Вилучення ключових слів є одним із найважливіших завдань обробки природної мови, і воно відповідає за визначення різних методів вилучення значної кількості слів і фраз із колекції текстів. Усе це робиться для узагальнення та допомоги у відповідній та добре організованій організації, зберіганні, пошуку та відновленні вмісту.

Існує безліч доступних алгоритмів вилучення ключових слів, кожен з яких використовує унікальний набір фундаментальних і теоретичних методів для вирішення цього типу проблем.

Існують різні типи алгоритмів НЛП, деякі з яких витягують лише слова, а інші витягують і слова, і фрази. Існують також алгоритми НЛП, які виділяють ключові слова на основі повного змісту текстів, а також алгоритми, які виділяють ключові слова на основі всього змісту текстів.

Нижче наведено деякі з найвідоміших алгоритмів вилучення ключових слів:

- TextRank: цей алгоритм працює за тією ж ідеєю, що й PageRank. Google використовує цей метод для оцінки важливості різних веб-сайтів в Інтернеті;

- частота термінів – зворотна частота документів (TF-IDF): повна версія TF-IDF – частота термінів – зворотна частота документів, яка намагається краще визначити важливість терміна в документі. Крім того, враховуйте зв’язки між текстами з одного корпусу;
- RAKE: RAKE означає швидке автоматичне вилучення ключових слів і є різновидом методу НЛП. Це може витягти ключові слова та ключові фрази з вмісту одного документа, не беручи до уваги інші документи в тій же колекції.

3.6 Регулярні вирази

Регулярний вираз (regexp, regex) – це послідовність символів, що визначає шаблон пошуку. Наприклад:

- . – будь-який символ, окрім перекладу рядка;
- \w – один символ;
- d – одна цифра;
- \s – одна прогалина;
- \W – один НЕсимвол;
- \D – одна НЕцифра;
- \S – один НЕпробіл;
- [abc] – знаходить будь-який із зазначених символів match any of a, b, or c;
- [^abc] – знаходить будь-який символ, крім зазначених;
- [a-g] – знаходить символ у проміжку від a до g.

Регулярні вирази використовують зворотний слеш (\) для позначення спеціальних форм або для дозволу використання спецсимволів. Це суперечить використанню зворотного слешу в Python: наприклад, щоб буквально позначити зворотний слеш, необхідно написати '\\\' як шаблон для пошуку, тому що регулярне вираз має виглядати як \\, де кожен зворотний слеш повинен бути екранований.

Рішення – використовувати нотацію raw string для шаблонів пошуку; зворотні слеші не будуть особливим чином оброблятися, якщо використані з префіксом r.

Таким чином, r”\n” – це рядок з двома символами (“\” та “n”), а “\n” – рядок з одним символом (переклад рядка).

Ми можемо використовувати налаштування для додаткового фільтрування нашого тексту. Наприклад, можна усунути всі символи, які не є словами. У багатьох випадках пунктуація не потрібна і її легко забрати за допомогою регулярних виразів.

Модуль re в Python представляє операції з регулярними виразами. Ми можемо використовувати функцію re.sub, щоб замінити все, що підходить під шаблон пошуку, на вказаний рядок.

Переваги:

- це невід’ємна частина методів попередньої обробки даних;
- функціональність цих методів полягає у виконанні операцій пошуку, форматування, заміни в тексті.

Недоліки:

- послідовність інформації не запам'ятовується.

3.7 Мішок слів

Алгоритми машинного навчання не можуть працювати безпосередньо з сирим текстом, тому необхідно конвертувати текст у набори цифр (вектори). Це називається вилученням ознак.

Мішок слів – це популярна та проста техніка вилучення ознак, що використовується під час роботи з текстом. Вона визначає входження кожного слова до тексту.

Щоб використати модель, нам потрібно:

- визначити словник відомих слів (токенів);
- вибрати рівень присутності відомих слів.

Будь-яка інформація про порядок чи структуру слів ігнорується. Ось чому це називається мішком слів. Ця модель намагається зрозуміти, чи зустрічається знайоме слово у документі, але не знає, де саме воно зустрічається.

Інтуїція нагадує, що подібні документи мають схожий вміст. Також завдяки вмісту ми можемо дізнатися дещо про зміст документа.

Складність цієї моделі полягає в тому, як визначити словник і як підрахувати входження слів.

Коли розмір словника збільшується, вектор документа також зростає. У прикладі вище довжина вектора дорівнює кількості відомих слів.

У деяких випадках, у нас може бути великий обсяг даних і тоді вектор може складатися з тисяч або мільйонів елементів. Більше того, кожен документ може містити лише малу частину слів зі словника.

Як наслідок, у векторному поданні буде багато нулів. Вектори з великою кількістю нулів називаються розрідженими векторами (sparse vectors), вони вимагають більше пам'яті та обчислювальних ресурсів.

Однак ми можемо зменшити кількість відомих слів, коли використовуємо цю модель, щоб зменшити вимоги до обчислювальних ресурсів. Для цього можна використовувати ту саму техніку, що ми вже розглядали до створення мішка слів:

- ігнорування регістру слів;
- ігнорування пунктуації;
- викидання стоп-слів;
- приведення слів до їх базових форм (лематизація та стемінг);
- виправлення неправильно написаних слів.

Інший, складніший спосіб створення словника – використовувати згруповані слова. Це змінить розмір словника і дасть мішку слів більше деталей документа. Такий підхід називається "N-грама".

N-грама це послідовність будь-яких сутностей (слів, букв, чисел, цифр тощо). У контексті мовних корпусів під N-грамою зазвичай розуміють послідовність слів. Юніграма це одне слово, біграма це послідовність двох слів, триграма – три слова тощо. Цифра N означає, скільки згрупованих слів входить у N-граму. У модель потрапляють в повному обсязі можливі N-грами, лише ті, що фігурують у корпусі.

Коли створено словник, слід оцінити наявність слів. Ми вже розглядали простий, бінарний підхід (1 є слово, 0 немає слова).

Є й інші методи:

- кількість – підраховується скільки разів кожне слово зустрічається в документі;
- частотність – підраховується, як часто кожне слово зустрічається в тексті (стосовно загальної кількості слів).

Переваги:

- цей процес підраховує частоту токенів;
- реалізація дуже проста;
- застосунки для класифікації та виділення ознак можуть базуватися на цій техніці.

Недоліки:

- кількість токенів у пакеті збільшується зі збільшенням довжини даних.
- розрідженість матриці також зростатиме зі збільшенням розміру вхідних даних. Кількість нулів у матриці розрідженості більше, ніж ненульових чисел.
- немає зв’язку/семантичного зв’язку один з одним, оскільки текст розбитий на незалежні слова.

3.8 TF-IDF

У частотного скорингу є проблема: слова з найбільшою частотністю мають відповідно найбільшу оцінку. У цих словах може бути не так багато інформаційного виграшу для моделі, як у менш частих словах. Один із способів виправити ситуацію – знижувати оцінку слова, що нерідко зустрічається у всіх подібних документах. Це називається TF-IDF.

TF-IDF (скорочення від term frequency - inverse document frequency) – це статистичний захід для оцінки важливості слова в документі, який є частиною колекції або корпусу.

Скоринг по TF-IDF зростає пропорційно до частоти появи слова в документі, але це компенсується кількістю документів, що містять це слово.

Формула скорингу для слова X у документі Y:

$$W_{x,y} = tf_{x,y} * \log\left(\frac{N}{df_x}\right), \quad (3.1)$$

де:

- $tf_{x,y}$ – частота входження слова x в документі y,
- df_x – кількість документів які включають слово x,
- N – загальна кількість документів.

TF (term frequency – частота слова) – відношення числа входжень слова до загального числа слів документа:

$$TF(\text{слово}) = \frac{\text{Кількість входжень слова слова в документі}}{\text{Загальна кількість елементів в документі}}. \quad (3.2)$$

IDF (inverse document frequency – зворотна частота документа) – інверсія частоти, з якою деяке слово зустрічається в документах колекції:

$$IDF(\text{слово}) = \log\left(\frac{\text{Загальна кількість документів}}{\text{Кількість документів,що включають слово}}\right). \quad (3.3)$$

У результаті обчислити TF-IDF для слова можна так:

$$TFIDF(\text{слово}) = TF(\text{слово}) * IDF(\text{слово}). \quad (3.4)$$

Переваги TF-IDF:

- це трохи перевершує семантичну інформацію між лексемами.

Недоліки TF-IDF:

- такий спосіб дає можливість моделі перефітуватися;
- не стільки семантичний зв'язок між лексемами.

Висновки до розділу

У цьому розділі було розібрано основи NLP для тексту, а саме:

- NLP дозволяє застосовувати алгоритми машинного навчання для тексту та мовлення;
- NLTK (Natural Language Toolkit) – провідна платформа для створення програм NLP на Python;
- токенізація за пропозиціями – це процес поділу письмової мови на пропозиції-компоненти;

- токенізація за словами – це процес поділу речень на слова-компоненти;
- лематизація і стемінг мають на меті привести всі зустрічаються словоформи до однієї, нормальної словникової форми;
- стоп-слова – це слова, які викидаються із тексту до/після обробки тексту;
- регулярне вираз (регулярка, regexp, regex) – це послідовність символів, що визначає шаблон пошуку;
- мішок слів – це популярна і проста техніка вилучення ознак, що використовується під час роботи з текстом. Вона визначає входження кожного слова до тексту.

4 ОГЛЯД МОДЕЛЕЙ ДЛЯ ВИРІШЕННЯ ЗАДАЧІ NLI

4.1 Визначення задачі NLI

Natural Language Inference, також відома як Recognizing Textual Entailment (RTE), – це задача визначення того, чи дана «гіпотеза» та «передумова» логічно слідують (наслідок) чи не слідують (протиріччя), чи є невизначеними (нейтральними) одне до одного. Наприклад, розглянемо гіпотезу: коти не вміють літати. Тепер беремо твердження: мій кіт вчора перелетів через будинок. Завдання моделі NLI полягає в тому, щоб передбачити, чи є одне речення відносно іншого наслідоком, протиріччям або нейтральним. У цьому випадку йдеться про протиріччя.

Основна відмінність між NLP і NLI полягає в тому, що NLP є ширшим спектром задач, який містить дві підмножини: розуміння природної мови (NLU) і генерацію природної мови (NLG). Ми більше зацікавлені в NLU, оскільки NLI підпадає саме під цю задачу. NLU в основному робить комп’ютер здатним розуміти, що означає конкретний текстовий блок. NLI, який відноситься до NLU, полягає в тому, щоб зрозуміти два наведені твердження та класифікувати їх як наслідок, протиріччя або нейтральні речення. Під час роботи з даними більшість завдань NLP включають етапи попередньої обробки, як-от видалення стоп-слів, спеціальних символів, тощо. Але у випадку NLI потрібно просто надати моделі два речення. Потім модель сама обробляє дані та виводить зв’язок між двома реченнями.

4.2 Датасет SNLI

Набір даних SNLI базується на підписах зображень із датасету Flickr30k, де підписи зображень використовуються як передумови. Гіпотези були створені вручну працівниками Mechanical Turk відповідно до такої інструкції:

- наслідок: напишіть один альтернативний підпис, який точно описує фотографію;
- нейтральний: напишіть один альтернативний підпис, який може бути точним описом фотографії;

- протиріччя: напишіть один альтернативний підпис, який є неправильним описом фотографії.

SNLI має два істотних недоліки:

- передумова обмежуються короткими описами фотографій і, таким чином, не містять часових міркувань, вірувань чи модальностей;
- прості та короткі твердження вимагають простих та коротких гіпотез, тому еталонний тест недостатньо складний, коли моделі можуть легко досягти рівнів точності людини.

4.3 Датасет MultiNLI

MultiNLI (MNLI) моделюється за схемою набору даних SNLI, але охоплює діапазон усного та письмового тексту. Таким чином, MNLI можна використовувати в поєднанні з SNLI, а також він пропонує текст з десяти різних жанрів.

Передумови MNLI походять із десяти джерел або жанрів (на основі Відкритого Американського Національного Корпусу):

- віч-на-віч: транскрипції розмов двох осіб зі збірки оповідань і розмов Шарлотти;
- уряд: звіти, промови, листи та прес-релізи з публічних державних веб-сайтів;
- листи: листи від Індіанського центру міжкультурної комунікації дискурсу благодійного збору коштів;
- 9/11: публічний звіт Національної комісії з терористичних атак на Сполучені Штати;
- OUP: п’ять нон-фікшн творів про текстильну промисловість і розвиток дитини, опублікованих Oxford University Press;
- slate: статті про популярну культуру з архівів Slate Magazine;
- телефон: транскрипції двосторонніх телефонних розмов Консорціуму лінгвістичних даних Університету Пенсільванії Switchboard;
- подорожі: туристичні путівники видавництва Berlitz Publishing;
- Verbatim: пости про лінгвістику для нефахівців з архіву Verbatim;

- фантастика: кілька вільнодоступних творів фантастики, написаних між 1912 і 2010 роками.

Процес створення гіпотези виглядає наступним чином: співпрацівнику надають передумову і просять скласти три нових речення:

- наслідок: те, що обов'язково є істинним або доречним, якщо передумова істинна;
- протиріччя: таке, яке обов’язково є хибним або недоречним, коли передумова істинна;
- нейтральний: і такий, де не застосовується жодна умова.

Важливою особливістю набору даних є те, що лише п’ять із десяти жанрів з’являються в навчальному наборі, що робить інші п’ять жанрів невидимими для моделі. Ці невидимі жанри можна використовувати для оцінки того, наскільки добре модель може узагальнити невидимі джерела тексту.

4.4 Датасет SuperGLUE

SuperGLUE – це набір із десяти тестів, які вимірюють ефективність моделі NLP у трьох завданнях:

- natural language inference;
- question answering;
- coreference resolution.

У SuperGLUE є два тести NLI: RTE та CB. RTE, або Recognizing Textual Entailment, містить набори даних RTE1, RTE2, RTE3 і RTE5. Самі дані взяті з Вікіпедії та новинних статей. Відмінною рисою RTE є те, що він розрахований на класифікацію за двома класами замість класифікації за трьома класами, тому не має нейтральної позначки.

CB, або CommitmentBank, – це збірка коротких текстів, у яких принаймні одне речення є складним. Кожна з його частин є анотованою із зазначенням ступеня, до якого особа, що написала текст, впевнена в правдивості твердження. Кожен приклад

складається з передумови, що містить вбудоване речення, а відповідна гіпотеза є вилученням з цього речення.

4.5 Датасет FEVER

Цей набір даних відрізняється від SNLI та MNLI, оскільки він підтримує як NLI, так і перевірку фактів. Замість передумови, набір даних надає URL-адресу сторінки Вікіпедії, з якої можна отримати передумову. Набір даних також надає кількість вихідних речень на сторінках для спрощення використання в NLI.

Твердження були створені людиною, вручну перевірені на відповідність вступним розділам сторінок Вікіпедії та позначені як «Підтримується», «Спростовано» або «Недостатньо інформації».

Твердження були згенеровані шляхом перефразування фактів з Вікіпедії та зміни їх різними способами, деякі з яких змінювали значення. Для кожного твердження, та без знання того, звідки воно була створене, анотатори вибирали докази у формі речень із Вікіпедії, щоб виправдати маркування твердження.

4.6 Датасет WIKI-FACTCHECK

На відміну від SNLI, MNLI та FEVER, цей набір даних складається з реальних тверджень, отриманих із цитат у Вікіпедії. Передумови або доказові документи були документами, на які посилалися твердження. Крім того, набір даних надає контекст для кожного твердження, що може допомогти в неоднозначних випадках. Акцент на твердженнях реального світу справді ставить перед NLI більш складне завдання.

Недоліком цього набору даних є якість: твердження та передумови були автоматично витягнуті з Вікіпедії та іноді мали мало сенсу.

4.7 ANLI

ANLI є найдосконалішим тестом NLI на сьогодні. Процес збору в ANLI дуже відрізняється від інших наборів даних, оскільки він використовує техніку під назвою «Навчання людини та моделі в циклі» (HAMLET).

HAMLET організовує процес збору даних у раунди, де кожен раунд має такі кроки:

а) модель SOTA навчена;

б) людині-анотатору надається контекст (передумова) і бажана цільова мітка, і його просять створити гіпотезу, яка змусить модель неправильно класифікувати мітку;

в) якщо модель неправильно класифікує приклад, він демонструється двом перевіряльникам, щоб переконатися, що він правильний. Якщо вони не погоджуються, третя людина верифікатор розриває нічию;

г) приклад додається до навчального набору цього раунду та використовуватиметься для навчання моделі для наступного раунду.

З кожним раундом складність моделі та набору даних зростала:

- раунд 1. Модель: BERT-Large. Набір даних: SNLI + MNLI. Контексти: Wikipedia + HotpotQA;
- раунд 2. Модель: ансамбль моделей RoBERTa. Набір даних: SNLI+ MNLI + FEVER + дані раунду 1. Контексти: нові з Вікіпедії + HotpotQA;
- раунд 3. Модель: ансамбль моделей RoBERTa. Набір даних: SNLI + MNLI + FEVER + дані 1 раунду + дані 2 раунду. Контексти: різноманітні джерела, включаючи розмовні тексти та довші контексти.

У кожному раунді використовувана модель NLI була сильнішою, а контексти довшими та важчими для розуміння. Таким чином, анотатори-супротивники повинні були придумати більше прикладів кожного раунду, перш ніж їм вдасться обдурити модель. Наприклад, у першому раунді анотатори робили в середньому 3.4 спроби, перш ніж модель була обдурена, тоді як у раунді 3 їм потрібно було в середньому 6.4 спроби.

Набір даних ANLI збирається складніше, ніж інші, і надає довші контексти реального світу. Крім того, ANLI забезпечує механізм розширення шляхом додавання нових раундів, щоб еталон міг розвиватися разом із моделями SOTA.

4.8 Модель BERT

BERT (Bidirectional Encoder Representation Transformers) – був опублікований дослідниками Google AI у 2018 році. Він продемонстрував найсучасніші результати в багатьох завданнях NLP, таких як запитання та відповіді, завдання NLI, тощо. Це, по суті, стек кодерів трансформаторної архітектури. Він має дві версії BERT base і BERT large. BERT base має 12 шарів у стеці кодера та 110 млн параметрів, тоді як BERT large має 24 шари та 340 млн параметрів.

Попереднє навчання BERT складається з двох завдань:

- модель маскової мови (Masked Language Model) – у вхідній послідовності, надісланій моделі як вхідні дані, випадкові 15% слів маскуються, і моделі доручається передбачити ці маски, розуміючи контекст незамаскованих слів наприкінці навчання. Це допомагає моделі зрозуміти контекст речення;
- передбачення наступного речення (Next Sentence Prediction) – на вхідні дані моделі подаються пари з двох речень. У цьому завданні наприкінці навчання модель повинна передбачити, чи слідують речення одне за одним або не слідують. Це допомагає зрозуміти зв’язок між двома реченнями, що є основною метою таких завдань, як запитання та відповідь, NLI тощо.

Вхідними даними для моделі BERT є послідовність токенів, які перетворюються на вбудовування. Кожне вбудовування токена є комбінацією 3 вбудовувань. Детальніше пояснення приведено на рисунку 4.1.

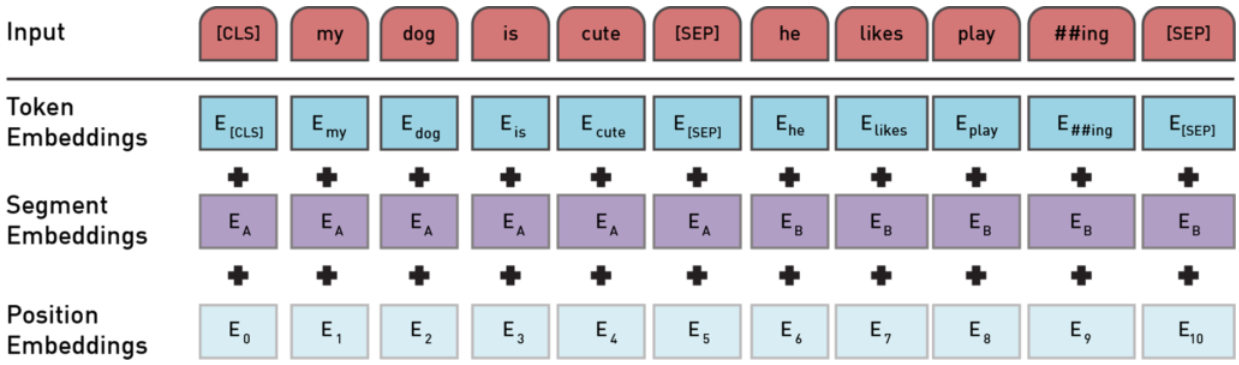

Рисунок 4.1 – Обробка вхідних даних для моделі BERT

Вбудовування токенів (Token Embeddings) – це вбудовування слів зі словника токенів WordPiece.

Вбудовування сегментів (Segment Embeddings) – ці вбудовування використовуються так як модель BERT приймає пару речень як вхідні дані, щоб допомогти моделі відрізнити вбудовування від різних речень. На зображенні вище EA представляє вкладення речення A, тоді як EB представляє вкладення з речення B.

Вбудовування позиції (Position Embeddings) – щоб зафіксувати інформацію про «послідовність» або «порядок», ці вбудовування використовуються для вираження позиції слів у реченні.

Вихід моделі BERT має такі ж номери маркерів, як у вхідних даних із додатковим маркером класифікації, який дає результати класифікації. Тобто показує чи йде речення Б після речення А, чи ні.

4.9 Модель DeBERTa

Це архітектура на основі Transformer і поточна модель SOTA для більшості завдань SuperGLUE: NLI (RTE, CommitmentBank), Common Sense Reasoning (ReCoRD), Question Answering (COPA, MultiRC).

DeBERTa:

- представляє механізм розмежування уваги, де кожне слово представлено за допомогою двох векторів, які кодують його зміст і позицію. Вагові

коефіцієнти уваги серед слів обчислюються за допомогою розділених матриць їхнього вмісту та відносних позицій;

- покращений декодер маски використовується для включення абсолютних позицій у рівень декодування для прогнозування замаскованих токенів під час попереднього навчання моделі;
- новий віртуальний змагальний метод навчання використовується для тонкого налаштування, щоб покращити узагальнення моделей.

4.10 Модель RoBERTa

Надійно оптимізований підхід попереднього навчання BERT (RoBERTa) був запропонований дослідниками Facebook. Вони виявили, що набагато більш надійно натренована модель BERT може краще виконувати завдання GLUE. Модель RoBERTa – це модель BERT із модифікованим підходом до навчання.

Нижче наведено кілька змін, внесених у модель RoBERTa порівняно з моделлю BERT.

а) дані – модель RoBERTa навчається з використанням набагато більшої кількості даних порівняно з BERT. Вона навчається на 160 ГБ нестиснених даних;

б) статичне та динамічне маскування – у моделі BERT дані були замасковані лише один раз під час попередньої обробки, що призводить до поодиноких статичних масок. Ці маски використовуються для всіх ітерацій під час навчання. Навпаки, дані, використані для навчання RoBERTa, дублювалися 10 разів з 10 різними шаблонами масок і навчалися протягом 40 епох. Це означає, що один шаблон маски використовується лише в 4 епохах. Це статичне маскування. Тоді як у динамічному маскуванні для кожної епохи під час навчання генерується різний шаблон маски;

в) видалення мети передбачення наступного речення (NSP) – дослідження виявили, що видалення втрати NSP значно покращило продуктивність моделі для завдань GLUE;

г) навчання на великих батчах – навчальна модель на великих батчах покращила точність моделі;
д) токенізація – RoBERTa використовує схему кодування Byte-Pair Encoding (BPE) на рівні байтів із словниковим запасом 50 тис., на відміну від BPE, на рівні символів BERT із словниковим запасом 30 тис.

Висновки до розділу

В третьому розділі було проаналізовано задачі аналізу природньої мови, моделі та підходи для їх рішення. Спираючись на переваги й недоліки проаналізованих моделей, для розробки свого рішення було обрано переднавчену модель RoBERTa, та розширений варіант датасету FEVER.

5 ВИБІР ТЕХНОЛОГІЙ РОЗРОБКИ

5.1 Мова програмування Python

Python – це мова програмування, яка підтримує створення широкого спектру програм. Розробники вважають його чудовим вибором для проектів штучного інтелекту (AI), машинного та глибокого навчання. Мова Python постачається з багатьма бібліотеками та фреймворками, які спрощують кодування. Це також значно економить час.

Найпопулярнішими бібліотеками є NumPy, яка використовується для наукових розрахунків; SciPy для складніших обчислень; і scikit для вивчення інтелектуального аналізу даних. Ці бібліотеки працюють разом із потужними фреймворками, такими як TensorFlow, CNTK і Apache Spark. Ці бібліотеки та фреймворки необхідні, коли мова йде про проекти машинного та глибокого навчання.

Код Python є лаконічним і читабельним навіть для нових розробників, що корисно для проектів машинного та глибокого навчання. Завдяки простому синтаксису розробка додатків на Python є швидкою порівняно з багатьма мовами програмування. Крім того, це дозволяє розробнику тестувати алгоритми без їх впровадження. Читальний код також життєво важливий для спільного кодування. Багато людей можуть працювати разом над складним проектом.

Python є мовою програмування з відкритим вихідним кодом і користується відмінною підтримкою багатьох ресурсів і якісної документації по всьому світу. Він також має велику та активну спільноту розробників, які надають допомогу на будь-якому етапі розробки. Python має легкий для розуміння та зрозумілий синтаксис. Крім того, численні фреймворки та бібліотеки прискорюють розробку програмного забезпечення. Використовуючи готові рішення, багато чого можна зробити за допомогою декількох рядків коду. Python хороший для розробки прототипів, що підвищує продуктивність.

Проекти Python можна інтегрувати з іншими системами, закодованими різними мовами програмування. Це означає, що його набагато легше поєднувати з іншими проектами AI, написаними іншими мовами. Крім того, оскільки Python є

розширюваним і портативним, його можна використовувати для виконання міжмовних завдань. Адаптивність Python полегшує науковцям і розробникам даних навчати моделі машинного навчання.

Python надає багато інструментів для перевірки коду та тестування. Розробники можуть швидко перевірити правильність і якість коду. Проекти штучного інтелекту, як правило, займають багато часу, тому потрібне добре структуроване середовище для тестування та перевірки на помилки. Python є ідеальною мовою, оскільки він підтримує ці функції.

Python відносно повільний порівняно з іншими мовами програмування. Незважаючи на те, що швидкість не є однією з сильних сторін Python, вона забезпечує рішення, відоме як Cython. Це надмножина мови Python, призначена для досягнення такої ж продуктивності коду, як і мова C. Розробники можуть використовувати Cython для кодування розширень C так само, як вони кодують у Python, оскільки його синтаксис майже однаковий. Cython значно підвищує продуктивність мови.

Python постачається з великою різноманітністю бібліотек. Деякі з цих фреймворків пропонують хороші інструменти візуалізації. У штучному інтелекті, машинному та глибокому навчанні важливо подавати дані в зручному для читання форматі. Тому Python є ідеальним вибором для реалізації цієї функції. Деякі бібліотеки, такі як Matplotlib, дозволяють створювати діаграми, гістограми та графіки для кращого представлення даних і візуалізації. Крім того, різні API, які підтримує Python, покращують процес візуалізації.

5.2 Бібліотека PyTorch

PyTorch – це платформа машинного навчання на основі бібліотеки Torch, яка використовується для таких програм, як комп’ютерне бачення та обробка природної мова. Спочатку розроблена Meta AI, а тепер є частиною Linux Foundation. Це безкоштовне програмне забезпечення з відкритим кодом, випущене за модифікованою ліцензією BSD. Хоча інтерфейс Python є більш відшліфованим і основним напрямком розробки, PyTorch також має інтерфейс C++.

На основі PyTorch створено ряд програм для глибокого навчання, зокрема Tesla Autopilot, Pyro від Uber, Transformers від Hugging Face, PyTorch Lightning і Catalyst.

PyTorch надає дві функції високого рівня:

- тензорні обчислення (наприклад, NumPy) із потужним прискоренням за допомогою графічних процесорів (GPU);
- глибокі нейронні мережі, побудовані на основі стрічкової системи автоматичного диференціювання.

PyTorch визначає клас під назвою Tensor (torch.Tensor) для зберігання та роботи з однорідними багатовимірними прямокутними масивами чисел. Тензори PyTorch схожі на масиви NumPy, але також можуть працювати на графічному процесорі NVIDIA, що підтримує CUDA. PyTorch також розробляє підтримку для інших платформ GPU, наприклад, ROCm від AMD і Metal Framework від Apple.

PyTorch використовує метод автоматичної диференціації. Рекордер записує виконані операції, а потім відтворює їх у зворотному напрямку, щоб обчислити градієнти. Цей метод особливо потужний при побудові нейронних мереж для економії часу на одній епосі шляхом обчислення диференціювання параметрів на прямому проході.

torch.optim – модуль, який реалізує різні алгоритми оптимізації, які використовуються для побудови нейронних мереж. Більшість поширених методів уже підтримуються, тому немає необхідності створювати їх з нуля.

Автоградація PyTorch дозволяє легко визначати обчислювальні графіки та приймати градієнти, але необроблена автоградація може бути надто низькорівневою для визначення складних нейронних мереж. Тут може допомогти модуль nn. Модуль nn надає рівні та інструменти для легкого створення нейронних мереж, просто визначаючи рівні мережі.

5.3 Бібліотека faiss

Faiss – це бібліотека для ефективного пошуку подібності та кластеризації щільних векторів. Він містить алгоритми, які здійснюють пошук у наборах векторів

будь-якого розміру, аж до тих, які, можливо, не поміщаються в оперативній пам’яті. Він також містить допоміжний код для оцінки та налаштування параметрів.

Faiss написано мовою C++ із повними оболонками для Python. Деякі з найбільш корисних алгоритмів реалізовані на GPU. Він розроблений Facebook AI Research.

5.4 Бібліотека NLTK

NLTK є провідною платформою для створення програм Python для роботи з даними людської мови. Вона надає прості у використанні інтерфейси для більш ніж 50 корпусів і лексичних ресурсів, таких як WordNet, а також набір бібліотек для обробки тексту для класифікації, токенізації, сформування основи, тегування, синтаксичного аналізу та семантичного міркування, оболонки для індустріальних бібліотек NLP, і активний дискусійний форум.

Завдяки практичному посібнику, який представляє основи програмування разом із темами з комп’ютерної лінгвістики, а також вичерпну документацію API, NLTK підходить як для лінгвістів, інженерів, студентів, викладачів, дослідників, так і для промислових користувачів. NLTK доступний для Windows, Mac OS X і Linux. Найкраще те, що NLTK є безкоштовним проектом із відкритим кодом, керованим спільнотою.

NLTK називають «чудовим інструментом для навчання та роботи в комп’ютерній лінгвістиці з використанням Python» і «дивовижною бібліотекою для гри з природною мовою».

Обробка природної мови за допомогою Python надає практичний вступ до програмування для обробки мови. Написана творцями NLTK, вона веде читача через основи написання програм на Python, роботи з корпусами, категоризації тексту, аналізу лінгвістичної структури тощо. Онлайн-версію книги оновлено для Python 3 і NLTK 3.

5.5 Фреймворк FLASK

Flask – це мікровеб-фреймворк, написаний на Python. Його класифікують як мікрофреймворк, оскільки він не потребує спеціальних інструментів чи бібліотек. У ньому немає рівня абстракції бази даних, перевірки форми або будь-яких інших компонентів, де вже існуючі бібліотеки сторонніх розробників забезпечують загальні функції. Однак Flask підтримує розширення, які можуть додавати функції додатків так, ніби вони реалізовані в самому Flask. Існують розширення для об'єктно-реляційних картографів, перевірки форм, обробки завантажень, різних відкритих технологій автентифікації та кількох загальних інструментів, пов'язаних із структурою.

Мікрофреймворк Flask є частиною Pallets Projects (раніше Pocoo) і базується на кількох інших з них, усі під ліцензією BSD.

Werkzeug – це службова бібліотека для мови програмування Python для додатків інтерфейсу шлюзу веб-сервера (WSGI). Werkzeug може створювати екземпляри об'єктів для запитів, відповідей і службових функцій. Його можна використовувати як основу для спеціального програмного забезпечення та підтримує Python 2.7 і 3.5 і пізніших версій.

Jinja, також від Ronacher, є рушієм шаблонів для мови програмування Python. Подібно до веб-фреймворку Django, він обробляє шаблони в пісочниці.

MarkupSafe – це бібліотека обробки рядків для мови програмування Python. Однойменний тип MarkupSafe розширює рядковий тип Python і позначає його вміст як "безпечний"; поєднання MarkupSafe зі звичайними рядками автоматично екранує непомічені рядки, уникаючи подвійного екранування вже позначених рядків.

ItsDangerous – безпечна бібліотека серіалізації даних для мови програмування Python. Він використовується для збереження сеансу програми Flask у файлі cookie, не дозволяючи користувачам змінювати вміст сеансу.

Flask надає такі можливості:

- сервер розробки та відладчик;
- інтегрована підтримка модульного тестування;

- відправлення запитів RESTful;
- використовує шаблони Jinja;
- підтримка безпечних файлів cookie (сесії на стороні клієнта);
- 100% сумісність з WSGI 1.0;
- на основі Unicode;
- повна документація;
- сумісність з Google App Engine;
- доступні розширення функціональності.

5.6 Angular

Angular – це платформа та фреймворк для створення односторінкових клієнтських програм за допомогою HTML і TypeScript. Angular написаний на TypeScript. Він реалізує основні та додаткові функції як набір бібліотек TypeScript, які ви імпортуєте у свої програми.

Архітектура програми Angular спирається на певні фундаментальні концепції. Основними будівельними блоками фреймворку Angular є компоненти Angular, які організовані в NgModules. NgModules збирають пов’язаний код у функціональні набори; додаток Angular визначається набором NgModules. Додаток завжди має принаймні кореневий модуль, який уможливлює початкове завантаження, і зазвичай має набагато більше модулів функцій.

Компоненти визначають представлення, які є наборами елементів екрану, які Angular може вибирати та змінювати відповідно до логіки та даних вашої програми. Компоненти використовують служби, які надають певні функції, не пов’язані безпосередньо з представленнями. Постачальники послуг можна вставляти в компоненти як залежності, роблячи ваш код модульним, придатним для повторного використання та ефективним.

Модулі, компоненти та служби – це класи, які використовують декоратори. Ці декоратори позначають свій тип і надають метадані, які вказують Angular, як їх використовувати.

Метадані для класу компонента пов’язують його з шаблоном, який визначає представлення. Шаблон поєднує в собі звичайний HTML з директивами Angular і розміткою прив’язки, що дозволяє Angular змінювати HTML перед його відтворенням для відображення.

Компоненти програми зазвичай визначають багато представлень, упорядкованих ієрархічно. Angular надає службу Router, щоб допомогти вам визначити шляхи навігації між видами. Маршрутизатор надає складні можливості навігації в браузері.

5.7 NumPy

NumPy – це основний пакет для наукових обчислень на Python. Це бібліотека Python, яка надає об’єкт багатовимірного масиву, різні похідні об’єкти (такі як замасковані масиви та матриці), а також набір процедур для швидких операцій над масивами, включаючи математичні, логічні, маніпуляції формою, сортування, вибір, введення/виведення, дискретне перетворення Фур'є, базова лінійна алгебра, основні статистичні операції, випадкове моделювання та багато іншого. В основі пакета NumPy лежить об’єкт ndarray. Це інкапсулює n-вимірні масиви однорідних типів даних, причому багато операцій виконуються в скомпільованому коді для підвищення продуктивності. Існує кілька важливих відмінностей між масивами NumPy і стандартними послідовностями Python:

- масиви NumPy мають фіксований розмір під час створення, на відміну від списків Python (які можуть динамічно зростати). Зміна розміру масиву створить новий масив і видалить вихідний;
- усі елементи в масиві NumPy мають мати однаковий тип даних і, таким чином, матиме однаковий розмір у пам’яті. Виняток: можна мати масиви (Python, включаючи NumPy) об’єктів, що дозволяє створювати масиви елементів різного розміру;
- масиви NumPy сприяють розширеним математичним та іншим типам операцій над великою кількістю даних. Як правило, такі операції

виконуються ефективніше та з меншою кількістю коду, ніж це можливо за допомогою вбудованих послідовностей Python;

- зростаюча кількість наукових і математичних пакетів на основі Python використовує масиви NumPy; хоча вони зазвичай підтримують введення послідовності Python, вони перетворюють такі введення в масиви NumPy перед обробкою, і вони часто виводять масиви NumPy. Іншими словами, щоб ефективно використовувати більшу частину (можливо, навіть більшість) сьогоднішнього наукового/математичного програмного забезпечення на основі Python, просто знати, як використовувати вбудовані типи послідовностей Python, недостатньо – потрібно також знати, як використовувати масиви NumPy.

5.8 SentenceTransformers

SentenceTransformers – це платформа Python для найсучаснішого вбудовування речень, тексту та зображень.

Можна використовувати цю структуру для обчислення вбудованих речень/тексту для більш ніж 100 мов. Потім ці вкладення можна порівняти, наприклад, з косинус-подібністю, щоб знайти речення з подібним значенням. Це може бути корисним для семантичного подібного тексту, семантичного пошуку або аналізу перефразів.

Фреймворк заснований на PyTorch і Transformers і пропонує велику колекцію попередньо навчених моделей, налаштованих для різних завдань. Крім того, можна легко налаштувати власні моделі.

5.9 spaCy

spaCy – це бібліотека програмного забезпечення з відкритим вихідним кодом для розширеної обробки природної мови, написана на мовах програмування Python і Cython. Бібліотека видається за ліцензією Массачусетського технологічного

інституту, а її головними розробниками є Метью Гоннібал та Інес Монтані, засновники програмної компанії Explosion.

На відміну від NLTK, який широко використовується для навчання та досліджень, spaCy зосереджується на наданні програмного забезпечення для виробничого використання. spaCy також підтримує робочі процеси глибокого навчання, які дозволяють підключати статистичні моделі, навчені популярними бібліотеками машинного навчання, як-от TensorFlow, PyTorch або MXNet, через власну бібліотеку машинного навчання Thinc. Використовуючи Thinc як серверну частину, spaCy містить моделі згорткових нейронних мереж для тегування частин мови , розбір залежностей, категоризація тексту та розпізнавання іменованих об’єктів (NER).

Попередньо створені моделі статистичних нейронних мереж для виконання цих завдань доступні для 23 мов, включаючи англійську, португальську, іспанську та китайську, а також є багатомовна модель NER. Додаткова підтримка токенізації для більш ніж 65 мов дозволяє користувачам також навчати власні моделі на власних наборах даних.

Основні можливості:

- неруйнівна токенізація;
- підтримка "альфа-токенізації" для понад 65 мов;
- вбудована підтримка компонентів конвеєра, які можна навчати, таких як розпізнавання іменованих об’єктів, додавання тегів до частини мови, розбір залежностей, класифікація тексту, зв’язування об’єктів тощо;
- статистичні моделі для 19 мов;
- багатозадачне навчання за допомогою попередньо підготовлених трансформаторів, таких як BERT;
- підтримка спеціальних моделей у PyTorch, TensorFlow та інших фреймворках
- найсучасніша швидкість і точність;
- готова до виробництва система навчання;
- вбудовані візуалізатори для синтаксису та іменованих сутностей;
- просте пакування моделі, розгортання та керування робочим процесом.

spaCy постачається з декількома розширеннями та візуалізаціями, які доступні як безкоштовні бібліотеки з відкритим кодом:

- thin: бібліотека машинного навчання, оптимізована для використання ЦП і глибокого навчання з введенням тексту;
- sense2vec: бібліотека для обчислення подібності слів на основі Word2vec;
- displaCy: візуалізатор дерева аналізу залежностей із відкритим кодом, створений за допомогою JavaScript, CSS і SVG;
- displaCyENT: візуалізатор іменованих сутностей із відкритим кодом, створений за допомогою JavaScript і CSS.

5.10 Scikit-learn

Scikit-learn (раніше scikits.learn і також відомий як sklearn) – це безкоштовна бібліотека машинного навчання для мови програмування Python. Він містить різні алгоритми класифікації, регресії та кластеризації, включаючи машини опорних векторів, випадкові ліси, посилення градієнта, k-середні та DBSCAN, і розроблений для взаємодії з чисельними та науковими бібліотеками Python NumPy та SciPy. Scikit-learn – це проект, який фінансується NumFOCUS.

Проект scikit-learn розпочався як scikits.learn, проект Google Summer of Code французького дослідника даних Девіда Курнапо. Його назва походить від уявлення про те, що це «SciKit» (SciPy Toolkit), окремо розроблене та розповсюджене стороннє розширення для SciPy. Оригінальна кодова база пізніше була переписана іншими розробниками. У 2010 році Фабіан Педрегоза, Гаель Варокво, Александр Грамфор і Вінсент Мішель, усі з Французького інституту досліджень комп'ютерних наук і автоматизації в Сакле, Франція, взяли на себе керівництво проектом і зробили перший публічний випуск 1 лютого 2010 року.

Різні scikits, scikit-learn, а також scikit-image були описані як «добре підтримувані та популярні» в листопаді 2012 року. Scikit-learn є однією з найпопулярніших бібліотек машинного навчання на GitHub.

Scikit-learn здебільшого написаний на Python і широко використовує NumPy для високопродуктивної лінійної алгебри та операцій з масивами. Крім того, деякі основні алгоритми написані на Cython для підвищення продуктивності. Машини опорних векторів реалізовані обгорткою Cython навколо LIBSVM; логістичної регресії та лінійних опорних векторних машин за допомогою подібної оболонки навколо LIBLINEAR. У таких випадках розширення цих методів за допомогою Python може бути неможливим.

Scikit-learn добре інтегрується з багатьма іншими бібліотеками Python, такими як Matplotlib і plotly для побудови графіків, NumPy для векторизації масивів, фрейми даних Pandas, SciPy та багато інших.

5.11 Git

Git – розподілена система керування версіями. Проект був створений Лінусом Торвальдсом для управління розробкою ядра Linux, першу версію випущено 7 квітня 2005 року. На сьогоднішній день його підтримує Джуніо Хамано.

Система спроектована як набір програм, спеціально розроблених з урахуванням їх використання у сценаріях. Це дозволяє зручно створювати спеціалізовані системи контролю версій на базі Git або інтерфейси користувача. Наприклад, Cogito є саме таким прикладом оболонки до репозиторій Git, а StGit використовує Git для керування колекцією виправлень (патчів).

Git підтримує швидкий поділ та злиття версій, включає інструменти для візуалізації та навігації з нелінійної історії розробки. Як і Darcs, BitKeeper, Mercurial, Bazaar та Monotone, Git надає кожному розробнику локальну копію всієї історії розробки, зміни копіюються з одного репозиторію до іншого.

Віддалений доступ до репозиторій Git забезпечується git-демоном, SSH або HTTP-сервером. TCP-сервіс git-daemon входить до дистрибутиву Git і є поряд з SSH найбільш поширеним і надійним методом доступу. Метод HTTP, незважаючи на низку обмежень, дуже популярний у контрольованих мережах, тому що дозволяє використовувати існуючі конфігурації мережевих фільтрів.

Ядро Git є набір утиліт командного рядка з параметрами. Усі налаштування зберігаються у текстових конфігураційних файлах. Така реалізація робить Git, що легко портується на будь-яку платформу і дає можливість легко інтегрувати Git в інші системи (зокрема, створювати графічні git-клієнти з будь-яким бажаним інтерфейсом).

Репозиторій Git є каталог файлової системи, в якому знаходяться файли конфігурації репозиторію, файли журналів, що зберігають операції, що виконуються над репозиторієм, індекс, що описує розташування файлів, і сховище, що містить власне файли. Структура сховища файлів не відображає реальну структуру файлового дерева, що зберігається в репозиторії, вона орієнтована на підвищення швидкості виконання операцій з репозиторієм. Коли ядро обробляє команду зміни (неважливо, при локальних змінах або при отриманні патчу від іншого вузла), воно створює нові файли в сховищі, що відповідають новим станам змінених файлів. Істотно, що ніякі операції не змінюють вміст файлів, що вже існують у сховищі.

За замовчуванням репозиторій зберігається у підкаталозі під назвою «.git» у кореневому каталозі робочої копії дерева файлів, що у репозиторії. Будь-яке файлове дерево в системі можна перетворити на репозиторій git, віддавши команду створення репозиторію з кореневого каталогу цього дерева (або вказавши кореневий каталог у параметрах програми). Репозиторій може бути імпортований з іншого вузла, доступного через мережу. При імпорті нового репозиторію автоматично створюється робоча копія, що відповідає останньому зафіксованому стану імпортованого репозиторію (тобто не копіюються зміни у робочій копії вихідного вузла, для яких на тому вузлі не було виконано команду commit).

5.12 GitHub

GitHub - найбільший веб-сервіс для хостингу IT-проектів та їхньої спільної розробки. Автори сайту називають GitHub «соціальною мережею для розробників».

Окрім розміщення коду, учасники можуть спілкуватися, коментувати редагування один одного, а також стежити за новинами знайомих.

За допомогою широких можливостей Git програмісти можуть поєднувати свої репозиторії - GitHub пропонує зручний інтерфейс для цього і може відображати вклад кожного учасника у вигляді дерева. Для проектів є особисті сторінки, невеликі Вікі та система відстеження помилок.

Додаткові можливості:

- прямо на сайті можна переглянути файли проектів із підсвічуванням синтаксису для більшості мов програмування;
- можна створювати приватні репозиторії, які будуть видні лише вам та вибраним вами людям. Раніше така нагода була платною;
- є можливість прямого додавання нових файлів до свого репозиторію через веб-інтерфейс сервиса;
- код проектів можна не лише скопіювати через Git, а й завантажити у вигляді звичайних архівів із сайту;
- крім Git, сервіс підтримує отримання та редагування коду через SVN та Mercurial;
- на сайті є pastebin-сервіс gist.github.com для швидкого опублікування фрагментів коду;
- файли з репозиторію можуть автоматично публікуватись у вигляді статичного сайту за допомогою GitHub Pages.

Раніше Ruby-проекти могли бути автоматично опубліковані в RubyGems-репозиторії сервісу, але в жовтні 2009 року GitHub відмовився від цього сервісу.

У 2019 році було запущено сервіс GitHub Packages, що дозволяє публікувати прямо на GitHub пакети RubyGems, NuGet, npm, Maven, а також образи Docker.

У тому року відбувся реліз системи автоматизації GitHub Action. Крім стандартних можливостей CI/CD, таких як збирання, тестування та публікація коду, сервіс пропонує тісну інтеграцію з іншими функціями GitHub, а також дозволяє взаємодіяти зі сторонніми сервісами. Розробники можуть публікувати модулі

(actions), що перевикористовуються, реалізують часто використовувані сценарії. Сервіс надається безкоштовно для публічних репозиторіїв.

Висновки до розділу

В даному розділі було розглянуто та аргументовано вибір технологій для виконання дипломного проєкту. Було порівняно можливі архітектури, та обрано кращу. Визначено підхід до виконання, обрано найбільш підходящий шаблон. Проаналізовано та вивчено всі бібліотеки які будуть корисними для розробки програмної системи. Як спосіб зберігання проєктів обрано платформу GitHub.

6 РОЗРОБЛЕННЯ ПРОГРАМНОЇ СИСТЕМИ

6.1 Принцип роботи програмного забезпечення

Задача автоматизованої перевірки тверджень може бути сформульована наступним чином: задано текстове твердження c і корпус, $D = \{d_1, d_2, \dots, d_n\}$, де кожен уривок d складається з речень s_j, $1 \leq j \leq k$. Система поверне набір доказових речень $\hat{S} \subset \cup d_i$ та мітки $\hat{y} \in$ {імовірно істинно, ймовірно хибно, непереконливо}.

Розроблювана система перевіряє твердження на достовірність в три етапи: пошук уривку в статичному сховищі, вибір речення та класифікація наслідків, як показано на рисунку 6.1. Мітка визначається наступним чином: спочатку ми виконуємо класифікацію наслідків на множині доказових речень. Коли кількість знайдених доказових речень, які підтверджують твердження, або суперечать йому, є низькою, ми позначаємо заяву як "непереконливу". Якщо кількість доказових речень, що підтверджують твердження, перевищує кількість речень, які його спростовують, ми присвоюємо твердженню позначку "ймовірно правда". В іншому випадку ми присвоюємо позначку "ймовірно хибне".

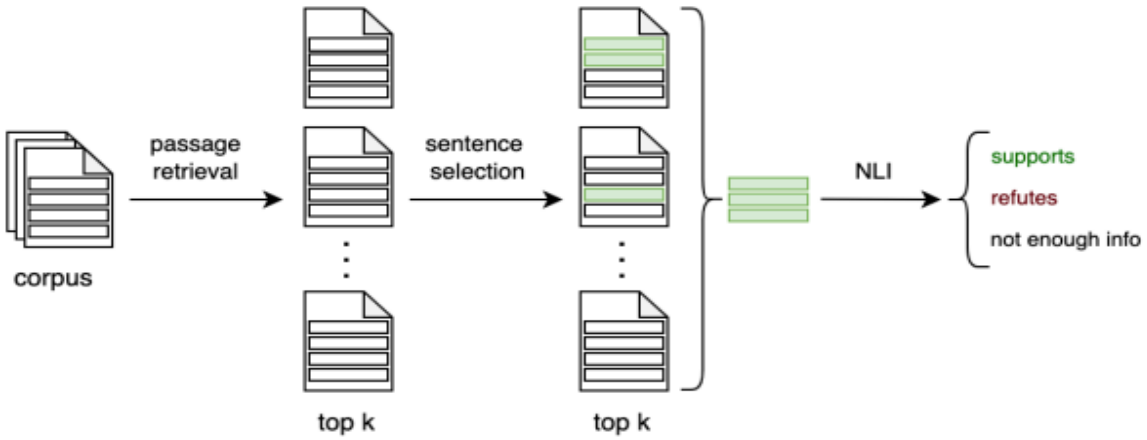

Рисунок 6.1 – Три етапи перевірки тверджень

6.2 Отримання уривків

Модель отримання уривків в даній системі базується на моделі щільного пошуку QR-BERT. Ця модель базується на BERT і створює щільні вектори для уривків шляхом обчислення середнього значення вбудовування токенів. Релевантність уривку d до формули c потім задається їх точковим добутком:

$$r(c,d) = \phi(c)^T \phi(d). \quad (6.1)$$

Точковий пошук уривків може ефективно працювати використовуючи апроксимацію індексу найближчих сусідів, реалізованого за допомогою бібліотеки FAISS. QR-BERT максимізує вибіркові втрати Функції softmax:

$$L_\theta = \sum_{(c,d) \in D_b^+} r_\theta\,(c,d) - \log(\sum_{d_i \in D_b} e^{r_\theta(c,d_i)}). \quad (6.2)$$

де D_b – множина уривків у навчальній серії b, D_b^+ множина позитивних пар тверджень-уривків у серії b, а θ – параметри BERT-моделі

У роботі використано набір даних Factual-NLI, який розширює набір даних FEVER більш різноманітними синтетичними прикладами, отриманими з наборів даних з відповідями на запитання. Існує 359 190 нових тверджень з відповідями та додатковими суперечливими твердженнями, що суперечать підходу, заснованому на правилах. Для забезпечення надійності було додано нову версію Factual-NLI під назвою Factual-NLI+1. Цей набір даних включає всі 5 мільйонів статей Вікіпедії з набору даних FEVER. Ми додаємо статті наступним чином. Для кожного твердження c у наборі даних FEVER ми отримуємо 30 найкращих веб-результатів з пошукової системи Google і зберігаємо уривки з найвищим показником BM25, які класифікуються як нейтральні за режимом спричинення наслідків. Для тверджень, згенерованих на основі запитів MSMARCO, ми включаємо нерелевантні уривки, які знайдені в наборі даних MSMARCO для цих запитів. Це створює 418 650 додаткових уривків. Новий набір даних краще відображає структуру великомасштабного корпусу, який буде використовуватися реальною системою фактчекінгу.

Розроблювана система використовує гібридну модель, яка поєднує результати з моделі щільного пошуку, описаної вище, та розрідженого пошуку BM25 для

отримання остаточного списку знайдених уривків. Для ефективного розрідженого пошуку використано повнотекстову пошукову систему Tantivy на базі Rust.

6.3 Підбір речень

Було проведено експеримент з двома методами вибору речень: методом вибору на основі вбудовування та методом вибору речень з урахуванням контексту.

Метод відбору на основі вкладень спирається на щільні представлення, отримані за допомогою моделі пошуку щільних уривків QR-BERT. Для заданого пункту формули c вибираються речення s_i з заданого уривку $d = \{s_1, s_2, \dots, s_n\}$, оцінка релевантності $r(c,s_i)$ яких більша за деякий поріг λ, який задається експериментально.

Метод виділення речень з урахуванням контексту використовує модель маркування послідовностей на основі BERT. На вхід моделі подається конкатенація токенізованого твердження $C = \{C_1, C_2, \dots, C_k\}$, спеціального [SEP] токену та токенізованого уривку доказу $E = \{E_1, E_2, \dots, E_k\}$, (рисунок 6.2). Для виводу моделі використаємо формат тегів BIO таким чином, що всі нерелевантні токени класифікуються як *O*. Перший токен доказового речення класифікується як доказ *B*, а решта токенів доказового речення – як докази *I*. Модель було навчено на основі RoBERTa-large, мінімізуючи перехресні втрати ентропії:

$$L_\theta = -\sum_{i=1}^{N}\sum_{j=1}^{l_i}\log(p_\theta(y_j^i)), \qquad (6.3)$$

де N – кількість прикладів у тренувальному пакеті, l_i кількість незаповнюючих токенів у i^{th}прикладі, і $p_\theta(y_j^i)$ – оцінка softmax ймовірності правильної мітки для j^{th} токену з i^{th}прикладу. Модель була натренована за допомогою FactualNLI з розміром батчів – 64, оптимізатором Adam та початковою швидкістю навчання 5×10^{-5} до збіжності.

6.4 Класифікація наслідків

Логічний висновок природною мовою (ЛВПМ), також відомий як класифікація текстових наслідків, є завданням виявлення того, чи випливає твердження гіпотези з уривку передумови.

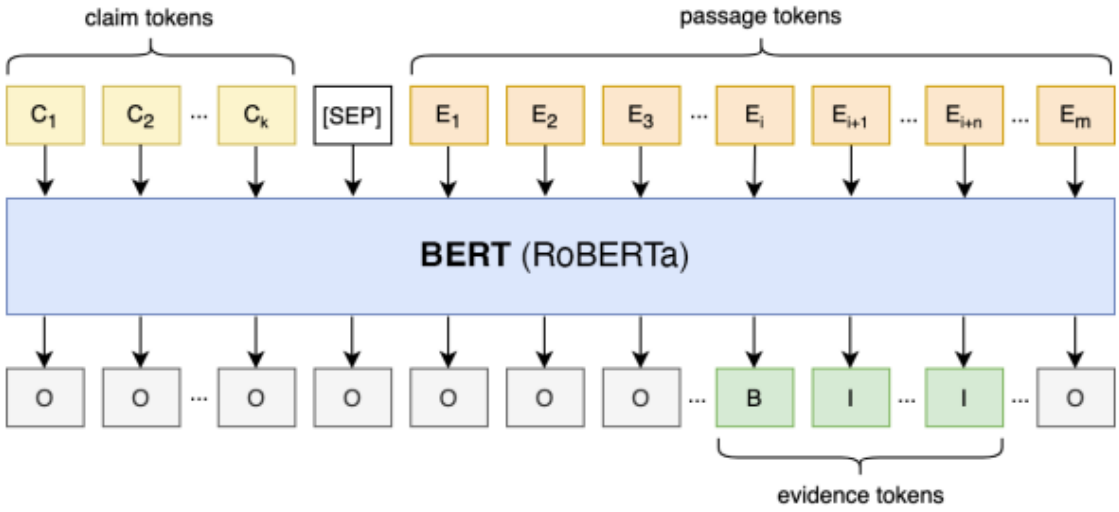

Рисунок 6.2 – Модель маркування послідовності для відбору доказів з уривку для певного твердження.

По суті, це задача класифікації тексту, де вхідними даними є: пара передумова-гіпотеза *(P, H)* та вихідна мітка $y \in$ {наслідок, протиріччя, нейтральне}. Модель NLI часто є основним компонентом багатьох автоматизованих систем перевірки фактів. Набори даних такі як Stanford Natural Language Inference corpus (SNLI), Multi-Genre Natural Language Inference corpus (Multi-NLI) та Adversarial-NLI сприяли розробці моделей для цієї задачі.

Незважаючи на те, що попередньо навчені моделі NLI, добре працюють на двох популярних наборах даних NLI (SNLI та Multi-NLI), вони не є настільки ефективними в реальних умовах. Це, можливо, пов'язано з упередженістю цих двох наборів даних, що негативно впливає на узагальнюючу здатність навчених моделей. Крім того, ці набори даних складаються з коротких односкладових припущень. Як наслідок, моделі, навчені на цих наборах даних, зазвичай не дуже добре працюють на

«зашумлених» реальних даних, що включають кілька речень. Ці проблеми призвели до розробки додаткових більш складних наборів даних, таких як Adversarial NLI. Розроблена система використовує модель NLI на основі RoBERTa-large з лінійним перетворенням вбудовуваних токенів [CLS]:

$$o = softmax(W \cdot BERT_{[CLS]}([P;H]) + a), \quad (6.4)$$

де $[P;H]$ – конкатенація передумови з гіпотезою, $W_{3\times1024}$ є лінійно трансформованою матрицею, $a_{3\times1}$ – матрицею зміщення. Модель було навчено шляхом мінімізації втрат перехресної ентропії при конкатенації трьох популярних NLI наборів даних (SNLI, Multi-NLI та Adversarial-NLI) з розміром батчу – 64, оптимізатором Adam та початковою швидкістю навчання 5×10^{-5} до збіжності.

6.5 Продуктивність розробленої системи

Оцінимо три окремі компоненти системи – пошук, відбір пропозицій та класифікація наслідків, і проведемо оцінку з використанням різних параметрів.

У таблицях 6.1-6.3 наведено значення recall@k та середній взаємний ранг (MRR@100) для моделей пошуку проходження моделей на FEVER та Factual-NLI+. Також порівняємо продуктивність на «зашумленому» розширенні набору даних FEVER, в який включені додаткові уривки з пошукової системи Google додані як «шумові» уривки. Бачимо, що при додаванні таких уривків до набору даних FEVER розрив між гібридною моделлю пошуку уривків у розробленій системі і розрідженим пошуком збільшується. Це демонструє обмеження використання розрідженого пошуку, і чому так важливо мати щільну модель пошуку для виявлення релевантних уривки із «зашумленого» тексту. В цілому, гібридна модель пошуку уривків дає найкращу продуктивність у порівнянні з BM25 та моделлю щільного пошуку.

Таблиця 6.1 – Датасет FEVER

Модель	R@5	R@10	R@20	R@100	MRR
BM25	50.52	58.92	67.93	82.93	0.381
Щільна	65.47	69.61	72.51	75.71	0.535
Гібридна	71.71	78.60	83.65	91.09	0.556

Таблиця 6.2 – Датасет FEVER з шумами

Модель	R@5	R@10	R@20	R@100	MRR
BM25	35.17	44.18	53.89	73.95	0.264
Щільна	54.10	62.13	68.09	75.24	0.405
Гібридна	54.89	64.61	73.33	86.11	0.407

Таблиця 6.3 – Датасет FACTUAL-NLI+

Модель	R@5	R@10	R@20	R@100	MRR
BM25	45.02	53.20	61.56	77.96	0.347
Щільна	59.66	67.09	72.23	78.52	0.461
Гібридна	61.29	70.03	77.51	87.90	0.465

У таблицях 6.4-6.5 показано точність на рівні токенів, відгук та оцінку F1 запропонованого методу відбору речень на основі баз даних Factual-NLI та специфічному для домену (medical) наборі даних для верифікації тверджень SciFact.

Таблиця 6.4 – Датасет Factual-NLI

Модель	Точність	Відгук	F1
Базовий підхід	67.74	91.87	77.98
Маркування послідовностей	94.78	92.11	93.43
На основі вбудовування	66.12	90.29	76.34

Таблиця 6.5 – Датасет Factual-NLI

Модель	Точність	Відгук	F1
Базовий підхід	62.21	71.54	66.55
Маркування послідовностей	69.38	68.45	68.91
На основі вбудовування	43.30	92.36	58.96

Також порівнюємо продуктивність з базовим підходом NLI на рівні речень, де ми виконуємо класифікацію наслідків. Для кожного речення та уривку відбираємо нейтральні речення як докази. Ми спостерігаємо, що модель маркування послідовності дає найвищу точність, відгук та оцінку F1 при тестуванні на Factual-NLI. Крім того, точність значно вища, ніж у інших методів.

З іншого боку, для набору даних SciFact ми бачимо, що метод маркування послідовностей залишається найкращим за точністю та показником F1, хоча його відгук нижчий, ніж у методу на основі вбудовування. Це свідчить про те, що модель маркування послідовностей здатна пом'якшити високий рівень помилкових спрацьовувань, що спостерігається при використанні методу відбору на основі вбудовування, за рахунок врахування навколишнього контексту.

Набір даних Factual-NLI+ містить твердження з уривками, які або підтримують, або спростовують твердження з деякими реченнями, позначеними як основні докази істини. Таблиці 6.6-6.7 показують ефективність моделі витягування для класифікації вхідних доказів, що підтверджують або спростовують твердження. Вхідні докази можуть бути у формі цілого уривка, речень, що підтверджують достовірність інформації, або речень, вибраних моделлю маркування послідовностей. Спостерігаємо, що модель класифікації наслідків погано працює, коли цілі уривки передаються як вхідні докази. Однак, коли на вхід подаються конкретні речення, показники точності, запам'ятовування та F1 покращуються. Причина в тому, що наша модель класифікації наслідків тренується в основному на коротких прикладах. Як наслідок, краще справляється з доказами на рівні речень порівняно з довшими уривками.

Таблиця 6.6 – Підтверджуючі докази

Вхід	Точність	Відгук	F1
Цілі уривки	63.40	52.93	52.28
Відмічені як «підтверджуючі достовірність» речення	82.15	60.05	69.38
Вибрані речення	74.40	52.68	64.34

Таблиця 6.7 – Спростовуючі докази

Вхід	Точність	Відгук	F1
Цілі уривки	33.95	40.65	37.00
Відмічені як «підтверджуючі недостовірність» речення	77.54	89.32	83.02
Вибрані речення	75.27	81.96	78.47

Було проведено наскрізну оцінку системи фактчекінгу на Factual-NLI+ з використанням різних конфігурацій пошуку *top-k* уривків (BM25, щільний, гібридний, для різних значень $k \in [5, 100]$) та підходів до відбору доказів (на основі вбудовування та маркування послідовностей). Таблиця 6.8 показує макросередній бал F1 для трьох класів (підтверджуючі, спростовуючі, нейтральні) для деяких з протестованих конфігурацій.

Таблиця 6.7 – Наскрізна перевірка за допомогою FactualNLI+ для різних конфігурацій.

Отримання уривків	Відбір речень	F1
BM25, k=5	На основі вбудовування	52.76
BM25, k=20	На основі вбудовування	47.65
BM25, k=5	Маркування послідовностей	49.65
Щільна модель, k=5	На основі вбудовування	49.03
Щільна модель, k=5	Маркування послідовностей	52.83

Продовження таблиці 6.7

Отримання уривків	Відбір речень	F1
Щільна модель, k=50	Маркування послідовностей	58.22
Гібридна модель, k=6	На основі вбудовування	50.29
Гібридна модель, k=6	Маркування послідовностей	57.24
Гібридна модель, k=50	Маркування послідовностей	52.60

Ми бачимо, що щільний або гібридний пошук з відбором доказів за допомогою запропонованої моделі маркування послідовностей дає найкращі результати. Хоча гібридний пошук призводить до дещо гіршої продуктивності, він вимагає значно меншої кількості уривків (6 замість 50) і робить систему більш ефективною.

6.6 Демонстрація роботи системи

Отже, на основі обраних моделей було створено систему для перевірки тверджень, використовуючи топ-20 результатів з веб-пошукової системи. Для заданого твердження система повертає релевантні уривки тексту з виділеними реченнями.

Уривки згруповані у два набори, що підтверджують і спростовують твердження. Система обчислює рейтинг правдивості на основі кількості підтверджуючих та спростовуючих доказів. Система повертає відповідь "ймовірно, правда", якщо більш вагомими є докази. В іншому випадку повертається "ймовірно, неправда". Коли кількість знайдених доказів є недостатньою, повертається відповідь "непереконливо".

На рисунку 6.4 показано інтерфейс системи з твердженням, яке було оцінено як ймовірно неправдиве на основі кількості доказів, що спростовують твердження (21 спростування проти 0 доказів). Система також може використовуватися на великих текстах.

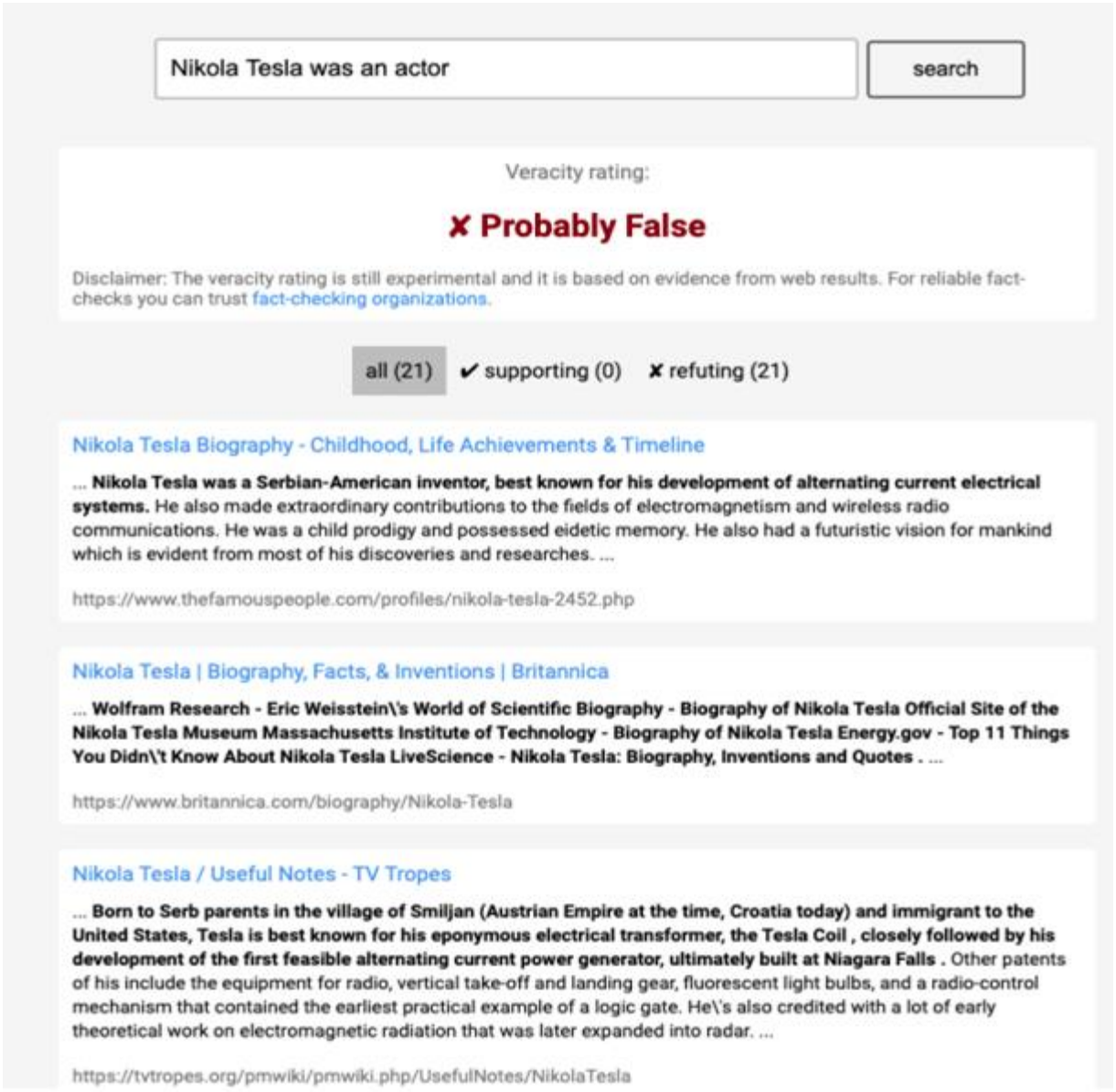

Рисунок 6.4 – Система повертає відповідні докази та оцінку правдивості твердження

Висновки до розділу

У цьому розділі було розроблено триступеневу систему фактчекінгу. Було продемонстровано, як модель щільного пошуку може призвести до більш високого рівня виявлення при пошуку уривків для фактчекінгу. Також було запропоновано дві схеми відбору релевантних речень: підхід на основі вбудовування та модель маркування послідовності для підвищення точності перевірки. Створена система дала багатообіцяючі результати, а також змогла перевіряти твердження, використовуючи результати веб-пошуку.

7 РОЗРОБЛЕННЯ СТАРТАП-ПРОЄКТУ

Стартап як форма малого ризикованого підприємництва протягом останнього десятиліття набула більш широкого розповсюдження у світі через появу Інтернету та, як наслідок, зниження бар'єрів входу на ринок. Наразі вважається однією із провідних складових інноваційної економіки, оскільки за рахунок мобільності, лабільності та великої чисельності стартап-проектів загальний об'єм інноваційних ідей зростає.

Однак, попри вище зазначене, варто відмітити, що розробка стартап-проектів та реалізація розроблених продуктів на ринок має риси підвищеної міри ризику, оскільки успіх здобуває лише невелика частка проектів (за різними дослідженнями від 10% до 20%). Провідним завданням керівника стартап-проекту на його початковому етапі розробки є перетворення ідеї проекту у бізнес-модель яка гарантовано буде ефективною, що починається із формування концепції товару для певної конкретно обраної цільової групи клієнтів при наявних ринкових умовах.

Розробка та безпосередня реалізація стартап-проекту на ринку передбачає здійснення низки кроків, межі котрих визначають ринкові перспективи проекту, його графік та принципи організації виробництва, фінансовий розгляд, дослідження ризиків і комплекс заходів з просування пропозиції для інвесторів.

Етап маркетингового аналізу проекту включає у себе:

а) розробку опису самої ідеї проекту та визначення загальних напрямів використання потенційного товару або послуги, а також їх унікальність відносно конкурентів;

б) аналіз ринкових можливостей щодо його реалізації;

в) розробку стратегії впровадження потенційного товару на ринку, засновану на основі дослідження ринкового середовища.

7.1 Опис ідеї проекту

Опис ідеї проекту подається у вигляді таблиць і містить у собі наступні пункти:

а) зміст основних ідей продукту;

б) напрямки можливостей застосування продукту;

в) основні переваги, які може отримати споживач від використання продукту;

г) чим даний продукт принципово відрізняється від вже існуючих аналогів та замінників.

Перші три аспекти подані у вигляді таблиці (таблиця 7.1) і дають комплексне уявлення про зміст ідеї та потенційні можливі ринки, в межах яких необхідно шукати групи імовірних споживачів продукту.

Таблиця 7.1 – Опис ідеї стартап-проекту

Зміст ідеї	Напрямки застосування	Вигоди для користувача
Продемонстрована система дає можливість швидко знаходити інформацію за заданим параметром з відкритих джерел мережі Інтернет та завдяки статистичному аналізу давати вірогідну оцінку правдивості заданого параметру.	1. Перевірка правдивості новинних повідомлень	Повна автоматизація процесу факт-чекінгу.
	2. Отримання статистично вірогідних відповідей на питання	Можливість перевірити будь-яку бажану інформацію
	3. Пошук малодоступної інформації	Доступність на різних платформах.

Аналіз техніко-економічних переваг, що заплановані ідеєю проекту у співставленні з пропозиціями проектів-конкурентів передбачає:

а) визначення чіткого переліку техніко-економічних властивостей та характерних рис ідеї;

б) окреслення попереднього кола конкурентів чи товарів-аналогів, що вже стабільно існують на ринку, а також проводиться збір інформації на рахунок значень техніко-економічних показників для власного проекту та проектів-конкурентів згідно з визначеним переліком;

в) проведення порівняльного аналізу показників: для власної ідеї формулюються показники, що характеризуються гіршими значеннями (W, слабкі), аналогічними (N, нейтральні) значеннями або кращими значеннями (S, сильні) (таблиця 7.2).

Таблиця 7.2 – Визначення характеристик ідеї проекту (сильних, слабких та нейтральних сторін)

№ п/п	Техніко-економічні характеристики ідеї	Товари/концепції конкурентів				W (слабка)	N(нейтральна)	S (сильна)
		Власна релізація	The Factual	Check by Meedan	Logically			
1	Простота використання	Зручний інтерфейс, без нагромадження зайвої інформації	Зручні та інтуїтивно зрозумілі користувацькі інтерфейси. Багато зайвої інформації	Досить високий поріг входження для використання користувачем	Зручний інтерфейс, але багато зайвої інформації			+

Продовження таблиці 7.2

№ п/п	Техніко-економічні характеристики ідеї	Товари/концепції конкурентів				W (слабка)	N(нейтральна)	S (сильна)
		Власна релізація	The Factual	Check by Meedan	Logically			
2	Багатофункціональність	Функціонал обмежується базовою перевіркою інформації різного роду	Неможливість аналізу конкретної інформації, тільки надані новини	Має великий та гнучкий набір інструментів для побудови потоків аналізу текстової інформації з налаштуванням під різні параметри та критерії	Неможливість аналізу конкретної інформації, тільки надані новини	+		
3	Існуючі інтерфейси	Є можливість створення клієнтів для інших платформ	Мультиплатформність що характеризується наявністю розсилки через email, веб-сайту, розширення	Надає інструменти, але не є готовою системою Не має готового додатку	Має тільки один інтерфейс – мобільний додаток. Має проблеми з оптимізацією на Android системах		+	

Продовження таблиці 7.2

№ п/п	Техніко-економічні характеристики ідеї	Товари/концепції конкурентів				W (слабка)	N(нейтральна)	S (сильна)
		Власна релізація	The Factual	Check by Meedan	Logically			
			для браузеру, мобільного додатку					
4	Точність аналізу інформації	Достовірність перевірки інформації з різних джерел, виключаючи людський фактор	Людське втручання в фінальний рейтинг достовірності інформації	Працює з багатьма популярними месенджерами, що розширюю спектр інформації для аналізу	Висока якість аналізу достовірності інформації			+
5	Доступність	Безкоштовний веб-додаток без потреби реєстрації	Обов’язкова реєстрація акаунту. Тільки платна підписка	Обов’язкова реєстрація акаунту. Тільки платна підписка	Безкоштовний мобільний додаток зі зручним інтерфейсом			+

Зазначений вище перелік слабких, сильних та нейтральних характеристик проекту та властивостей ідеї потенційного товару можна вважати підґрунтям для формування його конкурентоспроможності на ринку.

7.2 Технологічний аудит проекту

У рамках даного підрозділу проводиться облік технологій, за допомогою яких реалізацію ідеї проекту можна було б провести.

Визначення можливостей технологічної здійсненності концептуального проекту передбачає аналіз наступних складових (таблиця 7.3):

а) використання якої технології більш доцільно для виготовлення товару згідно ідеї проекту;

б) питання існування доцільної технології, або потреба її розробки;

в) доступність доцільних технологій авторам проекту.

Таблиця 7.3 – Технологічна здійсненність ідеї проекту

№ п/п	Ідея проекту	Технології її реалізації	Наявніс ть технологій	Доступність технологій
1	Розробка системи моделювання та прогнозування	Використання мови програмування Python	Наявна	Доступна
		Використання мови розмітки HTML	Наявна	Частково доступна
		Використання платформи Angular	Наявна	Частково доступна
		Використання мови програмування C#	Наявна	Доступна
		Використання ASP .Net Core	Наявна	Доступна

За результатами аналізу таблиці 7.3 можна зробити висновок щодо наявності технологічної можливості реалізації проекту. Так для технологічного шляху реалізації проекту, як найбільш доцільну, було обрано мову програмування Python.

7.3 Аналіз ринкових можливостей стартап-проекту

Визначення характеристик ринкових можливостей, що можна використати під час ринкового впровадження проекту, та ризиків на ринку, які можуть зашкодити реалізації проекту, дає можливість спланувати провідні напрямки розвитку проекту з врахуванням становища ринкової сфери, можливі запити потенційних клієнтів та варіанти їх вирішень проектами конкурентів.

Для початку проводиться аналіз попиту: його наявність, обсяг та динаміка розвитку ринку (таблиця 7.4).

Таблиця 7.4 – Попередня характеристика потенційного ринку стартап-проекту

№ п/п	Показники стану ринку (найменування)	Характеристика
1	Кількість головних гравців, од	3
2	Загальний обсяг продаж, грн/ум.од	2420000
3	Динаміка ринку (якісна оцінка)	Зростає
4	Наявність обмежень для входу (вказати характер обмежень)	Ліцензії, регуляторні
5	Специфічні вимоги до стандартизації та сертифікації	+
6	Середня норма рентабельності в галузі (або по ринку), %	8.3%

Результати аналізу таблиці за попередніми характеристиками потенційного ринку стартапу можна зробити висновок щодо певної міри привабливості даного ринку для реалізації продукту.

Наступний крок це опис потенційних груп споживачів для розробленого продукту, визначення їх ключових характеристик. Орієнтований перелік специфічних вимог до товару від кожної імовірної групи споживачів приводиться у таблиці 7.5.

Таблиця 7.5 – Характеристика потенційних клієнтів стартап-проекту

№ п/п	Потреба, що формує ринок	Цільова аудиторія (цільові сегменти ринку)	Відмінності у поведінці різних потенційних цільових груп клієнтів	Вимоги споживачів до товару
1.	Створення доступного, достовірного та швидкого додатку для перевірки тієї чи іншої інформації	Аналітичні відділи різних комерційних та державних структур	Автоматизація факт-чекінгу та перевірка інформаційних ресурсів на достовірність	Простота використання, автоматизація висока точність,
2.	Створення доступного, достовірного та швидкого додатку для перевірки тієї чи іншої інформації	Користувачі додатку	Зручність та простота використання. Доступність	Простота використання, достовірність швидкість обробки, різноманітність інформації, доступна ціна

Вслід за визначенням потенційних груп клієнтів робиться аналіз ринкового середовища. Для цього складаються таблиці факторів, що сприяють ринковій реалізації проекту, та факторів, що їй перешкоджають.

Таблиця 7.6 містить фактори загроз для продукту.

Таблиця 7.7 містить фактори можливостей для продукту.

Таблиця 7.6 – Фактори загроз

№ п/п	Фактор	Зміст загрози	Можлива реакція компанії
1	Конкуренція	Потенційна поява кращих продуктів на ринку з більш широким спектром характеристик, та кращою реалізацією у всіх напрямках розвитку продукту	Поліпшення програмного продукту, винахід та реалізація нових методів навчання нейронної мережі та її аналізу інформації. Пошук нових цільових сегментів. Розширення функціоналу.
2	Відсутність комерційної складової	Оскільки додаток буде розповсюджуватиметься на безоплатній основі, при низькому відвідуванні та/чи відсутності рекламодавців комерційної вигоди може не бути	Вкладення в рекламну складову, залучення SEO-спеціалістів, створення та введення платних функцій

Таблиця 7.7 – Фактори можливостей

№ п/п	Фактор	Зміст можливості	Можлива реакція компанії
1	Низька конкуренція	Ринок відносно новий, тому кількість продуктів що задовольняють потреби користувачів порівняно низька	Вдосконалення системи, розробка інтерфейсів на нові платформи, захоплення нових частин ринку
2	Вдосконалення методів навчання нейронних мереж	Розробка і реалізація нових та вдосконалення вже існуючих нейронних мереж, та способів їх навчання.	Розробка і реалізація нових та вдосконалення вже існуючих нейронних мереж, та способів їх навчання

Таблиця 7.8 містить у собі ступеневий аналіз загальних імовірних конкуренцій на ринку.

Таблиця 7.8 – Ступеневий аналіз конкуренції на ринку

Особливості конкурентного середовища	В чому проявляється дана характеристика	Вплив на діяльність підприємства
1. Вказати тип конкуренції – олігополія	На ринку присутня мала кількість компаній, продукти яких відрізняються в незначній мірі.	Вдосконалення продукту, розробка нових технологій факт-чекінгу, оптимізація цін

Продовження таблиці 7.8

Особливості конкурентного середовища	В чому проявляється дана характеристика	Вплив на діяльність підприємства
3. За галузевою ознакою – міжгалузева	Існує можливість використовувати продукт у різних галузях	Розробка та реалізація інструментів для локальних завдань
4. Конкуренція за видами товарів – товарно-видова	Конкуренція між програмним продуктами	Розробка власного програмного продукту з урахуванням слабких місць конкурентів
5. За характером конкурентних переваг – нецінова	Користувачі в першу чергу звертають на якісні характеристики програмного забезпечення	Поліпшення програмного продукту, винахід та реалізація нових методів навчання нейронної мережі та її аналізу інформації.
6. За інтенсивністю – не марочна	Наявність бренду не впливає на ефективність продукту.	Вкладення в рекламну складову, залучення SEO-спеціалістів, створення та введення платних функцій

Таблиця 7.9 містить у собі детальний аналіз умов конкуренції за моделлю Портера.

Це специфічна модель створена для аналізу конкуренції у конкретній галузі. М. Портер у своїй моделі вирізняє п’ять основоположних факторів, що безпосередньо впливають на привабливість ринку з урахування специфіки конкуренції. Цими факторами є:

- конкурент уже наявний в галузі;

- потенційний конкурент;
- існування товарів-замінників;
- постачальники;
- споживачі.

Сила позицій певної компанії за кожним окремим фактором визначає її можливості забезпечення динаміки обороту капіталу та здатність диктувати умови співпраці іншим агентам ринкових відносин.

Специфіка факторів, для різних галузей відповідно різна та змінна з плином часу. Сила кожного з факторів, тобто їх важливість, є виключеною до структури галузей та їх технічно-економічних специфік.

Таблиця 7.9 – Аналіз конкуренції в галузі за М. Портером

Складові аналізу	Прямі конкуренти в галузі	Потенційні конкуренти	По ста чальники	Клієнти	Товари-замінники
	The Factual Check by Meedan Logically	Наявність раніше вже існуючих рішень по проблематиці	–	Контроль якості, сприяння рейтингу та розширенню функціоналу	Більший функціонал

Продовження таблиці 7.9

Складові аналізу	Прямі конкуренти в галузі	Потенційні конкуренти	По ста чальники	Клієнти	Товари-замінники
Висновки:	На ринку існують декілька конкурентів, але кожен них має свої специфічні недоліки і з розвитком технологій нейронних мережі та розширенням інтерфейсів є високі можливості зайняття ніші на ринку	Високі ринкові можливості з розвитком технологій нейронних мережі та розширенням інтерфейсів, збільшення функціоналу	–	Клієнти диктують вимоги до програмного продукту. Основними критеріями є доступність, зручність інтерфейсу, достовірність інформації та розширення спектру інформації для перевірки	Ключовим є розвиток програмного продукту, введення нового функціоналу, розвиток різноманітних користувацьких інтерфейсів.

Проведений аналіз конкуренції, що міститься у таблиці 7.9, а також урахування сильних та слабких характеристик проекту, наведених у таблиці 7.2, вимог потенційних споживачів до товару, приведених у таблиці 7.5, та факторів загроз, факторів можливостей маркетингового середовища, котрі містяться у таблицях 7.6 та 7.7, дає змогу визначити та обґрунтувати перелік факторів конкурентоспроможності стартап-проекту.

Обґрунтування цих факторів конкурентоспроможності буде наведено у таблиці 7.10.

Таблиця 7.10 – Обґрунтування факторів конкурентоспроможності

№ п/п	Фактор конкурентоспроможності	Обґрунтування (наведення чинників, що роблять фактор для порівняння конкурентних проектів значущим)
1	Простота використання	Зручний інтерфейс веб-застосунку полегшує використання його для користувачів, а відсутність нагромадження зайвої інформації робить його максимально лаконічним та клієнтоорієнтованим
2	Багатофункціональність	Функціонал обмежується базовою перевіркою інформації різного роду, а не тільки заданого спектру новин, а також має потенціал до розширення спеціалізацій
3	Мультиплатформність	Оскільки ядро проекту міститься на веб-сервері, він має потенціал та інструментарій для порівняно легкого розширення на інші користувацькі платформи
4	Точність аналізу інформації	Розроблена нейронна мережа має здатність до достовірної перевірки інформації з різних джерел, оскільки виключає людський фактор
5	Доступність	Додаток планується запускати на безоплатній основі, та без потреби в реєстрації на сайті, що підвищує доступність сервісу серед більшої кількості потенційних клієнтів

За вище визначеними факторами конкурентоспроможності стартап-проекту у таблиці 7.10 проводиться додатковий аналіз сильних та слабких сторін проекту, що наведено у таблиці 7.11.

Таблиця 7.11 – Порівняльний аналіз сильних та слабких сторін стартап-проекту

№ п/п	Фактор конкурентоспроможності	Бали 1- 20	Рейтинг товарів-конкурентів у порівнянні з власним стартап-проектом						
			-3	-2	-1	0	+1	+2	+3
1	Простота використання	17						+	
2	Багатофункціональність	12				+			
3	Мультиплатформність	6		+					
4	Точність аналізу інформації	18						+	
5	Доступність	19							+

Заключним кроком ринкового аналізу можливостей реалізації стартап-проекту є SWOT-аналіз (матриці аналізу сильних (Strength) і слабких (Weak) сторін, загроз (Troubles) і можливостей (Oportunities) продукту) на основі раніше виділених загроз та можливостей, а також аналізу сильних і слабких сторін стартап-проекту, котрі наведено у таблиці 7.11.

Орієнтовний перелік ринкових загроз та можливостей на ринку для стартап-проекту складається на базі факторів загроз та факторів можливостей стартапу. Ринкові загрози та ринкові можливості є нічим іншим як наслідками впливу відповідних факторів і ще не є реалізованими на ринку, хоча потенційно можуть реалізуватися з певною ймовірністю. Себто зниження прибутків потенційних споживачів – це відповідний фактор загрози, на базі якого можна зробити ймовірний

прогноз щодо посилення значущості цінового фактору у виборі товару та ціновій конкуренції. Результати проведеного SWOT-аналізу наведено у таблиці 7.12.

Таблиця 7.12 – SWOT-аналіз стартап-проекту

Сильні сторони:	Слабкі сторони:
Простота використання Точність аналізу інформації Доступність	Функціональність Мультиплатформність
Можливості:	Загрози:
Розробка нових технологій навчання нейронних мереж	Монополізація ринку, зміна потреб користувачів, неприбутковість

На базі SWOT-аналізу доцільно розробити альтернативи ринкової поведінки для реалізації стартап-проекту на ринку, а також орієнтовний час реалізації альтернатив з огляду на ймовірні проекти конкурентів, що могли би бути виведені на ринок. Альтернативи розвитку продукту містяться у таблиці 7.9.

Визначені можливості альтернативи аналізуються з точки зору орієнтовних строків реалізації, а також ймовірності отримання ресурсів при виборі тих чи інших альтернатив. Ці данні містяться у таблиці 7.13

Таблиця 7.13 – Альтернативи ринкового впровадження стартап-проекту

№ п/п	Альтернатива (орієнтовний комплекс заходів) ринкової поведінки	Ймовірність отримання ресурсів	Строки реалізації
1	Одноразовий продаж продукту з подальшою платною технічною підтримкою	60%	12 місяців
2	Продаж продукту по підписці на місяць або рік	80%	16 місяців
3	Одноразовий продаж продукту з подальшим продажом додаткового функціоналу	45%	18 місяців
4	Безкоштовне розповсюдження продукту з подальшим продажом додаткового функціоналу	25%	20 місяців
5	Безкоштовне розповсюдження продукту з заробітком на внутрішній рекламі	15%	24 місяці

За результатами SWOT-аналізу було обрано альтернативу № 5.

7.4 Аналіз ринкової стратегії проекту

Розробка ринкової стратегії одним перших кроків передбачає визначення стратегії охоплення ринку, яка містить у своєму описі характеристику цільових груп потенційних користувачів продукту. Цільові групи користувачів застосунку містяться у таблиці 7.14.

Таблиця 7.14 – Вибір цільових груп споживачів

№ п/п	Опис профілю цільової групи потенційних клієнтів	Готовність споживачів сприйняти продукт	Орієнтовний попит в межах цільової групи (сегменту)	Інтенсивність конкуренції в сегменті	Простота входу у сегмент
1	Великі компанії, які ведуть свою діяльність у телевізійно-інформаційній сфері	Висока	Високий	Сильна	Складно
2	ФОП, що ведуть діяльність у сфері журналістики	Висока	Високий	Помірна	Складно
3	ФОП, що ведуть діяльність у SEO сфері.	Помірна	Помірний	Слабка	Просто
4	Побутове використання фізичними особами	Помірна	Помірний	Помірна	Просто

За результатами аналізу цільових груп вирішено обрати для праці з усі чотири цільові групи. Підсумки аналізу потенційних груп споживачів, що були б зацікавленні у надбанні або використанні продукту.

Визначення стратегії: для охоплення якомога ширшого ринку було вирішено використовувати стратегію масового маркетингу, себто компанія працює з усім

спектром ринку, пропонуючи стандартизовану програму з певними характеристиками товару.

За вище згаданим М. Портером, існує три базових стратегій розвитку, кожна з яких відрізняється ступенем охоплення цільового ринку, а також типом конкурентної переваги, що має реалізовуватись на ринку.

Перша з них, стратегія лідерства по витратах. Сутність її в тому, що за рахунок внутрішніх та зовнішніх чинників середовища компанія може виставляти більшу за конкурентну маржу між собівартістю та середньою ринковою ціною товару, в порівнянні з конкурентами.

Ця стратегія допускає, що за рахунок більших можливостей збуту товарів по об'єму і продуктивності, можна добитись менших витрат. Така специфіка стратегії, як правило, пов'язана з можливістю досягти ефекту масштабу та досвіду.

Для компаній що обирають цю базову стратегію розвитку, обов'язковим стає ретельний контроль за регулярними витратами, зниження виробничих, збутових та рекламних витрат, проведення інвестицій що зменшують витрати та реальне опрацювання конструювання нових товарів.

Перевагами даної стратегії є:

- здатність фірми протистояти своїм конкурентам у випадку цінової війни та здатні отримувати прибуток при мінімально допустимій ціні;
- не здатність сильних клієнтів добитись зниження ціни нижче прийнятного для найсильнішого конкурента рівня;
- забезпечення захисту проти сильних постачальників, за рахунок більшої гнучкості у випадках підвищення вхідних витрат;
- існування бар'єру входу для конкурентів-новачків, а також захист від товарів-замінників.

Якщо в конкурентній боротьбі буде застосована ця стратегія з ринку будуть вимушені піти менш ефективні та великі фірми, що нездібні до введення технологічних інновацій.

Наступна стратегія це – стратегія диференціації. Сутність цієї стратегії в опорі на інтереси споживача та надання товару з конкретними властивостями відмінними

від товарів-конкурентів. Ця відмінність може покладатись як на об'єктивні так і на суб'єктивні властивості товару, чи навіть на такі що не є реальними. Основним інструментом для реалізації стратегії диференціації служить ринкове позиціонування.

Перевагами даної стратегії є:

- зниження ступеню замінності товару, чутливості до ціни тим самим підвищуючи прихильність до марки та рентабельність;
- зменшення тиску на компанію за рахунок прихильності клієнтів та перешкоджання появі нової конкуренції та товарів-замінників;
- ріст рентабельності корелює зі стійкістю до витрат при впливові постачальника.

Специфіка цієї стратегіє полягає у високі витрат. Проте, при успішній реалізації стратегії, товар здобуває вищу рентабельність на ринку за рахунок цінової премії бренду.

Якщо в конкурентній боротьбі буде застосована ця стратегія з ринку будуть вимушені піти компанії, що не здатні визначити потреби своєї цільової аудиторії та оперативно реагувати на зміни у попиті.

Останньою є стратегія спеціалізації. Її сутність полягає у фокусуванні на одному цільовому сегменті ринку. Завданням є задоволення потреб обраного сегменту краще ніж у конкурентів. Специфікою цієї стратегії є те що вона може спиратись на перші дві вище зазначені стратегії, в тій чи іншій мірі, але у рамках свого мікро сегменту. Ризиком цієї стратегії є низька ринкова частка, котра може привести до не конкурентоспроможності компанії.

Базова стратегія розвитку, що була обрана для роботи з сегментами ринку сформована у таблиці 7.15.

Таблиця 7.15 – Визначення базової стратегії розвитку

№ п/п	Обрана альтернатива розвитку	Стратегія охоплення ринку	Ключові конкурентоспроможні позиції відповідно до обраної альтернативи	Базова стратегія розвитку
1	Безкоштовне розповсюдження продукту з внутрішньою контекстною рекламою	Для усіх сегментів ринку поставляється стандартизована послуга, і весь проект має одну стратегію маркетингу	Зручність використання та доступність продукту, точність перевірки.	Стратегія масового маркетингу

Виділяються чотири стратегії конкурентної поведінки, кожна з яких має свою специфіку.

а) стратегія лідера;

б) стратегія виклику лідера;

в) стратегія наслідування лідера;

г) стратегія заняття конкурентної ніші.

В залежності від ступеню сформованості ринку та специфіки конкурентної боротьби лідерські компанії обирають одну зі стратегій: первинного попиту, оборони, наступальну чи демаркетинг.

Стратегія розширення первинного попиту використовується зазвичай коли компанія зацікавлена у великій економічній віддачі нехтуючи невеликими компаніями-конкурентами. Компанія, у такому випадку, впроваджує заходи з формування попиту, пропаганди нових напрямків використання вже існуючих товарів та пошуком нових груп споживачів.

Ця стратегія має місце бути тільки на перших стадія життєвого циклу товарів, коли тиск конкуренції незначний.

Негативними сторонами цієї стратегії є її витратність та швидкоплинність. По мірі розширення ринку, компанії-імітатори будуть атакувати компанію-новатора. В

такому випадку, доцільною є перехід до стратегії оборони, що напрямлена на захист ринкової долі. Вона може бути:

- інноваційною, з метою встановлення технологічних бар'єрів входу на ринок;
- розширення товарного асортименту та захоплення каналів збути задля ліквідації вільних торгових ніш на ринку;
- проведення масштабних рекламних кампаній чи ведення цінової війни.

Наступним або альтернативним кроком є збільшення своєї частки ринку, що передбачається наступальною стратегією. Мета компанії, у такому випадку, це підвищення прибутковості кампанії за рахунок максимального використання масштабу. Ця стратегія також припускає інноваційні введення задля розриву між собою та компаніями-конкурентами.

Основними мінусами цієї стратегії є або економічна недоцільність боротьби за частку з дрібними виробниками або підпадання під дію антимонопольного законодавства. У другому випадку використовується стратегія демаркетингу – скорочення частки ринку та зниження попиту за рахунок підняття ціни.

Стратегія виклику лідеру більш характерна для других чи третіх компаній на ринку, що бажають стати лідером. Умовно вони можуть обрати атаку на лідера чи слідування за ним.

Перший варіант для компаній є відносно ризикованим, оскільки вимагає значних фінансових витрат, інновацій, кращого співвідношення ціни та якості, переваг у маркетингу і тому подібного. У випадку ж провалу, компанія, вірогідно, буде посунута з ринку. Для успішної атаки на лідера необхідно опрацювати наступне:

- проаналізувати свої та лідируючої компаній сильні та слабкі сторони;
- виявити можливі напрями атаки;
- провести ревізію своїх ресурсів та сил;
- проаналізувати можливі дії конкурентів та розробити методи захисту від них.

В залежності від результатів опрацювання, компанія може обрати фронтальну чи флангову атаку.

Перша – допускає атаку на сильні сторони компанії-лідера. При цьому, компанія повинна мати перевагу над тим, що атакує. Складність полягає у подальшому утриманню позицій, не тільки від компанії на яку було скоєно «атаку», а й від інших компаній що займають треті, четверті місця. З цих причин ця стратегія є доволі ризиковою.

Друга – допускає атаку на слабкі сторони компанії-лідера. Такими може стати неосвоєні сторони ринку, сегменти, значущий сервіс чи ціна. Найбільш ефективним є удар по ціновій політиці лідерської компанії, оскільки маючи більшу ринкову частку падіння ціни приносить великі витрати.

Стратегія слідування за лідером найбільш прийнятна для компаній з невеликою часткою ринку, які не конкурують з лідером, а йдуть фарватері з ним. Перевагою такої стратегії є економія фінансових ресурсів, що не витрачаються на розширення та утримання домінуючого положення.

Данна стратегія найбільш поширена в умовах олігополії, коли конкуренції уникають усі конкуренти, а також коли ефект масштабу не працює та дає істотних переваг на ринку. Також цю стратегію приймають ті компанії які не отримали успіху зі стратегією виклику лідерові.

Основною діяльністю компаній, що користуються цією стратегією є випуск товарів-імітаторів, що утримують ті долі ринку що не змогли захопити лідери. Також, вибір цієї стратегії може бути зумовлений локалізаційною перевагою.

Ефективна реалізація цієї стратегії передбачає опрацювання наступних чинників:

- аналіз сегментації ринку задля виявлення нових потенційних ринкових сегментів;
- фокусування на прибутку;
- систематичний аналіз витрат на логістику та виробництво;
- не привертати увагу фірм-лідерів;
- мати сильне керівництво.

Оскільки лідерами ринку може бути тільки обмежена кількість компаній, стратегія слідування за лідером є найпоширенішою.

Остання стратегія це – стратегія зайняття конкурентної ніші, хоча вона має і інші назви: стратегія фахівці або стратегія нішера. Вона полягає у виборі одного чи кількох невеликих сегментів ринку. Компанії що обирають цю стратегію обирають цю стратегію концентруються на своїй невеликій ніші.

Для ефективності цієї стратегії ніша повинна задовольняти певні умови:

- бути прибутковою достатньо аби компанія не працювала собі ж у збиток;
- бути стабільною довгий час;
- мати високі вхідні бар’єри;
- не бути привабливою для компаній-конкурентів;
- бути відповідною ресурсам та можливостям компанії.

У цій стратегії, основною задачею компанії є постійне вдосконалення свого продукту, підтримка конкурентної переваги, формування та підтримка лояльності та прихильності споживачів, а також витримування високих вхідних бар’єрів ринку.

Таблиця 7.16 містить у собі обрану стратегію конкурентної поведінки.

Таблиця 7.16 – Визначення базової стратегії конкурентної поведінки

№ п/п	Чи є проект «першопрохідцем» на ринку?	Чи буде компанія шукати нових споживачів, або забирати існуючих у конкурентів?	Чи буде компанія копіювати основні характеристики товару конкурента, і які?	Стратегія конкурентної поведінки*
1	Продукт є першопрохідцем на ринку на національному рівні	Проект буде шукати нових споживачів	Проект займатиметься розвитком та реалізацією вдосконаленої нейронної мережі незалежно від наявності схожих чи суміжних характеристик у конкурентів	Стратегія заняття конкурентної ніші

З опору на вимоги обраної цільової аудиторії споживачів до продукту та розробники зазначених у таблиці 7.5. З огляду на обрану базову стратегію розвитку продукту, що міститься у таблиці у таблиці 7.15. А також обрану стратегію конкурентної поведінки, яка наведена у таблиці 7.16. надалі розробляється стратегія позиціонування продукту, сутність якої полягає у формуванні ринкової позиції, за якою споживачі можуть впізнати торгівельну марку або сам проект. Дана стратегія позиціонування продукту міститься у таблиці 7.17.

Таблиця 7.17 – Визначення стратегії позиціонування

№ п/п	Вимоги цільової аудиторії до товару	Базова стратегія розвитку	Ключові конкурентоспроможні позиції власного проекту	Вибір асоціацій, які мають сформувати комплексну позицію власного проекту
1	Зручність інтерфейсу, точність перевірки даних доступність продукту	Стратегія масового маркету	Поширення стандартизованого легкодоступного, в порівнянні з конкурентами сервісу, з можливістю надбудов функцій та користувацьких інтерфейсів на національному ринку	Доступність, простота використання, достовірність результату, можливість ширшого використання

7.5 Розроблення маркетингової програми стартап-проекту

Для означення маркетингової концепції товару, для початку підсумуємо та наведемо у таблиці 7.18 загальні результати попереднього аналізу конкурентоспроможності продукту.

Таблиця 7.18 – Визначення ключових переваг концепції потенційного товару

№ п/п	Потреба	Вигода, яку пропонує товар	Ключові переваги перед конкурентами(існуючі або такі, що потрібно створити
1	Простота використання	Зручний інтерфейс веб-застосунку полегшує використання його для користувачів, а відсутність нагромадження зайвої інформації робить його максимально лаконічним та клієнтоорієнтованим	Зручний інтерфейс, та відсутність нагромадження зайвої інформації
2	Багатофункціональність	Функціонал обмежується базовою перевіркою інформації різного роду, а не тільки заданого спектру новин, а також має потенціал до розширення спеціалізацій	Можливість перевірки інформації різного роду
3	Мультиплатформність	Оскільки ядро проекту міститься на веб-сервері, він має потенціал та інструментарій для порівняно легкого розширення на інші користувацькі платформи	Можливість подальшого розширення на різні платформи

Продовження таблиці 7.18

№ п/п	Потреба	Вигода, яку пропонує товар	Ключові переваги перед конкурентами(існуючі або такі, що потрібно створити
4	Точність аналізу інформації	Розроблена нейронна мережа має здатність до достовірної перевірки інформації з різних джерел, оскільки виключає людський фактор	Відсутність людського фактору у перевірці інформаціє гарантує збільшення її достовірності
5	Доступність	Додаток планується запускати на безоплатній основі, та без потреби в реєстрації на сайті, що підвищує доступність сервісу серед більшої кількості потенційних клієнтів	Безкоштовність, відсутність реєстрації та локалізація на веб-домені полегшує доступ та його різного роду користувачам

Наступним кроком розробляється трирівнева маркетингова модель для товару: опис ідеї продукту, його фізичні характеристики, а також особливості впровадження продукту. Маркетингова модель продукту наведена у таблиці 7.19.

1-й рівень. Підчас уточнення ідеї товару визначається, засобом для вирішення якої проблеми буде розроблений нами продукт, його центральна користь для користувача. Проблематика напрямку пов’язана із розробкою технічного завдання вже у процесі розробки документації на продукт.

2-й рівень. Другий рівень описує стан вже реалізованого продукту. В цьому рівні описується, як якість продукту, так і його властивості, його дизайн, можлива упаковка та ціна.

3-й рівень. Третій містить у собі додаткові послуги та переваги для споживача, котрі опираються на опис товару за концепцією та опис товару у реальному виконанні.

Таблиця 7.19 – Опис трьох рівнів моделі товару

<table>
<tr><td>Рівні товару</td><td colspan="3">Сутність та складові</td></tr>
<tr><td>I. Товар за задумом</td><td colspan="3">Зручність, швидкість, доступність та достовірність перевірки інформації заданої за запитом, зокрема новинних фейків.</td></tr>
<tr><td rowspan="5">II. Товар у реальному виконанні</td><td>Властивості/характеристики</td><td>М/Нм</td><td>Вр/Тх/Тл/Е/Ор</td></tr>
<tr><td>- функція перевірки інформації за заданими критеріями
- отримання бази інформаційних джерел що підтверджують чи спростовують інформацію</td><td></td><td></td></tr>
<tr><td colspan="3">Якість: достовірність статистичного аналізу інформації</td></tr>
<tr><td colspan="3">Пакування: відсутнє</td></tr>
<tr><td colspan="3">Марка: DKFCheck</td></tr>
<tr><td rowspan="2">III. Товар із підкріпленням</td><td colspan="3">До продажу: відсутнє</td></tr>
<tr><td colspan="3">Після продажу: впровадження безкоштовних оновлень системи, технічна підтримка користувачів</td></tr>
<tr><td colspan="4">Вихідний код системи та код реалізації нейронної мережі будуть закриті.</td></tr>
</table>

Надалі необхідно визначити цінову політику та межі, які будуть використовуватись при реалізації програмного продукту та встановленні ціни на покупку чи користування. Для визначення цінових меж передбачається аналіз ціни на аналогічні продукти чи замінники конкурентів. Окрім цього варто провести аналіз рівня доходів обраної групи користувачів. Визначення меж встановлення ціни зазначено у таблиці 7.20.

Таблиця 7.20 – Визначення меж встановлення ціни

№ п/п	Рівень цін на товари-замінники	Рівень цін на товари-аналоги	Рівень доходів цільової групи споживачів	Верхня та нижня межі встановлення ціни на товар/послугу
1	12$	10$	Рівень доходів достатній майже у всіх рівнів користувачів	Безкоштовне розповсюдження з контекстною рекламою по 5-10$ на місяць

Після цього визначається оптимальна система збуту для розробленого продукту. В межах цього рішення можуть бути варіанти:

- власного або залученого збуту;
- обґрунтований вибір глибини каналу;
- обґрунтований вибір посередників.

Вибір та обґрунтування системи збуту для даного проекту наведена у таблиці 7.21.

Таблиця 7.21 – Формування системи збуту

№ п/п	Специфіка закупівельної поведінки цільових клієнтів	Функції збуту, які має виконувати постачальник товару	Глибина каналу збуту	Оптимальна система збуту
1	Бажання автоматизовано та достовірно перевірити певну інформацію для подальшого виростання	Формування попиту, стимулювання збуту. Встановлення контактів із споживачами. Збір маркетингової інформації.	Перший рівень (від розробни ка до споживача)	Прямий канал збуту до споживача власними силами

Останнім компонентом розробки повноцінної маркетингової програми є побудова певної концепції маркетингових комунікацій, яка враховує обрану політику позиціонування. Ця концепція маркетингових комунікацій описується у таблиці 7.22.

Таблиця 7.22 – Концепція маркетингових комунікацій

№ п/п	Специфіка поведінки цільових клієнтів	Канали комунікацій, якими користуються цільові клієнти	Ключові позиції, обрані для позиціонування	Завдання рекламного повідомлення	Концепція рекламного звернення
1	Цільові клієнти мають на меті впровадити засоби, що дають змогу достовірно, оптимально та автоматизовано перевіряти інформацію	Інтерне, конференції та семінари	Позиція на основі порівняння фірми з товарами конкурентів та орієнтація на національний ринок	Формування репутації компанії, збільшення кількості клієнтів, підвищення рівню впізнаваності компанії	DKFCheck – вір тільки перевіреним новинам!

Результатом розробки у даному підрозділі стартап-проекту, стала створена стійка ринкова програма, котра включає у себе концепцію програмного продукту, концепцію збуту програмного продукту, його просування та попередній аналіз наявності умов ціноутворення для розробленої системи, а також враховує цінності та потреби потенційних споживачів, конкурентні переваги стартап-проекту та динаміку ринкового середовища, межі якого будуть впроваджувати проект та обрану для нього альтернативу поведінки на ринку збуту.

Висновки до розділу

Даний розділ дипломного проекту містить у собі проведений аналіз програмного продукту у якості стартап-проекту.

Стартап-проект має можливість ринкової комерціалізації. Динаміка та попит на ринку цього роду послуг зростаюча, особливу якщо розглядати національний ринок.

З огляду на потенційні групи клієнтів можна засвідчити низькі бар'єри входження та порівняно невелику конкуренцію.

На даний момент на ринку представлена олігополічна конкуренція, тобто існує декілька схожих проектів від невеликих компаній, які пропонують схожі функціональні можливості для користувачів. На національному ринку даний стартап-проект міг би виступити монополістичним лідером, для чого необхідна необхідно розробити додаткову альтернативну стратегію виходу, рекламну кампанію та готову стратегію впровадження для комерційних груп потенційних споживачів.

ВИСНОВКИ

В результаті виконання магістерської роботи було створено систему перевірки достовірності тверджень.

Було проведено аналіз предметної області, визначено цілі і задачі розробки, а також її основні особливості та критерії ефективності. Був проведений аналіз існуючих альтернатив та проаналізовано переваги й недоліки кожної. На їх основі створено порівняльну таблицю. З отриманих даних було сформовано функціональні та нефункціональні вимоги до системи. На етапі проектування застосунку було проведено вибір технологій розробки, з детальним обґрунтуванням вибору кожної. На основі вимог до системи, визначених раніше було розроблено автоматизовану систему фактчекінгу.

Було продемонстровано, що прихована модель скалярного добутку на основі BERT, навчена за допомогою вибірки softmaxloss, може перевершити традиційне розріджене пошук як окрема модель і призводить до значних покращень при використанні в багатоступеневій архітектурі ранжирування. Було створен новий синтетичний набір даних для фактичної перевірки оцінки пошуку доказів під назвою Factual-NLI+ і доведено, що попереднє навчання на існуючих наборах даних NLI та новий набір даних значно покращує модель пошуку.

Використовуючи навчену модель пошуку, було створено систему семантичного пошуку статей новин, щоб продемонструвати її ефективність у масштабному наборі даних реального світу. Розроблена модель і використані набори даних опубліковані як частина системи семантичного пошук. У майбутньому було б цікаво побачити, як щільний пошук принесе користь повній системі перевірки фактів у набагато більшому масштабі. Доповнення Factual-NLI+ шумом із веб-результатів дав більш складний контрольний показник, який демонструє кращу ефективність і відгук у порівнянні з традиційним розрідженим пошуком в реальних умовах.

Також був проведений аналіз програмного продукту у якості стартап-проекту. Стартап-проект має можливість ринкової комерціалізації. Динаміка та попит на ринку цього роду послуг зростаюча, особливу якщо розглядати національний ринок.

На даний момент на ринку представлена олігополічна конкуренція, тобто існує декілька схожих проектів від невеликих компаній, які пропонують схожі функціональні можливості для користувачів. На національному ринку даний стартап-проект міг би виступити монополістичним лідером, для чого необхідна необхідно розробити додаткову альтернативну стратегію виходу, рекламну кампанію та готову стратегію впровадження для комерційних груп потенційних споживачів.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1) Pepa Atanasova. Generating fact checking explanations /Jakob Grue Simonsen, Christina Lioma, and Isabelle Augenste // In Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL), 2020. – pp.20–56.
2) Karen S. Jones. Natural language processing: a historical review // Cambridge: Computer Laboratory, University of Cambridge, 2001. – p.2
3) Andreas Hanselowski. Ukp-athene: Multi-sentence textual entailment for claim verification / Hao Zhang, Zile Li, Daniil Sorokin, Benjamin Schiller, Claudia Schulz, and Iryna Gurevych. // In Proceedings of the First Workshop on Fact Extraction and VERification (FEVER), 2018. – pp.103–108.
4) Анисимов А. В., Марченко А. А. Система обработки текстов на естественном языке // «Штучний інтелект». – 2002. – 157 с.
5) H.R. Madala and A.G. Ivakhnenko, “Inductive learning algorithms for complex systems modelling”, Boca Raton, Florida, USA, CRC Press, 1994.
6) Zeynep Akkalyoncu Yilmaz. Crossdomain modeling of sentence-level evidence for document retrieval / Wei Yang, Haotian Zhang, and Jimmy Lin. // In EMNLP, 2019. – pp.62–112.
7) Patrice Bellot. Overview of INEX 2013 / Antoine Doucet, Shlomo Geva, Sairam Gurajada, Jaap Kamps, Gabriella Kazai, Marijn Koolen, Arunav Mishra, Veronique Moriceau, ´ Josiane Mothe, Michael Preminger, Eric Sanjuan, Ralf Schenkel, Xavier Tannier, Martin Theobald, Matthew Trappett, and Qiuyue Wang. // In CLEF, 2013. – pp.13–38.
8) lex Nikolov. Identifying check-worthy tweets with transformer models / Giovanni Da San Martino, Ivan Koychev, and Preslav Nakov. Team Alex // In CLEF, 2020. – pp.90–164.
9) Kenton Lee. Latent retrieval for weakly supervised open domain question answering / Ming-Wei Chang, and Kristina Toutanova. // In Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL), 2019. – pp.44–78.
10) Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Matthew Kelcey, Jacob Devlin,

Kenton Lee. Natural questions: a benchmark for question answering research / Kristina N. Toutanova, Llion Jones, Ming-Wei Chang, Andrew Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov // Transactions of the Association of Computational Linguistics, 2019. – pp.18–23.

11) Ivakhnenko A. G., Ivakhnenko G. A. The review of problems solvable by algorithms of the group method of data handling (GMDH) //Pattern Recognition And Image Analysis C/C Of Raspoznavaniye Obrazov I Analiz Izobrazhenii. –1995. – T. 5. – C. 527-535.
12) Seshadri S. Angular: up and running: learning angular, step by step / Shyam Seshadri. – [Б. м.] : O'Reilly Media, 2018. – 312 с.
13) ASP .Net Core URL: https://docs.microsoft.com/en-us/aspnet/core/introduction-to-aspnet-core?view=aspnetcore-5.0
14) Alber, Maximilian. iNNvestigate neural networks! / Sebastian Lapuschkin, Philipp Seegerer, Miriam Hägele, Kristof T. Schütt, Grégoire Montavon, Wojciech Samek, Klaus-Robert Müller, Sven Dähne, 2019. – pp.1–50.
15) Long J. Fuly convolutional networks for semantic segmentation / J. Long, E. Shelhamer, T. Darrell. // CoRR. – 2014. – pp.85–113.
16) Gunji N. Classifier entry / N. Gunji, T. Higuchi, K. Yasumoto // ILSVRC. – 2012. – pp.3–20.
17) Dominik Stammbach. e-fever: Explanations and summaries for automated fact checking // In Proceedings of the Conference on Truth and Trust Online (TTO) – 2020. pp.79–98.
18) Jeff Johnson. Billion-scale similarity search with gpus // IEEE Transactions on Big Data. – 2019. – pp.3–20.
19) Takuma Yoneda. Ucl machine reading group: Four factor framework for fact finding (hexaf) // In Proceedings of the First Workshop on Fact Extraction and VERification (FEVER). . – 2019. – pp.89–92.
20) Adina Williams. A broad-coverage challenge corpus for sentence understanding through inference // In Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics. – 2018. – pp.2–7.