ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Кафедра інженерії програмного забезпечення

Пояснювальна записка

до магістерської роботи
на ступінь вищої освіти магістр

на тему: **«ОПТИМІЗАЦІЯ ОБРОБКИ ВІДГУКІВ КЛІЄНТІВ МЕРЕЖІ СУПЕРМАРКЕТІВ З ВИКОРИСТАННЯМ МЕТОДІВ NATURAL LANGUAGE PROCESSING»**

Виконав: студент 6 курсу, групи ПДМ-62
спеціальності
121 Інженерія програмного забезпечення
(шифр і назва спеціальності)

Шведченко І.С.
(прізвище та ініціали)

Керівник Аверічев І.М.
(прізвище та ініціали)

Рецензент
(прізвище та ініціали)

Київ – 2022

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Кафедра Інженерії програмного забезпечення_____
Ступінь вищої освіти – «Магістр»_____
Спеціальність підготовки – 121 «Інженерія програмного забезпечення»_____

ЗАТВЕРДЖУЮ
Завідувач кафедри
Інженерії програмного
забезпечення
_____Негоденко О.В.___
“____” _____2022 року

З А В Д А Н Н Я
НА МАГІСТЕРСЬКУ РОБОТУ СТУДЕНТА

ШВЕДЧЕНКО ІВАНА СЕРГІЙОВИЧА
(прізвище, ім’я, по батькові)

1. Тема роботи: «Оптимізація обробки відгуків клієнтів мережі супермаркетів з використанням методів Natural Language Processing»

Керівник роботи: Аверічев І.М., к.е.н.
(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)

Затверджені наказом закладу вищої освіти від «14» жовтня 2022 року №122.
2. Строк подання студентом роботи «31» грудня 2022 року
3. Вхідні дані до роботи
Матеріали переддипломної практики, методи моделювання;
Науково-технічна література з питань, пов’язаних з використанням методів Natural Language Processing

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити).
4.1 Огляд предметної області.
4.2 Аналіз існуючих підходів і методів оброблення природньої мови
4.3 Розробка моделей та методів.
4.4 Розробка програмного забезпечення моделей.
4.5 Моделювання та аналіз результатів.

5. Графічна частина роботи представлена на 12 слайдах презентації.

1. Мета, об’єкт та предмет дослідження
2. Актуальність проблеми
3. Аналіз існуючих підходів і методів
4. Алгоритм вибору найкращої системи
5. Визначення оптимальної системи управління
6. Висновки

6. Дата видачі завдання «14» жовтня 2022 року

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів магістерської роботи	Строк виконання етапів роботи	Примітка
1.	Отримання завдання на магістерську роботу	14.10.2022	Виконано
2.	Огляд предметної області	17.10.2022	Виконано
3.	Аналіз існуючих підходів і методів оброблення природньої мови	20.11.2022	Виконано
4.	Розробка моделей та методів	22.11.2022	Виконано
5.	Розробка програмного забезпечення моделей	25.11.2022	Виконано
6.	Тестування програмного забезпечення	24.11.2022	Виконано
7.	Моделювання та аналіз результатів	26.11.2022	Виконано
8.	Написання та оформлення пояснювальної записки	27.11.2022	Виконано
9.	Розробка графічних та презентаційних матеріалів	11.12.2022	Виконано
10.	Захист магістерської роботи	18.01.2023	Виконано

Студент _____ Шведченко І.С.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Аверічев І.М.
(підпис) (прізвище та ініціали)

РЕФЕРАТ

Атестаційна робота магістра містить: 75 с., 52 рис., 2 табл., 35 джерел.

МАШИННЕ НАВЧАННЯ, АНАЛІЗ ДАНИХ, МЕТОДИ МАШИННОГО НАВЧАННЯ, ТОНАЛЬНІСТЬ ТЕКСТУ, МЕРЕЖІ-ТРАНСФОРМЕРИ.

Об’єкт дослідження: процес керування обробки відгуків клієнтів мережі супермаркетів.

Ціллю роботи є дослідження методів аналізу тексту з використанням мереж-трансформерів.

Методи дослідження, які були використані під час роботи включають в себе теоретичне дослідження трьох моделей: XLNet, BERT та ULMFiT для розуміння актуальних досягнень у досліджуваному напрямку, а саме класифікації відгуків клієнтів товарів мережі супермаркетів, а також практичне дослідження у вигляді експериментів з цими моделями для аналізу та порівняння їх придатності до використання у аналізі тексту.

В результаті роботи було виявлено найбільш оптимальну для вирішення вищевказаної задачі за показником точності. Результати роботи можуть бути використані для подальшого дослідження розглянутих мовних моделей для вирішення поставленої задачі.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

API	–	Application Programming Interface
MFCC	–	Mel-frequency
DCT	–	Discrete Cosine Transform
DTW	–	Dynamic Time Wrapping
NN	–	Neural Networks
IDE	–	Integrated Development Environment
LPC	–	Linear Predictive Coding
АЧХ	–	амплитудно-частотна характеристика
ПММ	–	прихована марківська модель
DSP	–	цифрова обробка сигналів
САРМ	–	система автоматичного розпізнавання мови
BERT	–	Bidirectional Encoder Representations from Transformers
ULMFiT	–	Universal Language Model Fine-tuning for Text Classification
XLNet	–	узагальнений метод авторегресивного попереднього навчання
LSTM	–	штучна архітектура рекурентних неиронних мереж (RNN)
ШІ	–	штучнии інтелект
ОПМ	–	оброблення природньої мови
НМ	–	нейрона мережа
MLM	–	masked language modelling

ВСТУП

З метою створення ефективних систем спілкування між людьми феномен мовленнєвої комунікації інженери та вчені вивчали ще до винаходу Олександра Грема Белла. Починаючи з 1960-х років, цифрова обробка сигналів (DSP) відігравала центральну роль у вивченні та обробки мовлення. Сьогодні DSP є втіленням плодів знань, отриманих протягом десятиліть досліджень. Одночасні досягнення в технології інтегральних схем і комп'ютерної архітектури створили технологічне середовище з практично безмежними можливостями для інновацій у додатках мовної комунікації.

У сучасних дослідженнях роль методів DSP та застосуванні мовленнєвої комунікації відіграє головну роль. Також надається огляд цифрової обробки мовлення, який змінюється від основної природи мовного сигналу, через різноманітні методи представлення мовлення в цифровій формі, до застосувань у голосовому спілкуванні та автоматичному синтезі і розпізнаванні мовлення. Визначення розмовної мови у наші дні залишається актуальним завданням, оскільки досі немає методів, дозволяють однозначно вирішити це завдання.

Обробка природної мови – загальний напрямок штучного інтелекту та лінгвістики. Він вивчає проблеми комп'ютерного аналізу та синтезу природної мови. Стосовно штучного інтелекту аналіз означає розуміння мови, а синтез – генерацію розумного тексту.

Сучасні технології дозволяють автоматизувати процеси, необхідною умовою яких раніше була наявність спеціаліста, що відбирав дані і вручну проводив попередню обробку для подальшого аналізу і побудови моделей прогнозування.

Не є виключенням і аналіз відгуків споживачів супермаркетів, що є необхідною частиною маркетингового аналізу з метою поліпшення якості обслуговування клієнтів. Оскільки це відповідає саме задачам класифікації та прогнозу, актуальним є використання різних методів обробки природної мови, що відноситься до машинного навчання. Для вирішення задачі класифікації та прогнозування виконуються процес, що містить два етапи:

створення моделі на навчальній вибірці та використання цієї моделі для передбачення значення залежної змінної на нових даних.

Так, як сучасні інформаційні системи працюють в умовах великих обсягів даних, необхідною умовою є використання ефективних алгоритмів аналізу та прогнозування. Також ці алгоритми мають давати точний результат. В області NLP (natural language processing) на сьогодні одним з найефективніших методів є використання нейронних мереж-трансформерів. Цей метод заснований на тонкій настройці (fine-tuning) моделі за допомогою мережі-трансформера і подальшому використанні класифікатора, який здійснює аналіз тексту.

Таким чином завдання розробки моделі вдосконалення голосового керування обробки відгуків клієнтів мережі супермаркетів з використанням методів Natural Language Processing є сучасним та актуальним.

Об’єкт дослідження: процес керування обробки відгуків клієнтів мережі супермаркетів.

Предмет дослідження: методи аналізу тексту з використанням мереж-трансформерів.

Мета роботи: вдосконалення існуючих алгоритмів голосового керування обробки відгуків клієнтів мережі супермаркетів з використанням моделей ULMFiT (Universal Language Model Fine-tuning for Text Classification), BERT (Bidirectional Encoder Representations from Transformers) та XLNet.

Методи дослідження, які були використані під час роботи включають в себе теоретичне дослідження трьох моделей: XLNet, BERT та ULMFiT для розуміння актуальних досягнень у досліджуваному напрямку, а також практичне дослідження у вигляді експериментів з цими моделями для аналізу та порівняння їх придатності до використання у аналізі тексту, а саме класифікації відгуків споживачів товарів мережі супермаркетів.

Задачами дослідження є проведення експериментів спрямованих на виявлення моделі з найбільшим показником точності результату на наборі даних, що містить відгуки користувачів мережі супермаркетів, а також розробка веб-застосунку, що надаватиме можливість побачити результати

аналізу довільного відгуку за допомогою трьох моделей.

Практичне значення одержаних результатів підтверджує придатність розглянутих моделей для вирішення задачі багатокласової класифікації для набору даних, що складається з відгуків про товари.

Науковий результат: Оптимізація обробки відгуків клієнтів мережі супермаркетів з використанням методів Natural Language Processing.

Апробація та публікації:

1. Аверічев І.М., Шведченко І.С. Аналіз моделей придатності для класифікації відгуків клієнтів мережі супермаркетів з використанням методів Natural Language Processing // XV Науково-технічна конференція «Сучасні інфокомунікаційні технології» – Київ: ДУТ, 2022.
2. Аверічев І.М., Шведченко І.С. Оптимізація обробки відгуків клієнтів мережі супермаркетів з використанням методів Natural Language Processing // III Науково-практичній конференції «Проблеми комп’ютерної інженерії» – Київ: ДУТ, 2022.

1 ОГЛЯД НАУКОВОЇ І ПАТЕНТНОЇ ЛІТЕРАТУРИ

1.1 Відгуки та їх вплив на роботу підприємства

Основним джерелом отримання інформації про бізнес є відгуки клієнтів. Перед тим як користуватися послугою чи продуктом всі шукають рекомендації та відгуки [1].

Оцінка підприємства клієнтами насправді робить набагато більше, ніж просто закріплює довіру та статус. Відгуки також впливають на загальний дохід компанії та продуктивність продажів [1].

Саме через це підприємство та його оточення збирає і аналізує будь-які дані, що мають як позитивний, так і негативний акцент.

Портал brightlocal.com оприлюднив результати дослідження про залежність наявності відгуків про підприємство та відношення до них зі сторони клієнтів [1]:

- 86% споживачів читають відгуки про місцеві підприємства (в тому числі 95% людей у віці від 18 до 34 років);
- споживачі читають в середньому 10 відгуків, перш ніж відчувають, що можуть довіряти місцевому підприємству;
- 40% споживачів беруть до уваги тільки відгуки, написані за останні 2 тижні;
- 57% споживачів використовуватимуть підприємство, тільки якщо у нього 4 або більше зірок;
- 80% людей у віці від 18 до 34 років написали відгуки – в порівнянні з 41% споживачів старше 55 років;
- 91% споживачів у віці від 18 до 34 років довіряють відгукам стільки ж, скільки особистим рекомендаціям;
- 89% споживачів читають відповіді підприємств на відгуки.

За даними Digital.com [3] 40% користувачів в середньому читають 7 коментарів перед покупкою (рис. 1.1).

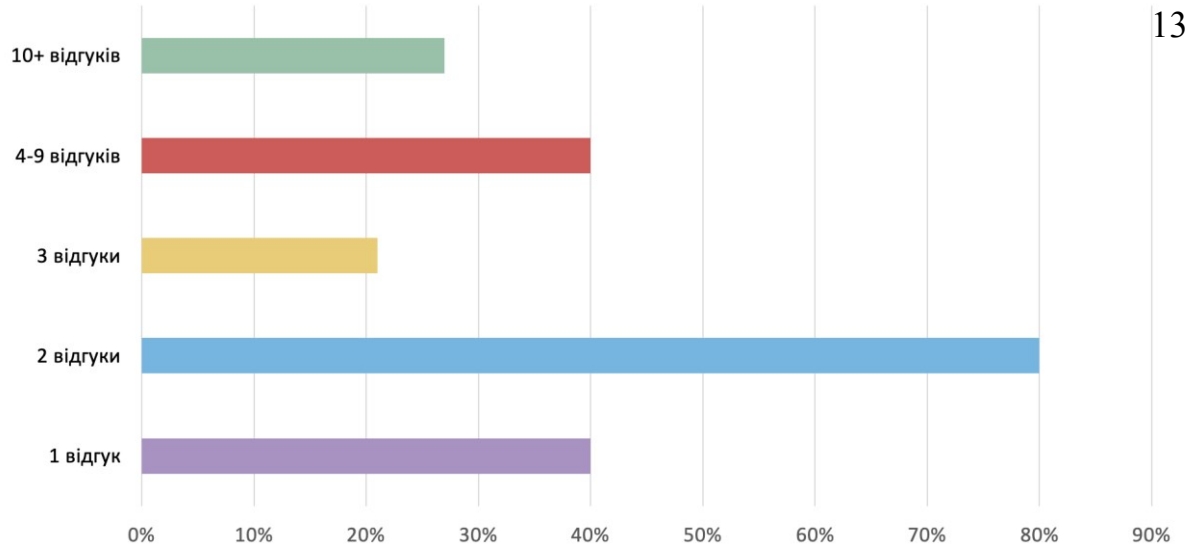

Рис. 1 – Кількість відгуків, які читають користувачі перед покупкою [3]

Якщо користувач обирає серед декількох товарів, він має ознайомитись із відгуками до кожного з них. Звичайно, що не кожен користувач має досить вільного часу, аби приділити його ознайомленню із усіма відгуками, які залишили покупці. Тому зазвичай коментарі читають для того, аби по декільком із них зрозуміти, чи в реальності відповідає товар його опису та чи справжній його рейтинг [1].

Відзначимо, що коментар чи відгуки майже не пишуться, якщо придбаний товар або послуга знаходиться в рамках можливих очікувань, тобто, коли користувача все влаштовує. Реакція з'являється лише в тих випадках, коли товар або послуга знаходяться в крайніх або близько до них положеннях, тобто якщо такий продукт або вразив, або засмутив [1].

Так само має місце і така ситуація, коли відгуки є елементами протистояння, з метою знизити імідж свого конкурента. Такий негатив дуже важко нівелювати, якщо дана ситуація не відстежується або на неї немає своєчасної реакції [1].

Також можливий варіант, коли позитивні відгуки «накручують», тобто постачальник послуги або товару самостійно покращує свій рейтинг на даних сервісах [1].

Таким чином система «відгуків» є частиною системи управління репутацією підприємства, яка крім іншого має на увазі і комплекс заходів спрямованих на:

- усунення негативу і його впливу;
- наявність постійного інформаційного каналу в основі якого є позитивна інформація про підприємство, його товар та бренд [1].

Безумовно, система відгуків, має важливе аналітичне значення, яка дозволяє крім усього іншого отримати реальну оцінку втрат в грошовому вираженні. Сервіси Play Store та App Store мають систему оцінки втрат, яка виражається даною формулою [1]:

$$LossR = (5 - St) \times 0.07 \times Мот, \quad (1)$$

де LossR – збитки низької кількості зірок;
St – кількість зірок, яка є у додатку;

Мот – значимість майданчика з відгуками. Значимість визначається діленням загальної кількості відгуків з окремого сайту на сумарну кількість відгуків про підприємство. Для мобільних додатків значимість знаходиться на рівні від 96 до 100 відсотків.

Таким чином, одна «зірка» «коштує» інвесторам і розробникам від 5% до 9%.

Ще один аспект який має значення в наш час, частота, і безперервність відгуків. У дослідженні Local Consumer Review Survey [1] було виявлено кілька ключових результатів, які важливі в ранжируванні:

- 77% споживачів не довіряють онлайн відгуками, яким понад три місяці;
- 18% опитаних не довіряють відгуками, яким понад два тижні;
- 4% беруть до уваги відгуки, яким понад рік.

Наведені вище дані дозволяють стверджувати, що робота з відгуками, причому як з позитивними, так і з негативними, дозволяє завоювати довіру покупців, а наявність продуманої стратегії з аналізу та управління дозволяє підприємству отримувати додатковий фінансовий результат.

1.2 Аналіз ринку та існуючих шляхів вирішення проблеми

Сьогодні існує дуже багато сервісів, які дозволяють користувачам оцінити роботу підприємств, закладів громадського харчування, інтернет-магазинів. Орієнтуючись на рейтинги, користувачі вирішують, де провести свій вільний час та придбати ті чи інші товари. Відгуки дають користувачам інформацію про плюси та мінуси конкретного підприємства, специфіку роботи та допомагають сформувати попереднє враження про підприємство та вирішити, чи підходить воно користувачу [1].

Ціль таких сервісів – відображення актуальної та правдивої інформації про підприємства та надання клієнтам можливості самим оцінювати їх роботу. Під час дослідження ринку, знайдено існуючі рішення, які здатні надати користувачам схожі можливості. Нижче наведено детальний опис схожих систем [1].

Otzovyk – це споживчий портал, створений для зручного пошуку та додавання відгуків про підприємства України [3].

Переваги:

- велика кількість українських підприємств, користувачів та відгуків на сервісі;
- можливість додавання підприємств в список будь-яким користувачем для написання відгуків;
- наявність цілодобової технічної підтримки на сайті;
- наявність категорій підприємств;
- автоматичне видалення відгуків з нецензурним змістом;
- можливість додавання графічних зображень до відгуків.

Недоліки:

– відсутність мобільного додатку;

– можливість залишати відгуки без реєстрації користувача, що дає можливість створення фейкових відгуків [3].

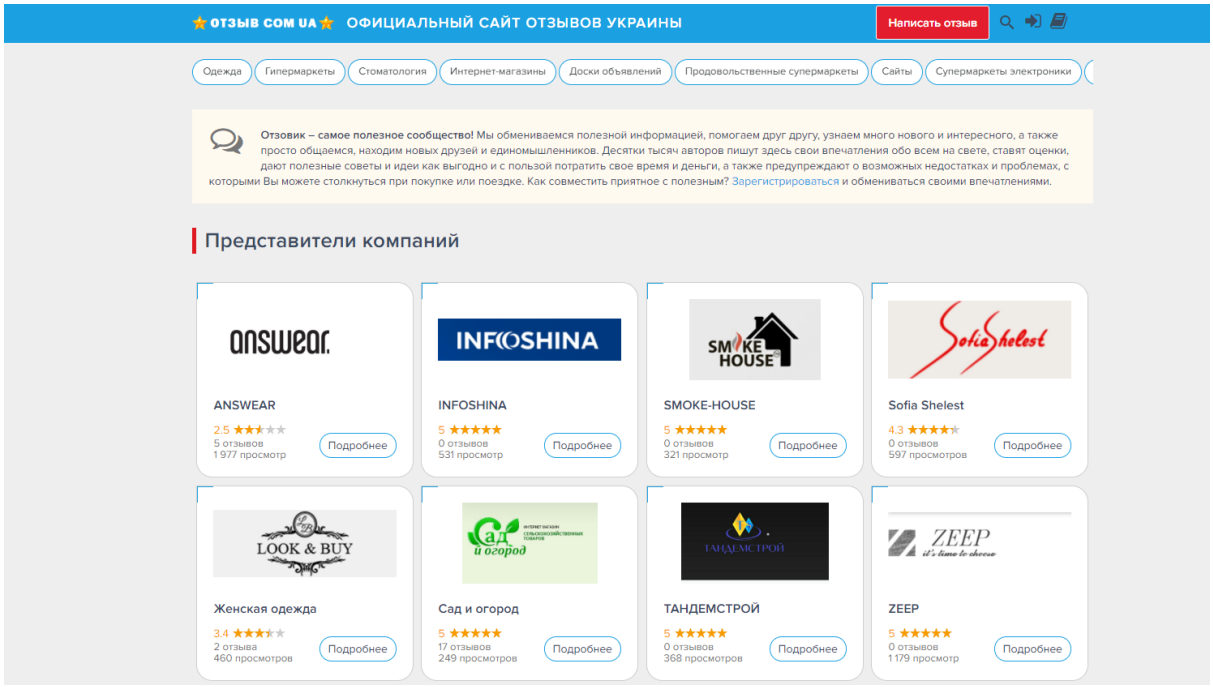

Рис. 2 – Головна сторінка порталу otzovyk.com.ua

Otzyvua – український сайт відгуків, який надає можливість користувачам публікувати власні відгуки про українські підприємства та задавати питання представникам компанії на сайті [4].

Переваги:

– наявність віджета для додавання інструменту на сторонні сайти;

– велика кількість українських підприємств, користувачів та відгуків на сервісі;

– наявність представників компанії на сайті, які мають можливість відповідати на відгуки користувачів;

– автоматичне видалення відгуків з нецензурним змістом;

– можливість додавання графічних зображень та відео до відгуків;

– наявність категорій підприємств.

Недоліки:

– відсутність мобільного додатку;

– можливість написання відгуків лише для тих підприємств, які вже додані на сайті [4].

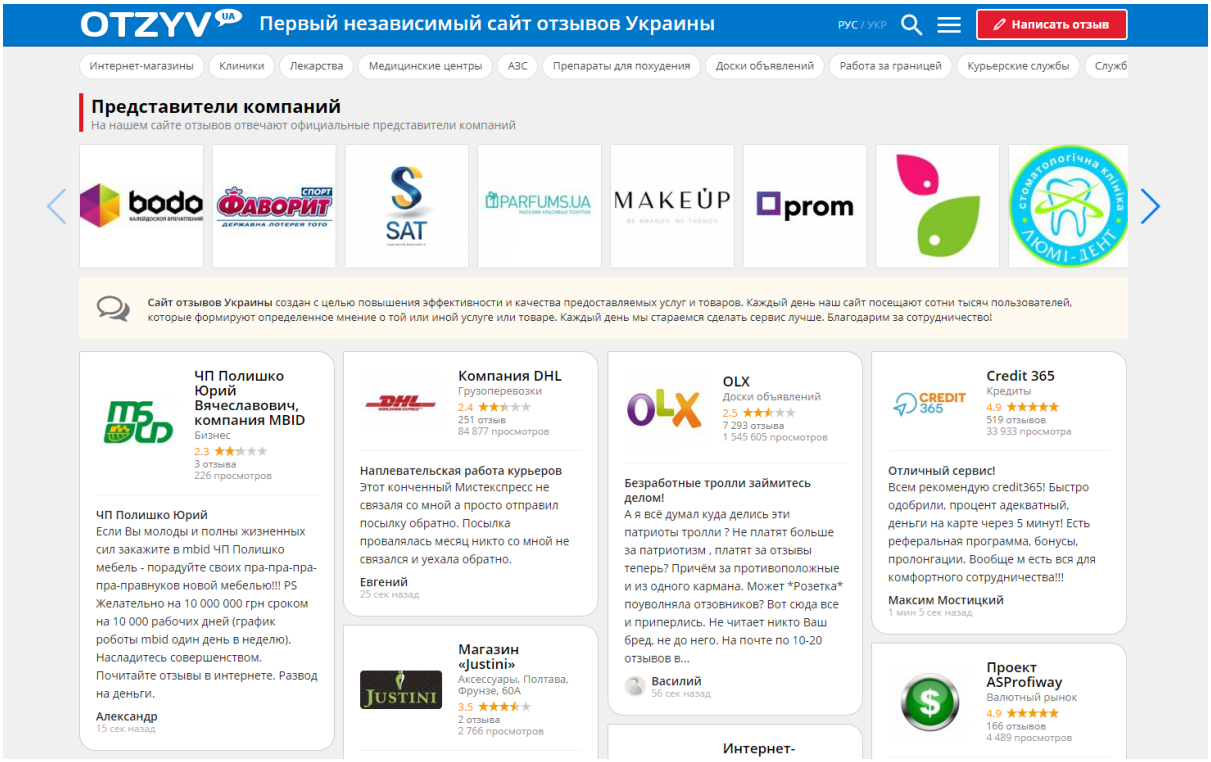

Рис. 3 – Головна сторінка порталу otzyvua.net

Trustpilot – безкоштовна, відкрита платформа оглядів заснована у 2007 році у Данії, на якій розміщені огляди підприємств по всьому світу [5]. За словами Пітера Гольтена Мюльмана, засновника і генерального директора: “Для користувачів ми є місцем для зв’язку та впливу на бізнес, а для бізнесу – платформою для прогресу та способом удосконалення та інновацій шляхом співпраці зі споживачами” [6].

Переваги:

– наявність категорій підприємств;

– широкі можливості для бізнесу;

– наявність веб-додатку для браузера Chrome, для зручної публікації відгуків без відвідування веб-сайту.

Недоліки:

– величезна ціна за додаткові функції для бізнесу, що обмежує можливість використання даних функцій малим бізнесом;

– відсутність мобільного додатку [6].

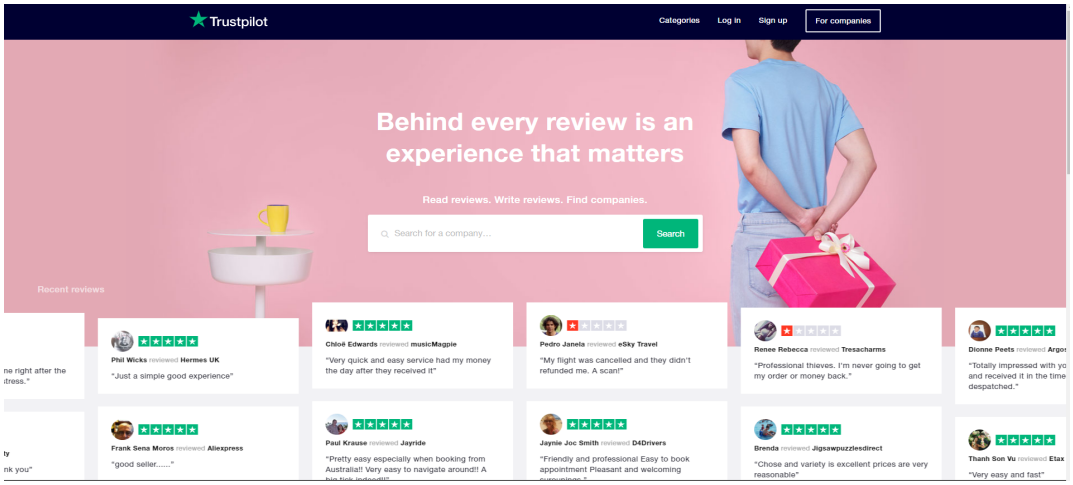

Рис. 4 – Головна сторінка порталу trustpilot

DOU.ua – велика платформа, що має на меті розповсюджувати новини, аналітичні статті та відгуки про компанії, пов'язані з інформаційними технологіями. Веб-сервіс налічує близько 400 000 користувачів та 9 млн. переглядів сторінок у місяць [7].

Переваги:

– це найбільша платформа, на якій розміщено майже всі топ компанії, що так чи інакше відносяться до інформаційних технологій;

– рейтинг, що формується за допомогою отриманих балів від працівників підприємства та клієнтів;

– сайт надає можливість переглянути реальні фото компанії та умови, які вони пропонують;

– більшість (якщо не всі) компанії, які розміщені на платформі, залишають посилання на сайт dou на власних офіційних сторінках, що тільки додає впевненості у достовірності інформації;

– можливість створити сторінку для власної компанії на сайті, якщо вона відповідає вимогам.

Недоліки:

– веб-сайт призначений не тільки для відгуків про компанії, що значно погіршує досвід користувача;

– на платформі розміщені тільки компанії, які відносяться до інформаційних технологій;

– відсутність мобільного додатку [7].

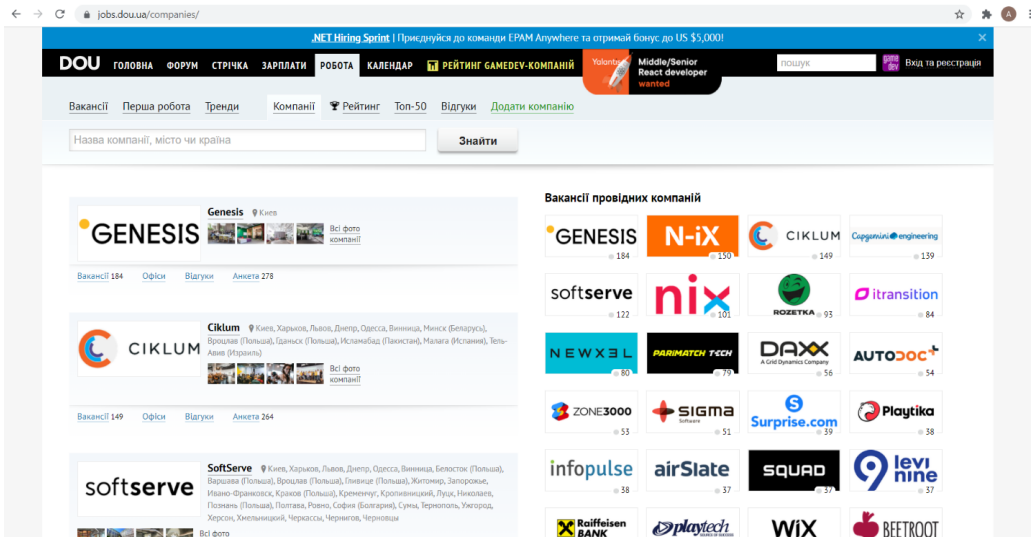

Рис. 5 – Сторінка з компаніями на dou.ua

1.3 Опис обробки природної мови

Обробка природної мови (англ. Natural-language processing, NLP) — загальний напрям інформатики, штучного інтелекту та математичної лінгвістики. Він вивчає проблеми комп'ютерного аналізу та синтезу природної мови. Стосовно штучного інтелекту аналіз означає розуміння мови, а синтез — генерацію розумного тексту. Розв'язок цих проблем буде означати створення зручнішої форми взаємодії комп'ютера та людини [4].

На думку дослідниці в галузі штучного інтелекту (ШІ) Елізабет Лідді, обробка природної мови — це комп'ютеризований підхід до аналізу тексту, що базується на низці теорій та наборі технологій. Ця галузь не має одного загально прийнятого визначення, адже вона перебуває у стані постійних досліджень та розробок. Однак, існують певні аспекти, які б об'єднували усі існуючі визначення [1].

Можна визначити обробку природної мови як міждисциплінарна галузь науки, що охоплює методики обчислювальної лінгвістики татеорії ШІ, основною метою якої є забезпечення вербальної та невербальної комунікації між людьми та комп'ютерними системами [9].

1.4 Історія обробки природних мов (NLP)

Історія обробки природних мов почалася в 30-х роках XX сторіччя з наукових праць на тему машинного перекладу тексту таких авторів, як Жорж Артсруні та Петро Троянський. Їх праці були пов'язані з двостороннім перекладом між двома мовами та базовими правилами роботи з граматичними ролями [10].

Обробка природної мови сягає корінням у 1950-ті роки. Вже в 1950 році Алан Тьюрінг опублікував статтю під назвою "Обчислювальна техніка та інтелект", в якій запропонував те, що зараз називається тестом Тьюрінга, як критерій інтелекту, завдання, яке включає автоматизовану інтерпретацію та генерацію природної мови, але на той час не сформульоване як проблема, окремо від штучного інтелекту [12].

Передумова виникнення NLP добре узагальнена експериментом Джона Серла з китайською кімнатою отримавши збірник правил (наприклад, китайський розмовник, із запитаннями та відповідями), комп'ютер імітує розуміння природної мови (або інші завдання NLP), застосовуючи надані правила до даних з якими стикається [13].

Деякими особливо успішними системами обробки природних мов, розробленими в 1960-х роках, були SHRDLU , система природних мов, що працює в обмежених "блокових світах" (алгоритм планування доменів в ШІ) із обмеженими словниками, та ELIZA , імітація "рогерійського психотерапевта", написана Джозефом Вайценбаумом між 1964 і 1966 роками.[4] Не використовуючи майже жодної інформації про людські думки чи емоції, ELIZA іноді забезпечувала вражаючу людську взаємодію.

Протягом 1970-х років багато програмістів почали писати "концептуальні онтології", які структурували реальну інформацію в зрозумілі для комп'ютера дані. Прикладами є MARGIE (Schank, 1975), SAM (Cullingford, 1978), PAM (Wilensky, 1978), TaleSpin (Meehan, 1976), QUALM (Lehnert, 1977), Politics (Carbonell, 1979) і Plot Units (Lehnert 1981). У цей час було написано перші чат-боти (наприклад, PARRY) [13].

1980-ті і початок 1990-х років відзначають розквіт природної обробки мови. Основними напрямками були дослідження синтаксичного аналізу на основі правил, морфології, семантики, посилань та інших областей розуміння природної мови. Інші напрямки досліджень були продовжені, наприклад, розвиток з віртуальним співрозмовником Racter і Jabberwacky [14].

До 1980-х років більшість систем обробки природних мов базувались на складних наборах рукописних правил. Однак, починаючи з кінця 1980-х, відбулася революція в обробці природної мови із запровадженням алгоритмів машинного навчання для обробки мови. Це було обумовлено постійним збільшенням обчислювальної потужності, так і поступовим зменшенням домінування Хомських теорій лінгвістики (трансформаційна граматика), теоретичні основи яких знеохочували той тип корпусної лінгвістики, який лежить в основі підходу до машинного навчання до мовної обробки [14].

Багато помітних ранніх успіхів щодо статистичних методів у NLP відбулися в галузі машинного перекладу, особливо завдяки роботі IBM Research. Ці системи змогли скористатися перевагами існуючих багатомовних текстових корпусів, вироблених парламентом Канади та Європейським Союзом як результат законів, що передбачають переклад усіх урядових проваджень на всі офіційні мови відповідних систем управління [14].

Із зростанням Інтернету з середини 1990-х років стає доступним зростаючий обсяг необроблених мовних даних. Таким чином, дослідження все більше зосереджуються на алгоритмах навчання без нагляду та напівконтролю.

Як правило, це завдання набагато складніше, ніж навчання під контролем, і, як правило, дає менш точні результати для заданого обсягу вхідних даних. Однак існує величезна кількість анотованих даних (включаючи, серед іншого, весь вміст Всесвітньої павутини), що часто може компенсувати гірші результати, якщо використовуваний алгоритм має досить низьку часову складність, щоб бути практичним [15].

У 2010-х роках представницьке навчання та методи машинного навчання в стилі глибоких нейронних мереж набули широкого поширення в обробці природних мов, частково завдяки безлічі результатів, які показують, що такі методи можуть досягти найсучасніших результатів у багатьох завданнях з природної мови, наприклад, у моделюванні мови, синтаксичному аналізі, та багатьох інших завданнях [15].

Це стає все більш важливим у медицині та охороні здоров'я, де NLP використовується для аналізу приміток та тексту в електронних медичних картrах, які в іншому випадку були б недоступні для вивчення під час вдосконалення медичної допомоги [6].

Сучасні тенденції в обробці спостерігаються в напрямку використання алгоритмів машинного навчання без вчителя або з частковим наглядом. Цей спосіб навчання моделей є більш складним, ніж навчання з вчителем на розмічених вхідних даних, однак через те що ж можливість використовувати нескінченну кількість даних для навчання з відкритих джерел, він є більш перспективним [6].

На відміну від навчання з вчителем на певному наборі даних, це дає змогу створювати більш загальновживані моделі [6].

1.5 Огляд актуальних наукових робіт

Сучасні тенденції розвитку обробки природних мов розвиваються у напрямку машинного навчання без вчителя, що підтверджується останніми науковими роботами з цієї галузі. Далі будемо розглядати три роботи, які пов’язані з глибинним навчанням: ULMFiT, BERT, XLNet.

1.5.1 ULMFit

ULMFiT (Universal Language Model Fine-tuning for Text Classification, універсальна мовна модель точного налаштування для класифікації тексту) – це метод трансферного навчання, що дозволяє ефективно проводити попереднє тренування моделі для задач обробки природних мов. Індуктивне трансферне навчання мало великий вплив на комп'ютерний зір (CV), де прикладні моделі для виявлення, класифікації та сегментації об'єктів не створюють з чистого аркуша, а налагоджують та підстроюють існуючі моделі, що були розроблені на загальновживаних наборах даних, таких як ImageNet, MS-COCO та інших [8].

До розробки цього методу використовувалися підходи глибинного навчання, коли розробка моделей проводилася з нуля, що потребувало багато часу і ресурсів та великі набори даних для навчання [17].

Дослідження в NLP були зосереджені переважно на трансдуктивному трансфері. Індуктивний трансфер для тонкої настройки попередньо підготовлених вбудовування слів (ембедингів) – проста техніка трансферу, яка націлена лише на перший шар моделі, мала великий вплив на практику і використовується в більшості найсучасніших моделей [17].

Підходи, розроблені останнім часом, що поєднують ембединги, похідні з інших прикладних задач із введенням даних у різні шари, навчають основну модель для конкретної задачі з нуля, що обмежує корисність цих ембедингів [9].

Таким чином, дослідження ULMFiT спрямоване на перенесення підходів, використаних в області задач комп’ютерного зору, в область обробки природної мови і розробці нових практик налагодження моделей під ці задачі [17].

ULMFiT складається з трьох етапів (див. рис. 6):

– мовна модель навчається на текстах загальної області знань для захоплення загальних особливостей мови в різних шарах;

– повна мовна модель адаптується на даних для цільової задачі, використовуючи дискримінаційну настройку та швидкість навчання за трикутною формою для вивчення певних особливостей завдання;

– класифікатор налаштовується на цільову задачу, використовуючи поступове розморожування для того, щоб зберегти базові шари моделі та адаптувати верхні шари під конкретну задачу [17].

Цей метод є універсальним, так як він:

– працює для різних задач, відмінних розміром, типами та кількістю міток;

– використовує один архітектурний та навчальний процес;

– не потребує ручної розробки та попередньої обробки ознак;

– не потребує специфічних для домену документів та міток

У роботі над ULMFiT автори також використали для налагодження коефіцієнтів моделі метод дискримінаційної тонкої настройки. Формулу цієї настройки коефіцієнтів наведено нижче [18].

$$\theta^{!} = \theta^{!} - \eta^{!} \cdot \nabla_{\text{m}!}(\theta), \tag{1}$$

де $\theta^{!}$ –параметри моделі, η – рейт навчання, $\nabla\theta\, J\,(\theta)$ – градієнт залежно від цільової функції моделі.

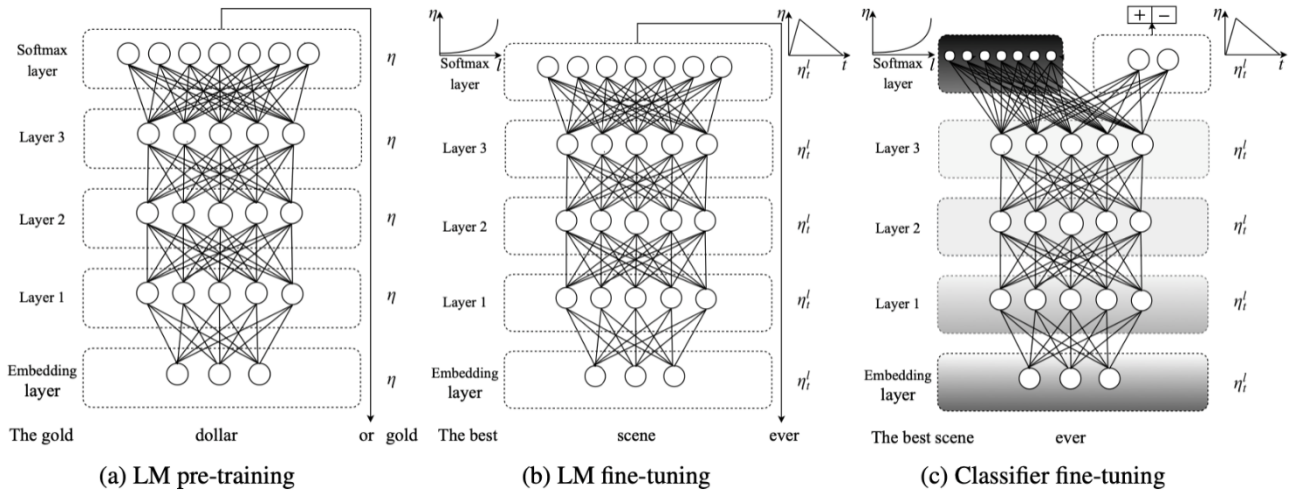

Рисунок 6 – Етапи методу ULMFiT

Для контролю коефіцієнта навчання використовується STLR (slanted triangular learning rates). Для адаптації параметрів моделі до особливостей поточної задачі модель на початку навчання швидко переходить до певної області параметрів, а потім лише підлаштовує ці параметри [18].

При використанні одного коефіцієнта навчання (learning rate) протягом всього процесу навчання не підходить для забезпечення такої поведінки [10]. У цьому випадку підходять STLR, які спочатку лінійно збільшують коефіцієнт навчання, а потім лінійно зменшують його (див. рис. 7).

Для тонкої настройки класифікатора, мовна модель, отримана з попереднього навчання, доповнюється двома додатковими лінійними блоками. Аналогічно стандартним практикам для класифікаторів з області комп’ютерного зору, кожен блок використовує пакетну нормалізацію та випадання, з функцією активації ReLU для проміжного шару та функцією активації softmax, що дозволяє отримати ймовірний розподіл на цільові класи в останньому шарі. Перший лінійний шар приймає на вхід об'єднаний останній прихований шар. Єдиними шарами, навчання яких проводиться з чистого аркуша є лише шари специфічні для конкретної задачі [18].

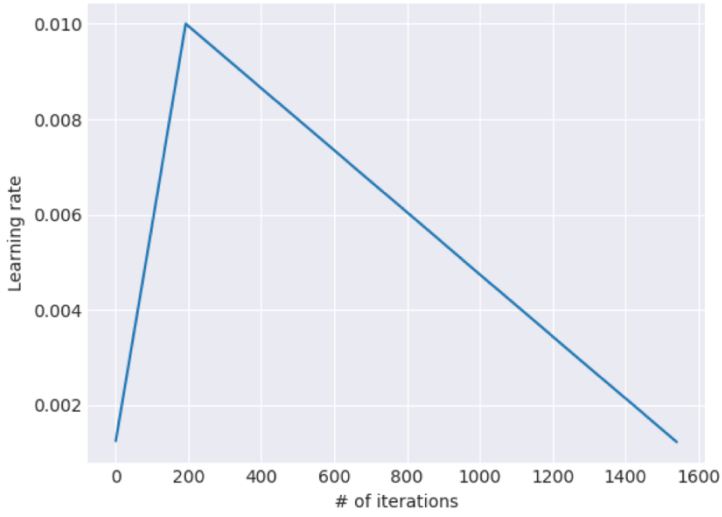

Рисунок 7 – Похилі трикутні коефіцієнти навчання

Останнім етапом тонкої настройки моделі є поступове розморожування шарів. Якщо налаштовувати усі шари одразу, існує великий ризик загубити результати навчання базових шарів. Щоб цього не було, налаштування відбувається поступово, починаючи з останнього, який відповідає найменш загальним знанням. В кожній епосі розморожується один шар і усі розморожені шари налагоджуються до зближення на останній ітерації [11].

Метод було протестовано на шістьох широко вивчених наборах даних, з різною кількістю документів та їх розміром, що використовуються в сучасній класифікації тексту. В якості задач класифікації було обрано класифікацію тематики, сентимент-аналіз та класифікація питань. Порівняння рівнів помилки наведено на рисунку 8.

	Model	Test
IMDb	CoVe (McCann et al., 2017)	8.2
	oh-LSTM (Johnson and Zhang, 2016)	5.9
	Virtual (Miyato et al., 2016)	5.9
	ULMFiT (ours)	**4.6**

Рисунок 8 – Порівняння рівня помилки ULMFiT з іншими моделями

В результаті, отриманий метод тонкої настройки моделі для задач NLP демонструє кращий результат, ніж попередні роботи, з рівнем помилки меншим на 18-24 відсотки на більших об’ємах даних [12].

На Рис. 9А зображено Схематичний вид агента навчання із підкріпленням, призначеного для синтаксичного аналізу. Середовище обробки мови надаватиме агенту стану та нагороди після кожної взаємодії. Функція винагороди може бути визначена різними способами, наприклад, позитивна винагорода у розмірі 10 може надаватися щоразу, коли застосовується відповідне правило граматики.

Рис.9А – Схематичний вигляд агента навчання з підкріпленням Розробленого для обробки мови

Середовище обробки мови надасть агенту стани та винагороди після кожної взаємодії. Функція винагороди визначається конкретним завданням обробки природної мови. Одна проста можливість для функції винагороди посилить кожну оптимальну дію за допомогою +1.

На Рис.9Б зображено Схематичний вигляд агента навчання з підкріпленням, розробленого для обробки мови. Агент мовної моделі діє шляхом додавання, заміни або видалення рядків слів. Стани - це рядки слів. Середовище обробки мови надасть агенту стани та винагороди після кожної взаємодії. Функція винагороди визначається конкретним завданням обробки природної мови. Одна проста можливість для функції винагороди посилить кожну оптимальну дію за допомогою +1.

Рис.9Г – Схематичний вигляд агента навчання з підкріпленням, призначеного для генерації мови як приклад програми

Рис. 9В зображено Схематичний вид агента навчання з підкріпленням, призначеного для розуміння тексту, як приклад програми. Агент мовної моделі діє, присвоюючи значення вектору чи списку змінних залежно від висловлювання користувача. Поточний користувач user02 обчислюється з висловлювання користувача. Наступний стан — це рядок, згенерований із вектора змінних у розумінні агента, наприклад, за допомогою агента-генератора тексту (див. рис. 9В). Середовище обробки мови надасть агенту стану та нагороди після кожної взаємодії. Середовище та функція винагороди визначаються розв'язуваним завданням розуміння мови, тобто інфобот

На Рис.9Б зображено Схематичний вигляд агента навчання з підкріпленням, розробленого для обробки мови. Агент мовної моделі діє шляхом додавання, заміни або видалення рядків слів. Стани - це рядки слів. Середовище обробки мови надасть агенту стани та винагороди після кожної взаємодії. Функція винагороди визначається конкретним завданням обробки природної мови. Одна проста можливість для функції винагороди посилить кожну оптимальну дію за допомогою +1.

Схематичний вигляд агента навчання з підкріпленням

На Рис. 10 наведено Інформаційний потік розмовної системи. Ця система отримує на вхід текстовий рядок, що містить питання або просто коментар, і відповідає іншим текстовим рядком, що містить відповідь. Ця взаємодія введення та відповіді зазвичай повторюється кілька разів. Для переходу від «Вступного текстового рядка x» до «Вихідного рядка відповіді x» потрібна послідовна програма агента розпізнавання тексту та агента генератора тексту.

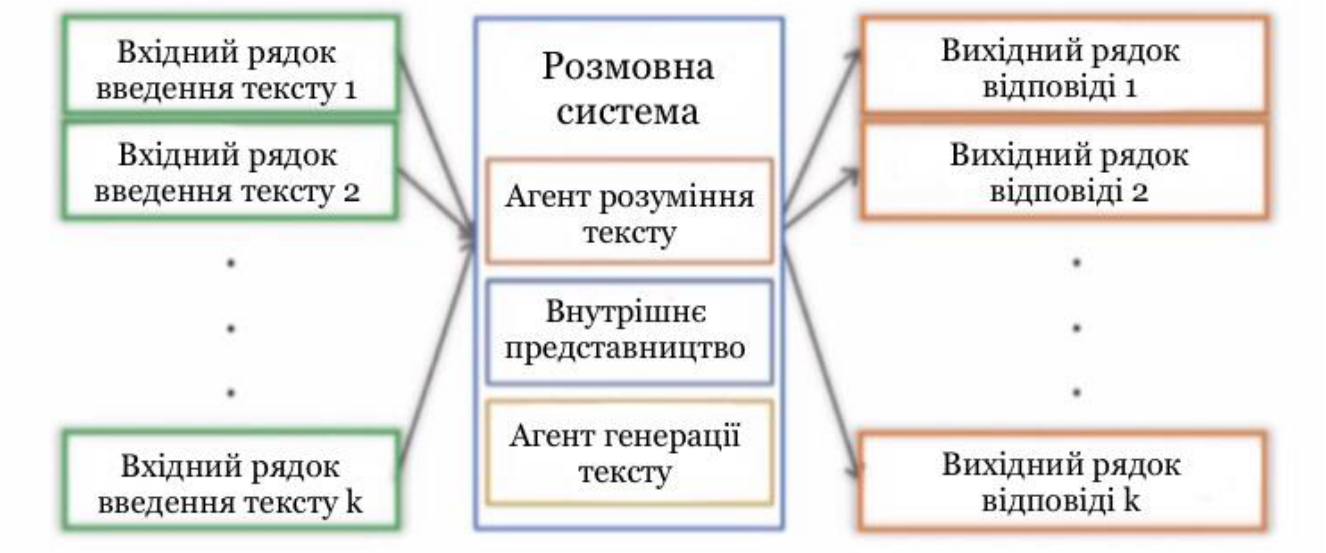

Рис.10 - Інформаційний потік розмовної системи

Розуміння мови можна також представити як MDP, і тому ми можемо застосовувати складні алгоритми навчання з підкріпленням, розроблені в останні роки. Крім того, ми можемо реалізувати їх разом з глибокими нейронними мережами, щоб впоратися з величезним обсягом даних, які зазвичай потрібні для розпізнавання тексту.

Розглянемо проблему розуміння природної мови (NLU). У такому завданні у нас може бути граматика, подібна до наведеної на рис. 5, яка дозволяє програмі автоматично визначати елементи речення, написаного англійською мовою. Використовуючи цю граматику, такі пропозиції, як «Покупець зі знижкою хоче відшкодування» та «Покупець зі знижкою скасував відшкодування», можуть бути проаналізовані автоматизованою системою для визначення наміру покупця, який у даному випадку полягає в тому, чи хоче він хоче відшкодування, чи вона хоче скасувати повернення, яке вона раніше запросила.

Таким чином, граматика може використовуватися для визначення намірів користувачів, а навчання з підкріпленням може використовуватися для вибору оптимальної послідовності замін у процесі аналізу вхідних пропозицій. Після того, як програма синтаксичного аналізу була використана для визначення граматичної ролі кожного слова у вхідному текстовому рядку, результат може бути збережений у структурі векторного типу, такий як [хто = користувач 02, намір = "хоче", зміст = "знижка"]. векторне уявлення змінних, які, намір і зміст, можуть бути використані іншою програмою для визначення найбільш відповідної дії, яка буде виконана наступним: Наприклад, інформування покупця про знижку.

1.5.2 BERT

BERT (Bidirectional Encoder Representations from Transformers) – мовна модель, розроблена корпорацією Google як продовження ідеї використання нейронних мереж з архітектурою «трансформер» для передтренування мовних моделей на великих обсягах текстових даних, що давало значну перевагу у виконанні задач обробки природних мов, порівняно з попередніми методами [13].

Вперше цю архітектуру було використано у OpenAI для задач обробки природних мов. Розроблена мережа називається GPT [13] і є однонаправленою, на

відміну від BERT, яка є двонаправленою. Модель від OpenAI мала деякий ряд недоліків, що наведені нижче [13]:

– великі вимоги до обчислень: підхід вимагає дорогого етапу попередньої підготовки – 1 місяць на 8 графічних процесорах. Порівняно з іншими роботами, ця модель є значно більшою і використовує більше обчислень та пам'яті (37-шарова (12 блокова) архітектура трансформерів і використовує послідовності до 512 лексем для тренування;
– обмеженість та упередженість пізнання: текст, наявний в Інтернеті, не містить точних та повних даних про світ. Через що, модель не може повноцінно навчатися адекватно цим даним, оскільки нерівномірний розподіл даних призводить до того, що модель починає його експлуатувати;
– крихке узагальнення: GPT має покращену ефективність у широкому спектрі задач, але поступається сучасним моделям глибинного навчання у сфері обробки природних мов в умовах систематичної, змагальної (adversarial) або поза-розподільної оцінки.

BERT, на відміну від GPT, має здатність навчати мовні моделі на основі всього набору слів, тобто виконує двонаправлене навчання (див. рис. 9). В GPT навчання виконується класичним способом в одному напрямку (зліва направо, чи справа наліво) [13].

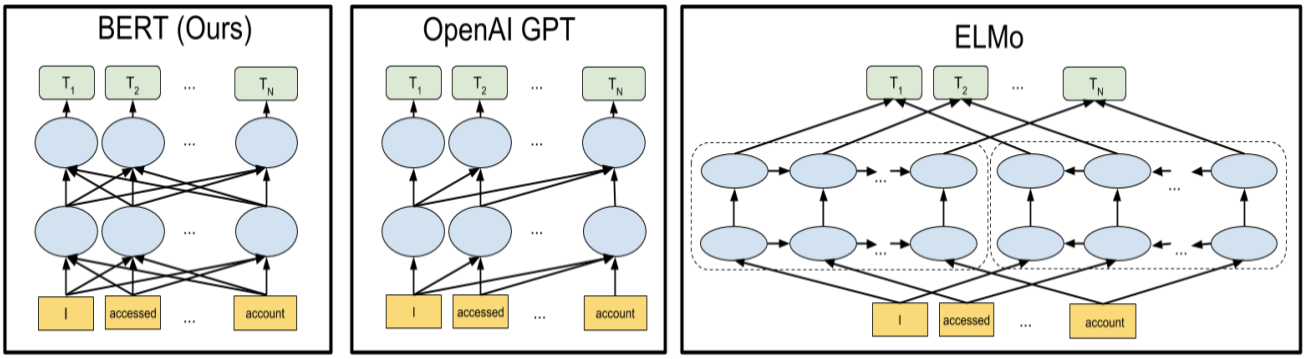

Рисунок 9 – Порівняння архітектури BERT, GPT та ELMo

BERT дозволяє мовній моделі розуміти контекст слова, базуючись на оточуючих словах, при чому не тільки безпосередньо сусідніх [14]. Наприклад, однонаправлена модель у реченні «Я отримав доступ до акаунта в банку» буде співставляти слово «акаунт» з «я отримав доступ до», але не буде брати до уваги слово «в банку», що не дає повного розуміння запиту. BERT у цьому випадку буде співставляти слово «акаунт» з цілим контекстом – «я отримав доступ до … в банку», що дає можливість знати повний контекст і сутність цього запиту [13].

BERT може вчитися моделювати зв’язки між реченнями у тексті, попередньо навчаючись на простих задачах, які можуть бути сформовані з будь-якого тексту. Наприклад, така задача може формулюватися наступним чином: розглядаючи два речення A і B, чи є B фактичним продовженням попереднього речення A, чи є випадковим реченням у тексті і не пов’язано з A [13]. Приклад наведено на рисунку 10.

Рисунок 10 – Приклад розпізнавання зв’язків між реченнями

Для розуміння основної складової моделі BERT далі описано роботу мережі за архітектурою «трансформер». Приклад наведено на рисунку 11.

Мережа в такому вигляді складається з енкодеру з шаром багатоголосової уваги (multi-head attention) та декодеру [13].

Енкодер отримує на вхід слова і видає метаданні, що відповідають цим словам для подальшого використання у декодері.

Кожне слово паралельно проходить через шари енкодеру, де частина – повнозв’язні шари (fully-connected layers), а деякі – скорочені (shortcut connections) [13].

Елементом, який відрізняє цю архітектуру від згорткової нейронної мережі або рекурентної нейронної мережі є шар багатоголосової уваги (mutli-head attention). Цей шар дає можливість кожному вхідному вектору взаємодіяти з

іншими через механізм уваги (attention mechanism), а не використовувати схований стан (hidden state), як це відбувається в RNN (recursive neural network) або сусідніх слів в CNN (convolutional neural network) [13]. Структура цього шару наведена на рисунку 12.

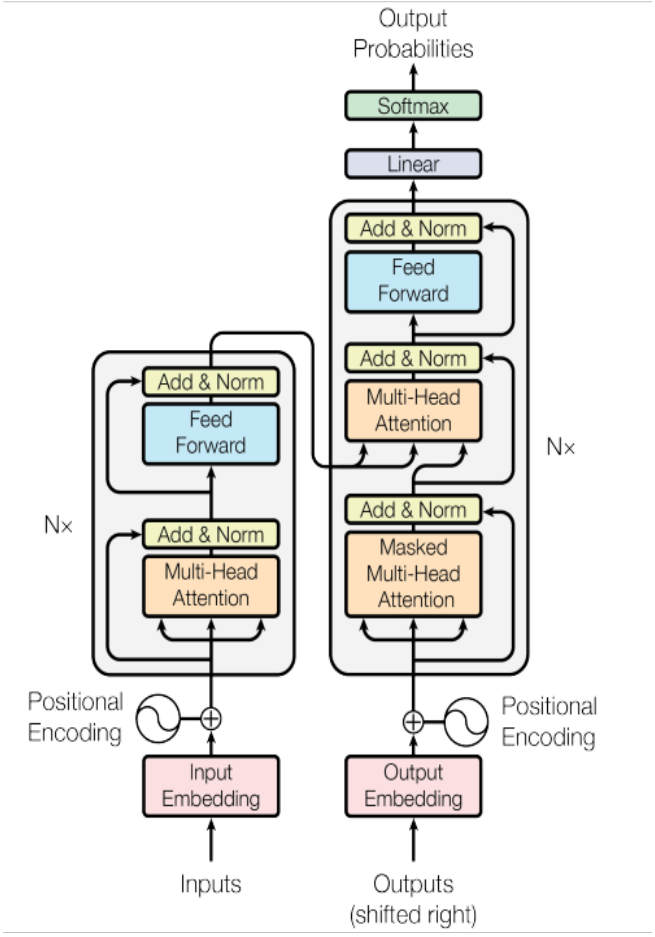

Рисунок 11 – Архітектура мережі-трансформера

В загальному випадку увага (attention) – обчислення середньозваженого середнього на виході попереднього шару. Цей метод використовувався у мовних моделях, що мають повторення, наприклад у LSTM (Long short-term memory), та у згорткових моделях, наприклад QRNN або quasi-RNN).

У роботі «Attention Is All You Need» [15] зазначається, що для гарних результатів додаткові компоненти не потрібні, і увага (attention) – це все, що потрібно. Цей підхід має досить помітні вимоги до пам'яті, але повністю паралельний, добре масштабується та є відносно простим [15].

На вхід шару подаються вектори Q і декілька пар K і V. Зазвичай K і V це один і той же вектор. Кожен з них перетворюється лінійним перетворенням зі здатністю навчатися, обчислюється скалярний добуток Q з усіма K по черзі. Результат скалярних добутків через функцію softmax. Отримуються ваги, з якими усі вектори V сумуються в один вектор [15].

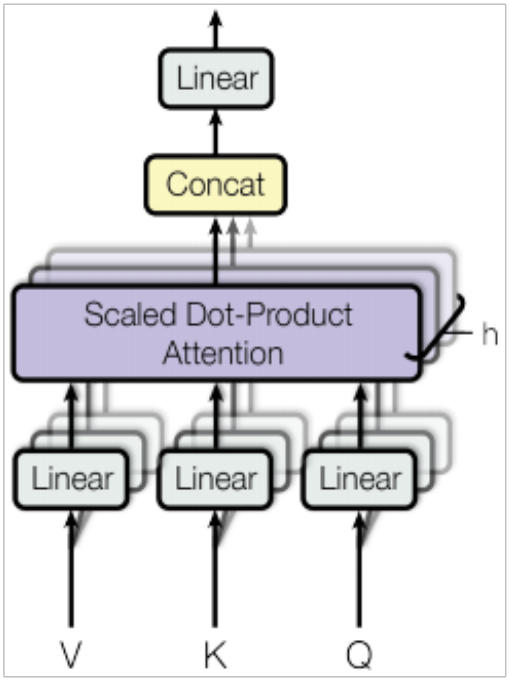

Рисунок 12 – Структура шару multi-head attention

Векторів уваги (attention) одночасно тренується декілька (h), після чого результат конкатенується і проходить через лінійне перетворення і передається далі у енкодері.

Завдяки тому що на виході з цього блоку отримуються вектори однакового розміру їх можна використати в мережі декілька разів [15].

Однією з ознак кожного слова є позиційне кодування (positional encoding), завдяки чому є здатність «звертати увагу» на його контекст – сусідні слова в реченні [15].

Декодер запускається на одне слово, отримує на вхід попереднє слово і повинен видаляє наступне.

У декодера є два типи використання багатоголосової уваги (multi-head attention) [16]:

– звернення до векторів минулих декодованих слів;
– звернення до виходу енкодера. В цьому випадку Q – це вхідний вектор декодеру, а пари K-V – це фінальні метадані енкодера, де один і той же вектор йде в якості і K, і V, але проходять через різні перетворення.

Процес повторюється кілька разів, де результат одного блоку передається наступному. Над результатом виконується функція softmax для отримання ймовірностей слів. Після цього виконується семплювання результатом якого буде наступне слово у реченні. Результат передається на вхід декодеру поки не закінчиться речення [16].

Попереднє навчання моделі BERT складається з двох задач:

– створення маскованої мовної моделі (mask language model)
– передбачення наступного речення

Створення маскованої мовної моделі полягає у тому, що у кожній послідовності маскують 15% токенів випадковим чином через спеціальний маркер «[MASK]». Спеціальний маркер ніколи не зустрінеться під час тонкої настройки, якщо не використовувати деякі евристичні техніки, що використовуються у BERT [16]:

– з вірогідністю 0.8 обрані слова замінюються на [MASK];
– з вірогідністю 0.1 обрані слова замінюються випадковим словом;
– з вірогідністю 0.1 слово не замінюється.

Модель буде передбачати тільки пропущені слова (замінені на «[MASK]»), але не буде мати інформацію про власне замінені слова, через що розмірність результату роботи моделі буде становити лише 15% від розмірності вхідної послідовності [16].

Для розуміння зв’язків між реченнями, що може бути використано для задач, в яких це необхідно, наприклад відповіді на запитання чи умовиводу, BERT вводить ще одне додатковий етап для підготовки бінарного класифікатора, який необхідний для визначення, чи є одне речення похідним від іншого [16].

необхідний для визначення, чи є одне речення похідним від іншого [16]. Таким чином розподіл зв'язків між парами речень буде наступним:

– у 50% випадків, речення B слідує за реченням A;

– у 50% випадків, речення B не слідує за реченням A.

Після обробки цього набору даних модель виводить мітку (бінарну), яка позначає, чи є речення B похідним від речення A.

Ці два етапи можливі для будь-якого текстового корпусу, головне щоб він був на одній мові, тому можливі набори даних не мають обмежень [16].

Після попереднього навчання відбувається етап введення ембедингів (embeddings), який складається за таких кроків (див. рис. 13):

– токенизація за допомогою WordPiece;

– введення ембедингів до сегментів;

– введення позиційних ембедингів.

Токенизаційна модель WordPiece вперше була використана для сегментації японських або корейських слів. Слова цих мов можна додатково розділити на більш дрібні одиниці, для більш ефективної обробки невідомі або просто рідкісних слів [19].

Введення ембедингів до сегментів відбувається наступним чином: якщо вхідна послідовність має два речення, вони будуть мати ембединги для речення A і для речення B відповідно та розділені спеціальним токеном «[SEP]». Якщо речення лише одне, то будуть використовуватись лише ембединги з речення A [19].

Позиційні ембединги не задаються вручну, а виводяться під час процесу навчання [19].

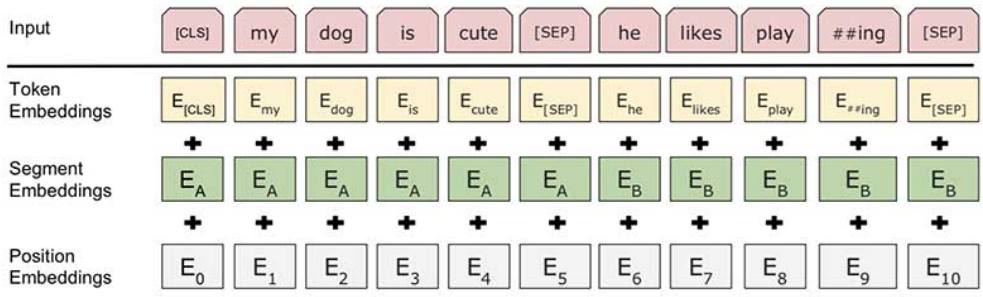

Рисунок 13 – Структура вхідних даних BERT

Як видно на рисунку першим елементом послідовності є «[CLS]», який пізніше буде використовуватися у більш прикладних задачах.

Наступним етапом є тонке налаштування моделі на конкретну задачу. Як і у випадку з GPT, BERT потребує лише невелику кількість додаткових параметрів для підлаштування під специфічну задачу, при чому процес попереднього тренування та тонкого налаштування не відрізняються архітектурно (див. рис. 14) [19].

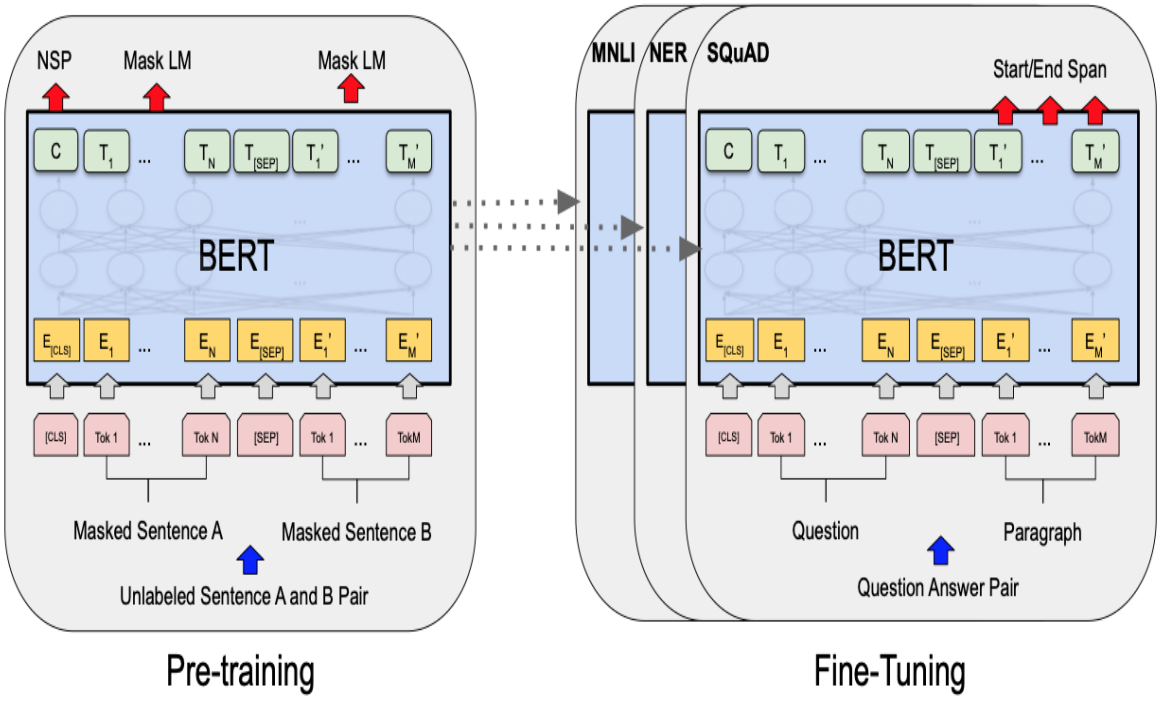

Рисунок 14 – Порівняння процесів попереднього навчання та налаштування

Експериментальні дослідження потребували 1 години на хмарному TPU, або кілька годин на GPU, що є дешевшою операцією, відносно попереднього навчання.

В результаті роботи науковців з Google було розроблено дві аналогічні моделі, різні за розміром: $\text{BERT}_{\text{BASE}}$ та $\text{BERT}_{\text{LARGE}}$ [19]:

– $\text{BERT}_{\text{BASE}}$ має наступні розміри: L = 12, H = 768, A = 12, кількість параметрів = 110 мільйонів;
– $\text{BERT}_{\text{LARGE}}$ має наступні розміри: L = 24, H = 1024, A = 16, загальні параметри = 340 мільйонів.

L означає кількість шарів (блоків мережі-трансформеру), H – розмірність схованого шару, A – кількість елементів уваги (attention).

У роботі наводять результати порівняння цих двох моделей з BiLSTM+ELMo+Attn (state-of-the-art рішення на той момент) та GPT у тесті GLUE (загальна оцінка розуміння мови), які наведені на рисунку 15.

System	MNLI-(m/mm) 392k	QQP 363k	QNLI 108k	SST-2 67k	CoLA 8.5k	STS-B 5.7k	MRPC 3.5k	RTE 2.5k	Average -
Pre-OpenAI SOTA	80.6/80.1	66.1	82.3	93.2	35.0	81.0	86.0	61.7	74.0
BiLSTM+ELMo+Attn	76.4/76.1	64.8	79.8	90.4	36.0	73.3	84.9	56.8	71.0
OpenAI GPT	82.1/81.4	70.3	87.4	91.3	45.4	80.0	82.3	56.0	75.1
$BERT_{BASE}$	84.6/83.4	71.2	90.5	93.5	52.1	85.8	88.9	66.4	79.6
$BERT_{LARGE}$	**86.7/85.9**	**72.1**	**92.7**	**94.9**	**60.5**	**86.5**	**89.3**	**70.1**	**82.1**

Рисунок 15 – Результати порівняння моделей BERT з іншими рішеннями

Як можна побачити з результатів, моделі BERT дають кращу точність, ніж існуючі на той момент рішення, демонструючи приріст на 4,5% та 7,0% відповідно для $BERT_{BASE}$ та $BERT_{LARGE}$ при виконанні різних задач (MNLI, QQP, QNLI, SST-2, CoLA, STS-B, MRPC, RTE) відповідно до тесту GLUE [19].

В результаті роботи автори зробили внесок в подальшому узагальненні висновків щодо трансферного навчання з мовними моделями, продемонстрували що попереднє навчання моделі є важливою частиною систем розуміння мови. Завдяки цьому підходу можливе використання моделей глибокого навчання з невеликими обчислювальними ресурсами.

Розроблена глибока двонаправлена архітектура дозволяє одній попередньо тренованій моделі бути використаною для вирішення широкого спектру задач обробки природних мов [19].

1.5.3 XLNet

XLNet – узагальнений метод авторегресивного попереднього навчання, який базується на BERT-моделі, тому не є по суті новою моделлю [17].

Авторегресивна мовна модель використовує контекстне слово у процесі передбачення слова, що йде після нього [17]. В рамках XLNet контекстне слово обмежене двома напрямками – прямим та оберненим (вперед або назад у послідовності).

Головною перевагою таких моделей є те, що вони гарно підходять для задач, пов'язаних з генерацію нових послідовностей, оскільки генерація нового контексту зазвичай пов'язана саме з прямим напрямком, але є й недолік, оскільки ці моделі можуть працювати лише в одному напрямку одночасно, тому відпадають позитивні ефекти використання обох напрямків одразу [17].

BERT у свою черго позиціонується як автоенкодуюча мовна модель, завдяки чому вона має можливість використовувати контекст в обох напрямках і з більшою точністю передбачати слова у середині послідовності, а також має властивість реконструювати речення, базуючись на контексті.

Недолік цього підходу полягає в тому, що спеціальний маскуючий токен, що використовується під час попереднього навчання буде відсутній на етапі тонкого налаштування, що може призводити до певних невідповідностей, а також виключається залежність між передбаченими словами, оскільки вони трактуються як незалежні [17].

Мета створення XLNet полягає у розробці такої мовної авторегресійної моделі, яка могла б навчатися з обох напрямків контексту одночасно, оминаючи недоліки, пов'язані з відсутністю зв'язків між маскованими словами як у моделі BERT [17].

Як і мовні моделі розглянуть раніше, XLNet складається з двох фаз:

– попереднє навчання;

– тонке налаштування.

Головною особливість цієї моделі є те, що у першій фазі присутній процес моделювання мови перестановок (permutation language modeling) [18].

На рисунку 16 наведені варіанти прогнозування частини послідовності x_3 у послідовності x, використовуючи різні перестановки.

Ідея AR на основі перестановки була досліджена раніше, однак XLNet відрізняється своїм підходом [18].

Ідея AR на основі перестановки була досліджена раніше, однак XLNet відрізняється своїм підходом [18].

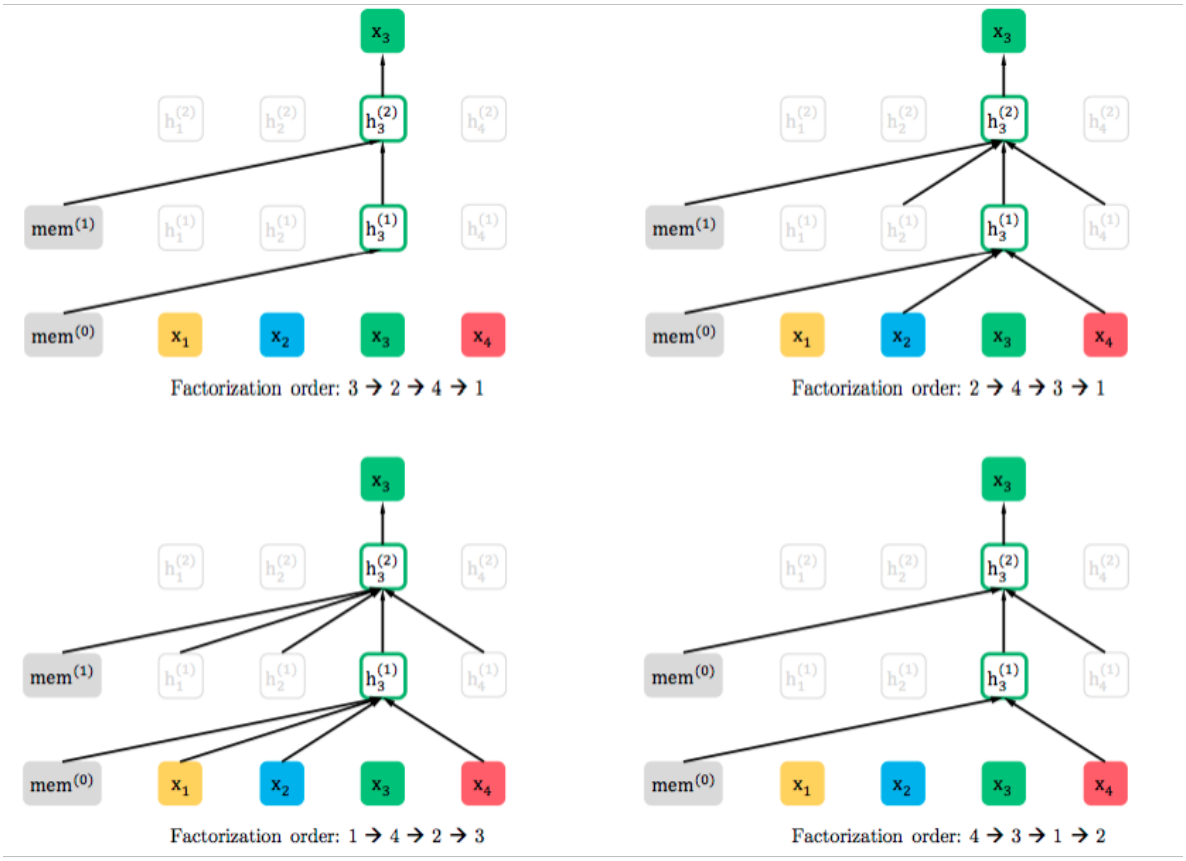

Рисунок 16 – Моделювання мови перестановок у XLNet

Перша відмінність полягає у тому, що попередні моделі покращують щільність оцінки через підготовку «невпорядкованого» індуктивного відхилення в моделі, а XLNet дозволяє мовній AR-моделі вивчати двонаправлені контексти.

Для побудови правильного розподілу передбачення, що орієнтоване на кінцеву ціль, технічно для побудови дійсного розподілу прогнозування, орієнтованого на ціль, XLNet додає до прихованого стану цільову позицію через двопоточну увагу (attention).

Попередні AR-моделі залежать від неявної інформованості про позицію, що властива їх архітектурам MLP (багатошаровий перцептрон). Слід зазначити, що порядок слів у вхідній послідовності не є випадковим, а лише може бути переставлений обмеженою кількістю перестановок [18]. XLNet запозичує ідеї від NADE (метод оцінки нейронного авторегресивного розподілу) [19] і пропонує метод моделювання мови перестановок.

XLNet запозичує ідеї від NADE (метод оцінки нейронного авторегресивного розподілу) [19] і пропонує метод моделювання мови перестановок, яка зберігає позитивні сторони мовних AR-моделей та надає можливість використовувати двосторонні контексти. Для послідовності X довжини T існує T! різних перестановок для виконання правильної авторегресивної факторизації. Таким чином, якщо параметри моделі спільні для всіх послідовностей факторизації, тому модель має навчитися отримувати інформацію з обох сторін цільового слова в послідовності [19].

Нехай Z_T є сукупністю всіх можливих перестановок послідовності індексу T, [1, 2, ..., T]. Для позначення t-го елемента та перших t − 1 елементів перестановки z ∈ Z_T використовуємо z_t і z < t. Тоді мета моделювання мови перестановок може бути позначена формулою (2) [19].

$$\max_{\theta} \mathbb{E}_{\mathbf{z}\sim Z_T} \sum_{t=1}^{T} \log p_\theta(x_{z_t} \mid \mathbf{x}_{z<t}), \quad (2)$$

де Z_T – набір перестановок, p_θ – функція схожості, x – текстова послідовність, z – порядок факторизації.

Важливою частиною моделі XLNet є механізм двопотокової самоуваги (Two-Stream Self-Attention). Ідея цілеорієнтованих представлень виключає неоднозначність в передбаченні цілей.

Також запропоновано зафіксуватися на цільовій позиції Zt для отримання інформації з контексту xz<t за допомогою уваги (attention). Для того, щоб така параметризація працювала є дві вимоги, що є суперечливими для стандартної архітектури мережі-трансформера.

Перша вимога – передбачення xz, g0(xz<t, zt) має використовувати лише позицію zt, а не зміст xzt, в іншому випадку мета стає тривіальною. Друга вимога – для передбачення інших токенів xzj, де j > t, g0(xz<t, zt) має кодувати зміст xzt щоб надавати повну інформацію з контексту [20].

Для вирішення цих проблем використовуються два набори прихованих представлень замість одного [20]:

– представлення контенту h_{zt}, що виконує роль схожу на роль прихованих станів у «трансформері». Це представлення кодує як контекст, так і x_{zt};

– представлення запиту $g_0(x_{z<t}, z_t)$ має доступ лише до контекстуальної інформації $x_{z<t}$ та позиції z_t без даних про зміст x_{zt}.

Два потоку уваги описуються формулою (3), наведеною нижче [20].

$$\begin{aligned} g_{*\#}^{(-)} &\leftarrow \text{Attention EQ} = g_{*\#}^{(-/))}, \text{KV} = \mathbf{h}_{+0'}^{(-/))}; \theta \text{K}, \text{(потік запиту)} \\ h_{*\#}^{(-)} &\leftarrow \text{Attention EQ} = h_{*\#}^{(-/))}, \text{KV} = \mathbf{h}_{+0'}^{(-/))}; \theta \text{K}, \text{(потік змісту)} \end{aligned}, \quad (3)$$

де $g_{*\#}^{(-)}$ – представлення запиту, $h_{*\#}^{(-)}$ – представлення контенту, Q – запит, K – ключ, V – значення в операції уваги

Архітектура двопотокового самоуваження для цільових уявлень (Two-Stream Self-Attention for Target-Aware Representations) подана на рисунку 17 [20].

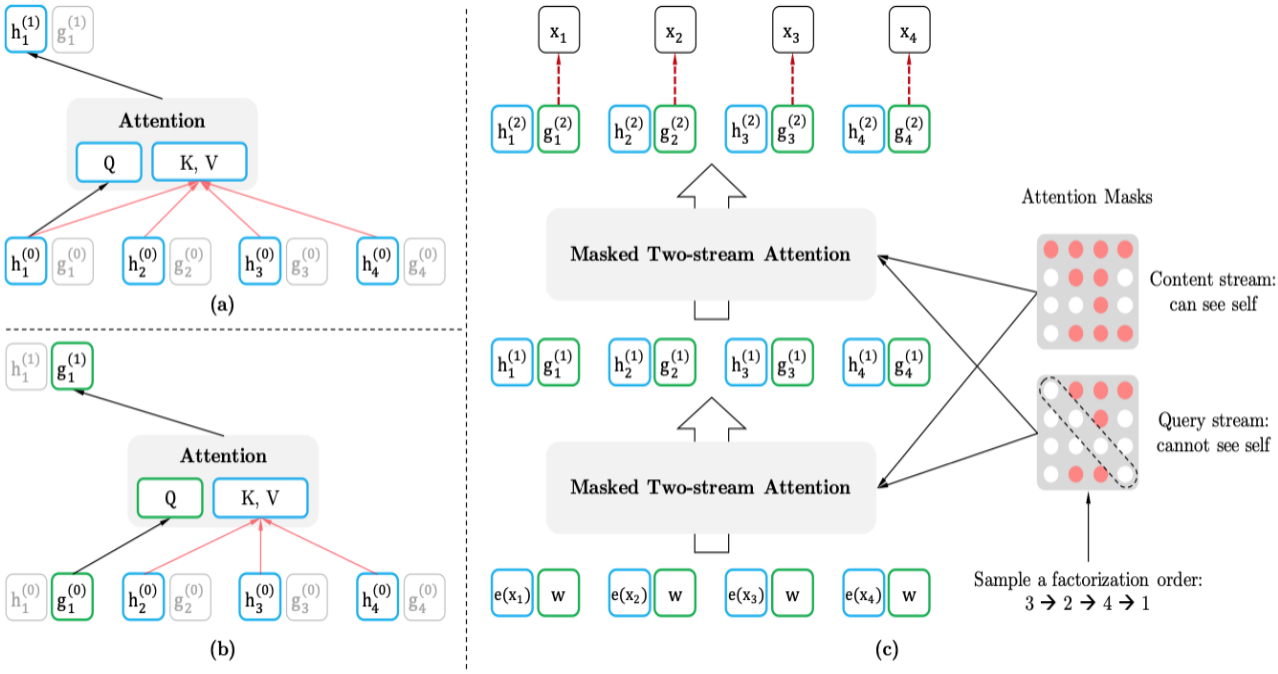

Рисунок 17 – Зображення процесу двопотокової самоуваги

Частина (a) рисунку зображує увагу для потоку змісту, частина (b) зображує увагу для потоку запиту, що не містить інформації про контент. (c) зображує процес навчання моделювання мови перестановок за допомогою двопоточної уваги [20].

В результаті експериментів автори отримали результат кращий за попередні рішення з використанням GPT, BERT та RoBERTa, що наведені на рисунку 18 [20].

Model	MNLI	QNLI	QQP	RTE	SST-2	MRPC	CoLA	STS-B	WNLI
Single-task single models on dev									
BERT [2]	86.6/-	92.3	91.3	70.4	93.2	88.0	60.6	90.0	-
RoBERTa [21]	90.2/90.2	94.7	92.2	**86.6**	96.4	**90.9**	68.0	92.4	-
XLNet	**90.8/90.8**	**94.9**	**92.3**	85.9	**97.0**	90.8	**69.0**	**92.5**	-
Multi-task ensembles on test (from leaderboard as of Oct 28, 2019)									
MT-DNN* [20]	87.9/87.4	96.0	89.9	86.3	96.5	92.7	68.4	91.1	89.0
RoBERTa* [21]	90.8/90.2	98.9	90.2	88.2	96.7	92.3	67.8	92.2	89.0
XLNet*	**90.9/90.9**†	**99.0**†	**90.4**†	**88.5**	**97.1**†	**92.9**	**70.2**	**93.0**	**92.5**

Рисунок 18 – Результати тесту GLUE для BERT, RoBERTa та XLNet

Як видно з результатів тесту, XLNet дає більшу точність на 4,2% ніж BERT та на 0.6% ніж RoBERTa.

Таким чином, в результаті роботи над XLNet було розроблено узагальнений метод попереднього навчання AR, який використовує моделювання мови перестановок для поєднання переваг методів авторегресії та автоенкодингу. Нейронна архітектура XLNet розроблена для безшовної роботи з метою AR, включаючи інтеграцію Transformer-XL. Також було спроектовано механізм двопотокової уваги. В результати, отримана модель XLNet є значним вдосконаленням, порівняно з попередніми моделями (BERT, RoBERTa і GPT) [20].

XLNet запозичує ідеї від NADE (метод оцінки нейронного авторегресивного розподілу) [19] і пропонує метод моделювання мови перестановок, яка зберігає позитивні сторони мовних AR-моделей та надає можливість використовувати двосторонні контексти. Для послідовності X довжини T існує T! різних перестановок для виконання правильної авторегресивної факторизації. Таким чином, якщо параметри моделі спільні для всіх послідовностей факторизації, тому модель має навчитися отримувати інформацію з обох сторін цільового слова в послідовності [19].

1.6 Постановка задачі дослідження

Звертаючи увагу на розглянуті вище роботи, де моделі тестувалися у загальних тестах (GLUE) для широкого спектру задач, виникає питання як будуть поводитися ці моделі на конкретній задачі аналізу тональності тексту для виявлення думки споживачів товарів інтернет-магазинів, а саме на наборі даних Amazon product data [21].

Наукова задача даної роботи полягає в тому, щоб виявити найкращу модель для вирішення цієї задачі, шляхом налаштування трьох моделей (ULMFiT, BERT та XLNet) для роботи з відгуками споживачів інтернет-магазину Amazon і проведення практичних експериментів з кожною моделлю на одному наборі даних. Це дозволить виявити найбільш придатну для цієї задачі модель, спираючись на метрику точності (accuracy), і в майбутньому використовувати її в інтернет-магазинах для дослідження голосу споживачів (voice of customer) з метою вдосконалення маркетингових компаній.

2 ПРОВЕДЕННЯ ТЕОРЕТИЧНИХ ДОСЛІДЖЕНЬ

2.1 Вибір напрямку дослідження

В якості напрямку дослідження було обрано аналіз трьох моделей, що є найкращими для використання у задачах, пов'язаних з обробкою природних мов, з метою з’ясування їх придатності для класифікації відгуків, що є підзадачею аналізу тональності тексту. В результаті дослідження існуючих наукових праць не було знайдено робіт пов’язаних з дослідженням і порівнянням моделей BERT, XLNet та ULMFiT для аналізу відгуків споживачів товарів [21].

Методом дослідження є аналіз існуючих робіт пов’язаних з сентимент-аналізом, де використовуються вище зазначені моделі, а також експерименти на наборі даних з відгуками про товари від клієнтів інтернет-магазину Amazon [21].

В якості експерименту було проведено класифікацію текстів відгуків за п’ятьма класами, що відповідають оцінкам від однієї до п’яти зірок з використанням BERT, XLNet та ULMFiT без тонкої настройки на конкретний датасет для порівняння моделей у базовому стані. Було проведено класифікацію і порівняно з очікуваними результатами для розрахунку метрик, таких як точність (accuracy) та інших [21].

2.2 Результати теоретичного дослідження існуючих робіт

Перед початком експериментів було досліджено існуючі результати використання моделей BERT, XLNet та ULMFiT специфічно для аналізу тексту, а саме його багатокласової класифікації [21]. В якості експерименту було проведено класифікацію текстів відгуків за п’ятьма класами, що відповідають оцінкам від однієї до п’яти зірок з використанням BERT, XLNet та ULMFiT без тонкої настройки на конкретний датасет для порівняння моделей у базовому стані.

2.2.1 Дослідження аналізу з використанням BERT

Під час дослідження було розглянуто роботу, пов'язану з використанням BERT для аналізу тексту відгуків про фільми [22]. Автори роботи використовували BERT з простою методикою тонкого налаштування моделі на набір даних для отримання найкращого результату [22].

В якості робочого набору даних було використано SST (Stanford Sentiment Treebank), що являє собою один з найбільш вживаних наборів даних серед відкритих джерел для роботи з багатокласовою класифікацією текстів, а саме аналізом тональності з розділенням на 5 класів: «дуже негативно», «негативно», «нейтрально», «позитивно», «дуже позитивно».

У наборі міститься 11855 відгуків про фільми з бази Rotten Tomatoes, що складаються з одного речення. Також у наборі містяться частини цих речень, розділені за допомогою спеціального парсера, що створює деревоподібну структуру речень, де коренем я ціле речення, а листя – частини цього речення, для яких вказано певний сентимент (див. рис. 19) [22].

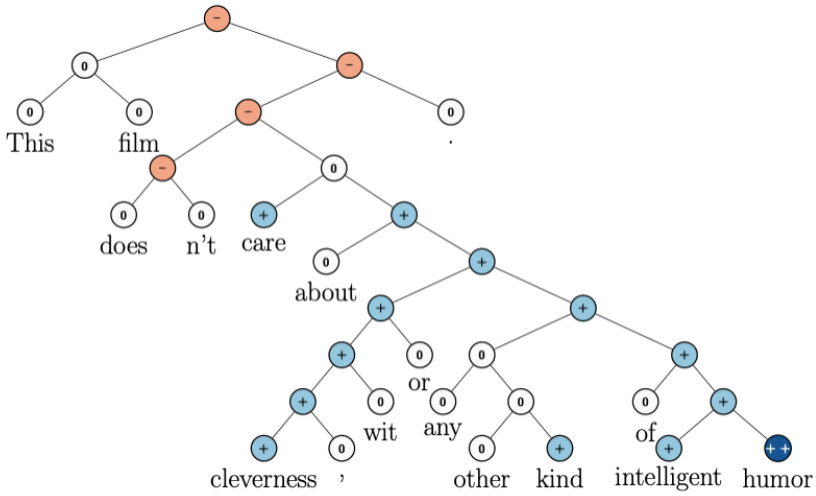

Рисунок 19 – Структура набору даних SST

Таким чином, сумарно, цей датасет містить 215,154 унікальних текстів різної довжини і має вищенаведену структуру.

Методика дослідження полягає у класифікації відгуків з в наведеного вище набору даних і складається з чотирьох етапів (див. рис. 20), не включаючи подання речення на вхід моделі і отримання мітки класу на виході [22]:

– попередня обробка вхідного відгуку (WordPiece) [23];
– введення BERT-специфічних ембедингів у токени відгуку;
– виключення незначущих нейронів з нейронної мережі (dropout);
– класифікація за допомогою softmax.

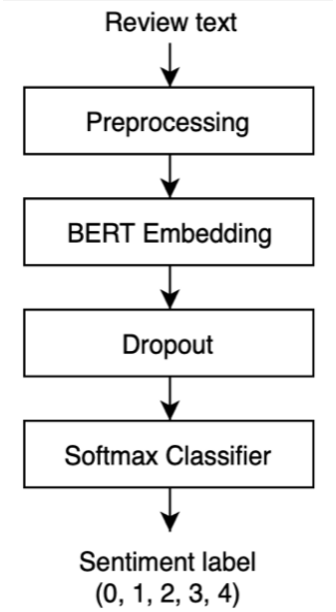

Рисунок 20 – Візуальний вигляд етапів обробки відгуків

Перший етап пов’язаний з попередньою обробки вхідної послідовності перед подачею на вхід до нейронної мережі полягає у канонікалізації, токенизації та додавання спеціальних токенів «[CLS]» та «[SEP]» до послідовностей (див. рис. 21) [22].

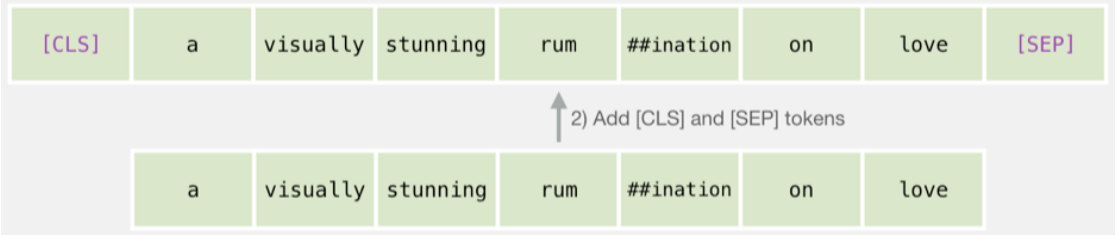

Рисунок 21 – Додавання BERT-специфічних токенів-маркерів

Канонікалізація має на увазі видаляення усіх цифр, наголосів та знаків пунктуації з речення, а також конвертацію усіх літер у рядкові [22].

Токенізація полягає у обробці тексту спеціальним токенізатором Word-Piece, який розбиває слова на морфеми: префікс, корінь та суфікс для більш точної обробки слів, що можуть не зустрітись у виборці для навчання, але будуть присутні у вибірці для тестування роботи моделі [22].

Додавання спеціальних лексем – етап що полягає у вставці спеціальних токенів-маркерів, специфічних для моделі BERT – «[CLS]» та «[SEP]». Маркер «[CLS]» вставляється перед початком речення, а «[SEP]» – після останнього слова в реченні [22].

Етап виключення незначущих нейронів з нейронної мережі (dropout) – це зміна структури нейронів таким чином, що деякі нейрони виключаються з мережі з певною однаковою вірогідністю. Виключення нейрона полягає в тому що при будь-якому вході він повертає 0 в якості результату і не мають впливу на процес оберненого поширення похибки (backpropagation). Це виключає проблему надмірного навчання (overfitting), так як ці нейрони не будуть впливати на процес адаптації інших нейронів, з якими вони пов’язані [22].

За умови, що лінійна проекція вхідного d_1-мірного вектору x на d_2-мірний простір вихідних значень виражена формулою (4) [23]:

$$h(x) = xW + b, \tag{4}$$

Функція активації позначена, як $a(h)$, тоді застосування процедури виключення нейронів може бути представлена як змінена функція активації, що виражена формулою (5) [23].

$$f(h) = D \odot a(h), \tag{5}$$

де $D = (X_), \cdots, X_{3_\%})$ – d_2-мірний вектор випадкових значень X_1, що розподілені за законом Бернуллі, $a(h)$ – функція активації.

Етап класифікації за допомогою softmax представляє собою повнозв'язний шар нейронної мережі з відповідною функцією активації (6) [23].

Обирається вихідний вузол з найбільшою вірогідністю в якості передбачуваної мітки моделі для відповідної вхідної послідовності [23].

$$\sigma(\mathbf{z})_1 = \frac{e^{*\&}}{\sum_{5\,()}^{4} e^{*,}}, \text{ де } i = 1, \dots, K, \tag{6}$$

де $\mathbf{z} = (z_), \dots, z_4) \in \mathbb{R}^4$ є проміжним результатом шару softmax, що називається «logits», $e^{*\&}$ – елемент вектору параметрів.

Під час експериментів автори порівнювали показник точності отриманої моделі BERT (accuracy) з іншими моделями, такими, як RNN, RNTN, LSTM та CNN (див. рис. 22) [23].

Model	SST-2	
	All	Root
Avg word vectors [9]	85.1	80.1
RNN [8]	86.1	82.4
RNTN [9]	87.6	85.4
Paragraph vectors [2]	–	87.8
LSTM [10]	–	84.9
BiLSTM [10]	–	87.5
CNN [11]	–	87.2
$\text{BERT}_{\text{BASE}}$	94.0	91.2
$\text{BERT}_{\text{LARGE}}$	**94.7**	**93.1**

Рисунок 22 – Порівняння метрики точності отриманої BERT-моделі

Як можна побачити, отримана авторами модель демонструє найкращий результат з різницею в 6,4% для BERT base та 7,1% для BERT large [24].

Таким чином автори довели, що з мінімальними змінами модель BERT придатна для аналізу тональності тексту, а саме класифікації за 5 класами [24].

автори довели, що з мінімальними змінами модель BERT придатна для аналізу тональності тексту, а саме класифікації за 5 класами [24].

2.2.2 Дослідження аналізу з використанням ULMFiT

Під час дослідження також було розглянуто роботу, де модель ULMFiT порівнюється з іншими методами рішення поставленої задачі – класифікація сентиментів на наборі даних Amazon Customer Reviews, що містить відгуки споживачів про товари інтернет-магазину Amazon. Автори роботи порівнювали метрику точності (accuracy) для BOW, CNN, HCNN, HAN та ULMFiT на вищевказаному наборі даних з різними варіантами попередньої обробки [25].

З набору даних було використано 10 тисяч відгуків для навчання, та 35 тисяч для тестування. Кожну одиницю даних (речення) було токенізовано за допомогою UDPipe [25]. Як і у випадку з дослідженням BERT у попередньому підпункту, у вибірці міститься 5 класів: : «дуже негативно», «негативно», «нейтрально», «позитивно», «дуже позитивно» [25].

Було досліджено вплив розміру вибірки на точність класифікації. Найбільша втрата точності спостерігалася для CNN, що можна побачити на рисунку 23 [25].

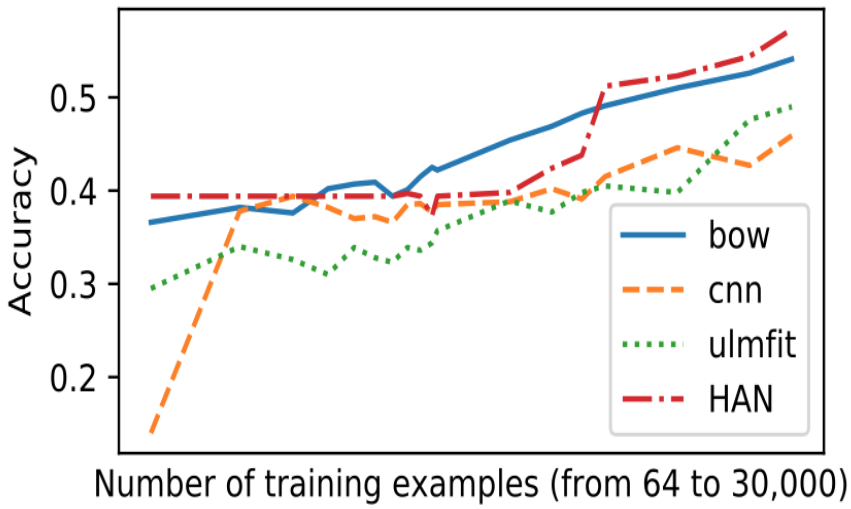

Рисунок 23 – Залежність точності від розміру набору даних

Фільтрація об’єктивних речень (без сентиментальної складової) з набору даних не проводилася, оскільки підходи останніх років дозволяють моделям самостійно відрізняти доцільність використання певних речень для навчання [26].

Дослідження впливу розміру речення на точність класифікації також показало, що HAN значно краще класифікує великі речення, ніж ULMFiT [26].

Тонке налаштування мовної моделі ULMFiT було виконано згідно з запропонованими методиками, використовуючи похилі трикутні коефіцієнти навчання та дискримінативне тонке налаштування [26].

В результаті роботи автори встановили протиріччя у тому, що в оригінальній роботі, пов’язаній з дослідженням ULMFiT, ця модель дає найбільшу точність на багатьох задачах обробки природних мов [26].

2.2.3 Дослідження аналізу з використанням XLNet

Під час дослідження було розглянуто роботу, де модель XLNet використовується для класифікації відгуків з набору даних Amazon Customer Reviews. Автори роботи порівнювали метрику точності (accuracy) для наступних моделей: DAmSDA, CNN-aux, AMN, HATN, HANP, BERT та XLNet (див. рис. 24).

В результаті тестування, було з’ясовано що BERT і XLNet мають значну перевагу перед усіма іншими методами [27].

Таким чином, BERT дає більшу точність класифікації, використовуючи близько 300 тренувальних, що в 20 разів менше даних, використаних для попередніх рішень. XLNet перевершує попередні методи, якщо провести тонку настройку на 50 зразках з навчальної вибірки, що приблизно в 120 разів менше даних.

Це каже про те, що використання попередньо тренованих моделей, що використовують нейронні мережі з архітектурою «трансформер», краще підходять для аналізу тональності тексту, оскільки вони є високоадаптивними у відношенні захоплення контексту на невеликій кількості зразків з навчальної вибірки [27].

Source	Target	DAmSDA	CNN-aux	AMN	HATN	HANP	BERT	XLNet
Books	DVD	86.12%	84.42%	85.62%	87.07%	88.12%	92.49%	**95.10%**
	Electronics	79.02%	80.63%	80.55%	85.75%	85.81%	93.13%	**95.92%**
	Kitchen	81.05%	83.38%	81.88%	87.03%	88.91%	94.08%	**96.54%**
	Video	84.98%	84.43%	87.25%	87.80%	89.21%	91.75%	**94.54%**
DVD	Books	85.17%	83.07%	84.53%	87.78%	89.18%	93.67%	**95.68%**
	Electronics	76.17%	80.35%	80.42%	86.32%	86.87%	93.25%	**95.17%**
	Kitchen	82.60%	81.68%	81.67%	87.47%	88.54%	94.15%	**96.42%**
	Video	83.80%	85.87%	87.40%	89.12%	91.25%	93.88%	**95.82%**
Electronics	Books	79.92%	77.38%	77.52%	84.03%	85.67%	91.83%	**93.56%**
	DVD	82.63%	79.07%	80.53%	84.32%	85.29%	89.93%	**91.99%**
	Kitchen	85.80%	87.15%	87.83%	90.08%	91.08%	95.37%	**96.79%**
	Video	81.70%	78.78%	82.12%	84.18%	85.96%	89.33%	**91.79%**
Kitchen	Books	80.55%	78.47%	79.05%	84.88%	85.04%	91.74%	**95.29%**
	DVD	82.18%	79.07%	79.50%	84.72%	86.47%	90.34%	**94.44%**
	Electronics	88.00%	86.73%	86.68%	89.33%	90.43%	94.82%	**96.46%**
	Video	81.47%	78.82%	82.15%	84.85%	85.93%	89.82%	**94.31%**
Video	Books	83.00%	81.48%	83.50%	87.10%	88.94%	93.05%	**95.31%**
	DVD	85.90%	85.25%	86.88%	87.90%	88.54%	93.32%	**95.60%**
	Electronics	77.67%	82.32%	79.68%	85.98%	86.11%	92.87%	**95.71%**
	Kitchen	79.52%	81.28%	80.98%	86.45%	87.21%	93.35%	**96.11%**
Average		82.36%	81.98%	82.79%	86.61%	87.76%	92.61%	**95.13%**

Рисунок 24 – Порівняння XLNet з попередніми методами класифікації

У порівнянні, XLNet має краще показники точності, ніж BERT, на будь-якому розмірі навчальної вибірки, що можна побачити на рисунку 25 [27].

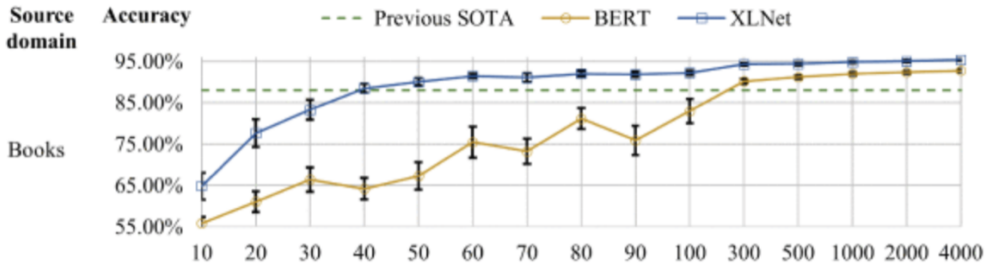

Рисунок 25 – Графіки залежності точності від розміру навчальної вибірки

XLNet є більш ефективним, ніж BERT при проведенні тестування, витрачаючи менше часу на 10%.

З іншого боку, для XLNet необхідно набагато більше обчислювальних потужностей при навчанні, таки чином XLNet працює на 20% повільніше за BERT

при кількості кроків навчання рівній 3000. Це відбувається через механізм повтору сегментів для фіксації контекстних залежностей у документах, що мають довжину більшу за максимальну. Під час тестування цей механізм скорочує час виконання XLNet менш, ніж BERT [27].

Таким чином, в результаті роботи автори довели, що XLNet є більш ефективним ніж BERT на задачах, пов’язаних з аналізом відгуків, оскільки він дає кращі показники точності при використанні меншої кількості зразків для тренуванняі здатен використовувати ширший контекст, використовуючи можливі віддалені залежності. При цьому XLNet потребує більше ресурсів під час навчання, але робить цей етап швидше за BERT [27].

3 РОЗРОБКА МОДЕЛІ ТА АНАЛІЗ ОТРИМАНИХ РЕЗУЛЬТАТІВ

Метою експериментального дослідження є класифікація текстів відгуків за п’ятьма класами, що відповідають оцінкам від однієї до п’яти зірок з використанням ULMFiT (LSTM), BERT, XLNet та без тонкої настройки на конкретний датасет для порівняння моделей у базовому стані. Для цього проведено класифікацію і порівняно з очікуваними результатами для розрахунку метрики точності (accuracy) [28].

Беручи до уваги результати теоретичного дослідження має сенс висунути гіпотезу, що серед розглянутих моделей найкраща точність має бути у моделі на базі XLNet, оскільки вона демонструє найкращий результат у більшості задач пов’язаних з обробкою природних мов, а точніше, для аналізу тональності тексту [28].

3.1 Опис проведених експериментальних досліджень

Як було вказано у пункті 2.1, в якості набору даних використовується Amazon Customer Reviews (категорія електроніка), що містить 16,8 мільйонів відгуків [29].

Розподіл даних за сентиментами від «дуже негативно» до «дуже позитивно» (5 градацій) наведено на рисунку 26 [29].

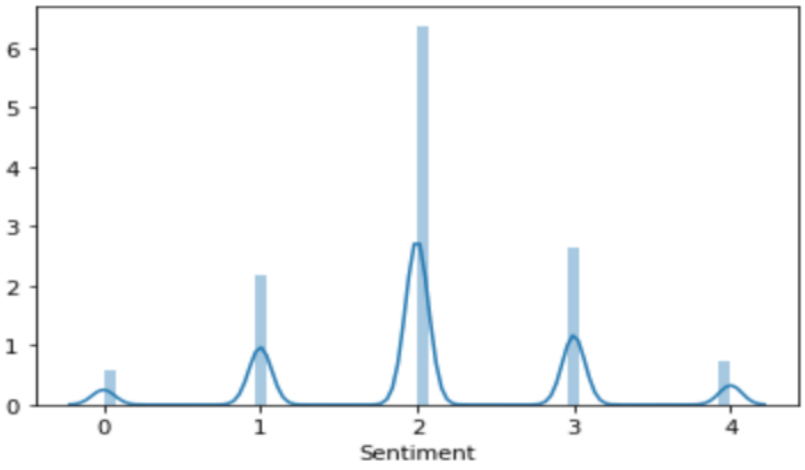

Рисунок 26 – Графік розподілу відгуків за класами у навчальній вибірці

З цього набору даних було обрано 124 тисячі відгуків в якості навчальної вибірки та 31 тисяча для тестування [29].

Кількісний розподіл навчальної вибірки наведено на рисунку 27. Як можна побачити, зразків з нейтральними відгуками значно більше, ніж в усіх інших класах [29].

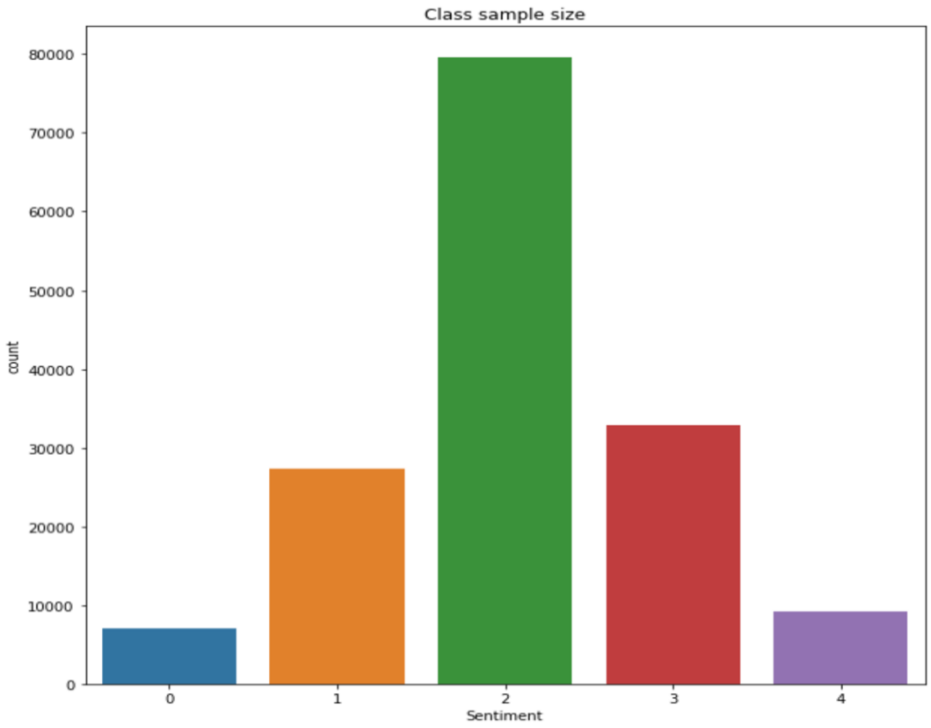

Рисунок 27 – Кількісний розподіл зразків за класами

Для оптимізації вхідних даних для навчання проведено попередню обробку, що буде описано далі [29].

Дані, представленні у табличному вигляді наведені на рисунку 28 [29].

Phrase	Sentiment
that never bothers to hand viewers a suitcase ...	2
confining color to Liyan 's backyard	2
that even its target audience talked all the w...	1

Рисунок 28 – Табличний вигляд набору даних

З набору даних використовуються 2 елементи: текст та прив’язаний до нього сентимент (що позначає клас від «дуже негативно» до «дуже позитивно») [29].

3.2 Попередня обробка даних

Зазвичай від якості даних дуже сильно залежить кінцевий результат якості моделі, тому вхідні текстові дані було оптимізовано. Для цього виконані наступні кроки [29]:

– видалено посилання з речень;

– приведено усі літери до малих;

– видалено усі незначущі частини речень (стоп-слова).

Прикладом цієї обробки є перетворення речення «They are familiar with the item» на «they familiar item». Функція для оптимізації тексту наведена на рисунку 29 [29].

```
#Text Pre-processing
def text_cleaning(sentence):
    letters = re.sub("[^a-zA-Z]"," ",sentence)
    ht = re.sub(r'http\S+', '',letters)
    mention = re.sub(r'@\w+', '', ht)
    p = re.sub(r'[^\w\s]','',mention)
    words = p.lower().split()
    stops = set(stopwords.words("english"))
    meaningful_words = [w for w in words if not w in stops]
    return( " ".join(meaningful_words))
df_dplr['text_clean']=df_dplr['Phrase'].apply(lambda x: text_cleaning(x))
```

Рисунок 29 – Функція фільтрації тексту

Для кожної моделі також було виконано додаткові кроки для виконання специфічної попередньої обробки даних під кожну модель, що описано у наступних підрозділах [29].

3.3 Розробка моделі з ULMFiT (на основі LSTM)

Оскільки ULMFiT є методом тонкої настройки моделей, для експерименту його було вирішено використати з моделлю LSTM (Long short-term memory).

Додатковим етапами попередньої обробки вхідних даних для цієї моделі є токенизація та додання відступів (padding). Код, використаний для токенизації та введення відступів наведено на рисунку 30 [29].

```
from keras.preprocessing.text import Tokenizer

#Tokenizer
tk = Tokenizer(num_words=max_words)
tk.fit_on_texts(X_train)
X_train_tk = tk.texts_to_sequences(X_train)
X_test_tk = tk.texts_to_sequences(X_test)


#Padding sequences
X_train_pad = sequence.pad_sequences(X_train_tk, maxlen=max_len)
X_test_pad = sequence.pad_sequences(X_test_tk, maxlen = max_len)
```

Рисунок 30 – Код для токенизації та введення відступів

Також ця модель є чутливою до збалансованості набору даних, тому було виконано розрахунок вагових коефіцієнтів класів (див. рис. 31) [29].

```
def get_weight(y):
    class_weight_current =  cw.compute_class_weight('balanced', np.unique(y), y)
    return class_weight_current
class_weight = get_weight(Y_train.flatten())
```

Рисунок 31 – Розрахунок вагових коефіцієнтів для класів

Базовий вигляд моделі LSTM без використання ULMFiT наведено на рисунку 32 [29].

Як можна побачити зі структури, міститься шар ембедингу, прихований шар LSTM та шар Dense для виводу вірогідності певного сентименту для вхідного тексту [29].

Layer (type)	Output Shape	Param #
embedding_1 (Embedding)	(None, 216, 128)	1917824
lstm_1 (LSTM)	(None, 128)	131584
dense_1 (Dense)	(None, 5)	645

Total params: 2,050,053
Trainable params: 2,050,053
Non-trainable params: 0

Рисунок 32 – Структура моделі без ULMFiT

Було виконано навчання цієї моделі протягом 10 епох з покращенням точності з 0.57 до 0.65 [29].

Після цього до моделі було додано шар Dropout для оптимізації, що зображено на рисунку 33 [29].

Layer (type)	Output Shape	Param #
embedding_2 (Embedding)	(None, 142, 128)	1526528
lstm_2 (LSTM)	(None, 128)	131584
dropout_1 (Dropout)	(None, 128)	0
dense_2 (Dense)	(None, 3)	387

Total params: 1,658,499
Trainable params: 1,658,499
Non-trainable params: 0

Рисунок 33 – Модель з доданим шаром Dropout

Було проведено навчання протягом 10 епох з покращенням точності з 0.65 до 0.82. Таким чином видно, що додавання шару Dropout дозволяє покращити точність моделі. В даному випадку це покращення складає 17% [30].

Наступним кроком оптимізації є додавання регуляризатора до шару LSTM (див. рис. 34) [30].

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding_3 (Embedding)      (None, 142, 128)          1526528
_________________________________________________________________
dropout_2 (Dropout)          (None, 142, 128)          0
_________________________________________________________________
lstm_3 (LSTM)                (None, 128)               131584
_________________________________________________________________
dense_3 (Dense)              (None, 3)                 387
=================================================================
Total params: 1,658,499
Trainable params: 1,658,499
Non-trainable params: 0
_________________________________________________________________
```

Рисунок 34 – Модель з регуляризатором в LSTM

Після цього також було проведено навчання протягом 10 епох. Після цього можна порівняти отримані показники точності для трьох моделей LSTM [30].

Результати порівняння моделей LSTM без ULMFiT та з ним наведені у таблиці 1 [30].

Таблиця 1 – Порівняння точності моделей LSTM з ULMFiT

Назва моделі	Точність
LSTM	0.57
LSTM + Dropout	0.65
LSTM + Regularizer + Dropout	0.82

Таким чином, можна побачити, що застосування ULMFiT з мовними моделями дає помітне покращення точності цих моделей.

У даному випадку було отримано приріст точності на 8% в порівняно з базовою моделлю.

Для подальшого порівняння буде використано показник точності моделі LSTM + Regularizer + Dropout [30].

3.4 Розробка моделі на основі BERT

Для можливості використання BERT вхідні данні необхідно додатково токенізувати наступним чином [30]:

– перед початком кожного речення вставити токен «[CLS]»;

– вставити токени слів речення;

– додати токен «[SEP]»

– додати відступ (padding).

В результаті тензор вхідної послідовності буде виглядати так, як показану на рисунку 35 [30].

```
[CLS] id : 101
[SEP] id : 102
[PAD] id : 0
Batch shape :  torch.Size([16, 80])
tensor([[ 101, 1011, 1048,  ..., 2095, 1012,  102],
        [ 101, 2031, 2000,  ...,    0,    0,    0],
        [ 101, 2065, 2017,  ...,    0,    0,    0],
        ...,
        [ 101, 1005, 1055,  ...,    0,    0,    0],
        [ 101, 2004, 2035,  ...,    0,    0,    0],
        [ 101, 3904, 1997,  ...,    0,    0,    0]])
```

Рисунок 35 – Тензор вхідної послідовності

Як можна побачити, токени «[CLS]», «[SEP]» та «[PAD]» були вставлені на відповідні місця [30].

В якості попередньо тренованої моделі BERT, що взята за основу, було використано bert-base-uncased [30].

Вихідна конфігурація моделі наведена на рисунку 36.

Ця модель містить шар енкодингу, 12 шарів-трансформерів (BERT) та вихідний шар класифікатору (див. рис. 36) [30].

Під час тренування моделі було застосовано методи похилого трикутного коефіцієнту навчання, дискримінативного коефіцієнту навчання та поступового розморожування шарів моделі [30].


```
BertConfig {
  "architectures": [
    "BertForMaskedLM"
  ],
  "attention_probs_dropout_prob": 0.1,
  "hidden_act": "gelu",
  "hidden_dropout_prob": 0.1,
  "hidden_size": 768,
  "id2label": {
    "0": "LABEL_0",
    "1": "LABEL_1",
    "2": "LABEL_2",
    "3": "LABEL_3",
    "4": "LABEL_4"
  },
  "initializer_range": 0.02,
  "intermediate_size": 3072,
  "label2id": {
    "LABEL_0": 0,
    "LABEL_1": 1,
    "LABEL_2": 2,
    "LABEL_3": 3,
    "LABEL_4": 4
  },
  "layer_norm_eps": 1e-12,
  "max_position_embeddings": 512,
  "model_type": "bert",
  "num_attention_heads": 12,
  "num_hidden_layers": 12,
  "pad_token_id": 0,
  "type_vocab_size": 2,
  "vocab_size": 30522
}
```


Рисунок 36 – Параметри моделі BERT

Першим етапом навчання є заморожування останнього шару моделі, пошук оптимального коефіцієнту навчання (див. рис. 37) та навчання моделі (див. рис. 38) [30].

```
[
 learner.model.transformer.bert.embeddings,
 learner.model.transformer.bert.encoder.layer[0],
 learner.model.transformer.bert.encoder.layer[1],
 learner.model.transformer.bert.encoder.layer[2],
 learner.model.transformer.bert.encoder.layer[3],
 learner.model.transformer.bert.encoder.layer[4],
 learner.model.transformer.bert.encoder.layer[5],
 learner.model.transformer.bert.encoder.layer[6],
 learner.model.transformer.bert.encoder.layer[7],
 learner.model.transformer.bert.encoder.layer[8],
 learner.model.transformer.bert.encoder.layer[9],
 learner.model.transformer.bert.encoder.layer[10],
 learner.model.transformer.bert.encoder.layer[11],
 learner.model.transformer.bert.pooler
]
```

Рисунок 37 – Структура моделі BERT

На рисунку 39 наведено графік залежності втрати (loss) від коефіцієнту навчання (learning rate) [30].

CustomTransformerModel

Layer (type)	Output Shape	Param #	Trainable
Linear	[80, 768]	2,360,064	False
LayerNorm	[80, 768]	1,536	False
Dropout	[80, 768]	0	False
Linear	[768]	590,592	True

Рисунок 38 – Модель із замороженими шарами

На рисунку 38 можна побачити, що усі попередні шари заморожені, крім Останнього [30].

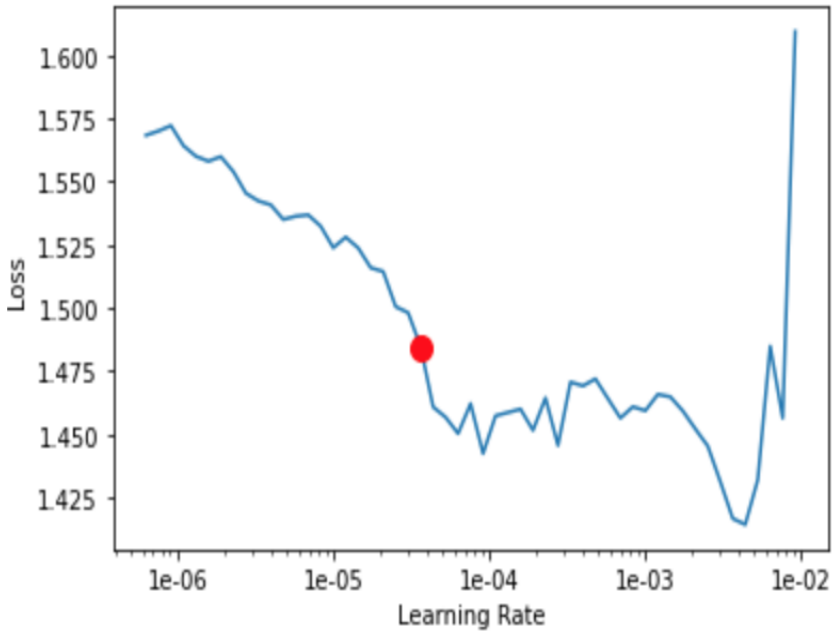

Рисунок 39 – Пошук оптимального коефіцієнту навчання

Виходячи з графіка, наведеного вище, було обрано коефіцієнт 1e-04. Після цього було запущено перший етап навчання нейронної мережі. На рисунку 39 наведено графік залежності train loss від кількості оброблених батчів [32].

Наступним кроком є повторення попередніх дій, але розморожується наступний з кінця шар моделі [32].

Як можна побачити на графіку мінімізації похибки (див. рис. 40), похибка вже менша ніж на попередньому етапі [32].

epoch	train_loss	valid_loss	accuracy	error_rate	time
0	0.939589	0.914978	0.621876	0.378124	1:00:08

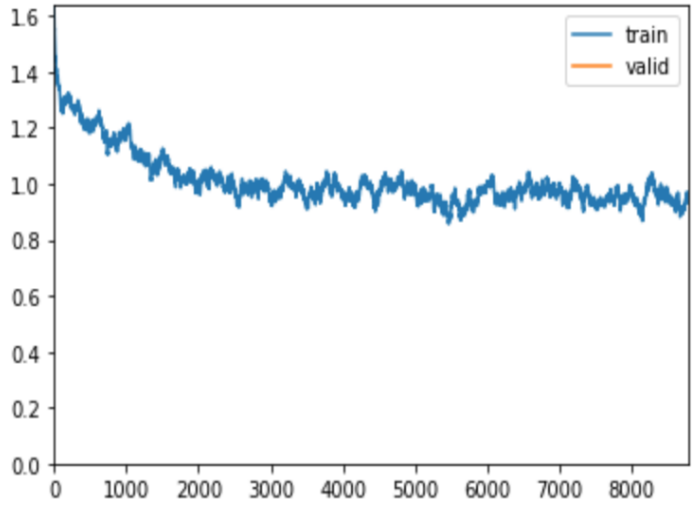

Рисунок 40 – Графік залежності похибки від кількості батчів на 1 етапі

Таким чином помітно зниження похибки з 0.93 до 0.85.

epoch	train_loss	valid_loss	accuracy	error_rate	time
0	0.856935	0.859312	0.644047	0.355953	1:20:33

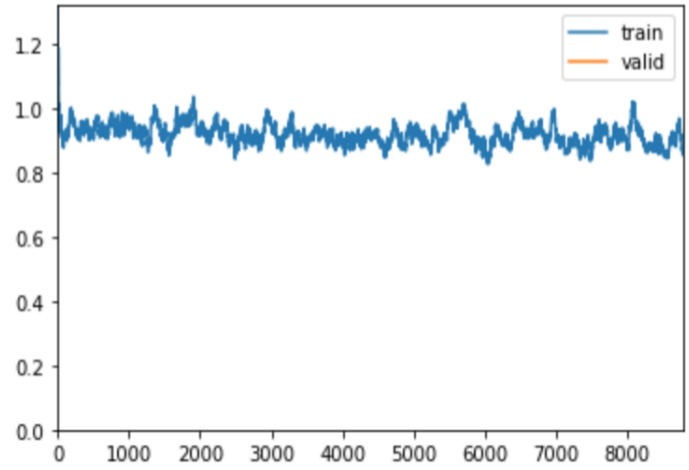

Рисунок 41 – Графік залежності похибки від кількості батчів на 2 етапі

Поступове розмороження відбувається далі і розморожується третій з кінця шар. Після цього виконується навчання. Графік похибки наведено на рисунку 41. Як видно, похибка зменшилася з 0.85 до 0.82 [32].

Після оптимізації останніх двох шарів моделі виконується завершальне навчання моделі [32].

epoch	train_loss	valid_loss	accuracy	error_rate	time
0	0.824566	0.827368	0.655581	0.344419	1:35:07

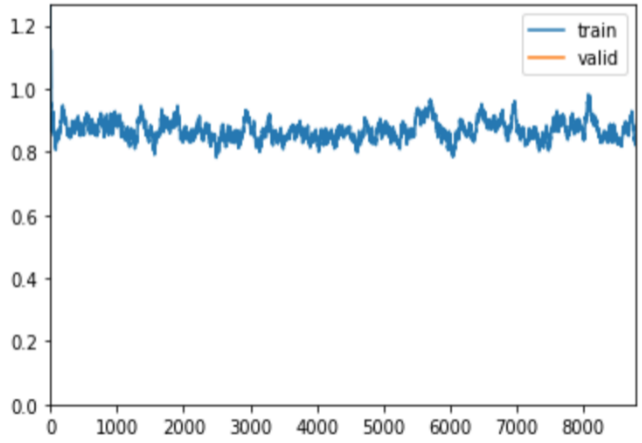

Рисунок 42 – Графік залежності похибки від кількості батчів на 3 етапі

Після повторного навчання рівень похибки зменшився з 0.82 до 0.63. Це можна побачити на рисунку 42 [32].

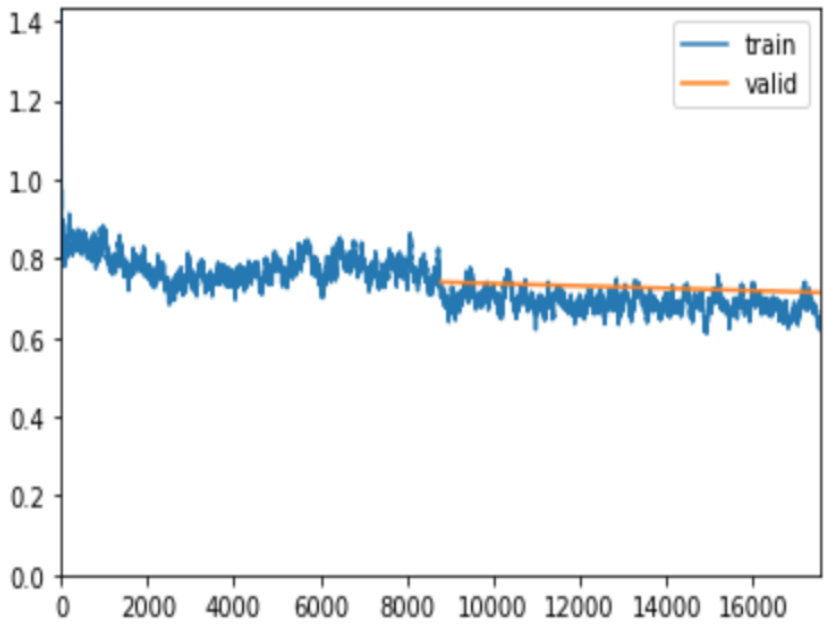

Рисунок 43 – Графік залежності похибки на фінальному етапі

В результаті, отримана модель на базі BERT має точність 0.87. Цей показник буде використано для порівняння з іншими моделями [32].

3.5 Розробка моделі на основі XLNet

Для можливості використання XLNet вхідні данні також треба токенізувати, але, на відміну від BERT, це робиться інакше [32].

– перед початком кожного речення вставити відступ – токен «[PAD]»;
– вставити токени слів речення;
– додати токен «[SEP]»
– додати токен «[CLS]».

В результаті тензор вхідної послідовності буде виглядати так, як показану на рисунку 44 [32].

```
[CLS] id : 3
[SEP] id : 4
[PAD] id : 5
Batch shape :  torch.Size([16, 94])
tensor([[   17,     9,     9,  ...,    13,     4,     3],
        [    5,     5,     5,  ...,     9,     4,     3],
        [    5,     5,     5,  ...,  1615,     4,     3],
        ...,
        [    5,     5,     5,  ..., 12330,     4,     3],
        [    5,     5,     5,  ...,   680,     4,     3],
        [    5,     5,     5,  ...,     9,     4,     3]])
```

Рисунок 44 – Тензор вхідної послідовності

Як можна побачити, токени та «[PAD]», «[SEP]» та «[CLS]» були вставлені на відповідні місця [32].

В якості попередньо тренованої моделі XLNet, що взята за основу, було використано xlnet-base-cased [32].

Вихідна конфігурація моделі наведена на рисунку 40.

Ця модель містить шар енкодингу, 12 шарів-трансформерів (XLNet) та вихідний шар класифікатору (див. рис. 45) [32].

Під час тренування моделі було застосовано методи похилого трикутного коефіцієнту навчання, дискримінативного коефіцієнту навчання та поступового розморожування шарів моделі [32].

Першим етапом навчання є заморожування останнього шару моделі (див. рис. 46), пошук оптимального коефіцієнту навчання (див. рис. 47) та навчання моделі [32].


```
XLNetConfig {
  "architectures": [
    "XLNetLMHeadModel"
  ],
  "attn_type": "bi",
  "bi_data": false,
  "bos_token_id": 1,
  "clamp_len": -1,
  "d_head": 64,
  "d_inner": 3072,
  "d_model": 768,
  "dropout": 0.1,
  "end_n_top": 5,
  "eos_token_id": 2,
  "ff_activation": "gelu",
  "id2label": {
    "0": "LABEL_0",
    "1": "LABEL_1",
    "2": "LABEL_2",
    "3": "LABEL_3",
    "4": "LABEL_4"
  },
  "initializer_range": 0.02,
  "label2id": {
    "LABEL_0": 0,
    "LABEL_1": 1,
    "LABEL_2": 2,
    "LABEL_3": 3,
    "LABEL_4": 4
  },
```


Рисунок 45 – Параметри моделі XLNet

```
[
  learner.model.transformer.transformer.word_embedding,
  learner.model.transformer.transformer.layer[0],
  learner.model.transformer.transformer.layer[1],
  learner.model.transformer.transformer.layer[2],
  learner.model.transformer.transformer.layer[3],
  learner.model.transformer.transformer.layer[4],
  learner.model.transformer.transformer.layer[5],
  learner.model.transformer.transformer.layer[6],
  learner.model.transformer.transformer.layer[7],
  learner.model.transformer.transformer.layer[8],
  learner.model.transformer.transformer.layer[9],
  learner.model.transformer.transformer.layer[10],
  learner.model.transformer.transformer.layer[11],
  learner.model.transformer.sequence_summary
 ]
```

Рисунок 46 – Структура моделі XLNet

На рисунку 47 наведено графік залежності втрати (loss) від коефіцієнту навчання (learning rate) [32].

CustomTransformerModel

Layer (type)	Output Shape	Param #	Trainable
Linear	[1, 768]	2,360,064	False
Dropout	[1, 768]	0	False
Dropout	[1, 768]	0	False
Linear	[768]	590,592	True

Рисунок 47 – Модель із замороженими шарами

На рисунку 47 можна побачити, що усі попередні шари заморожені, крім останнього [32].

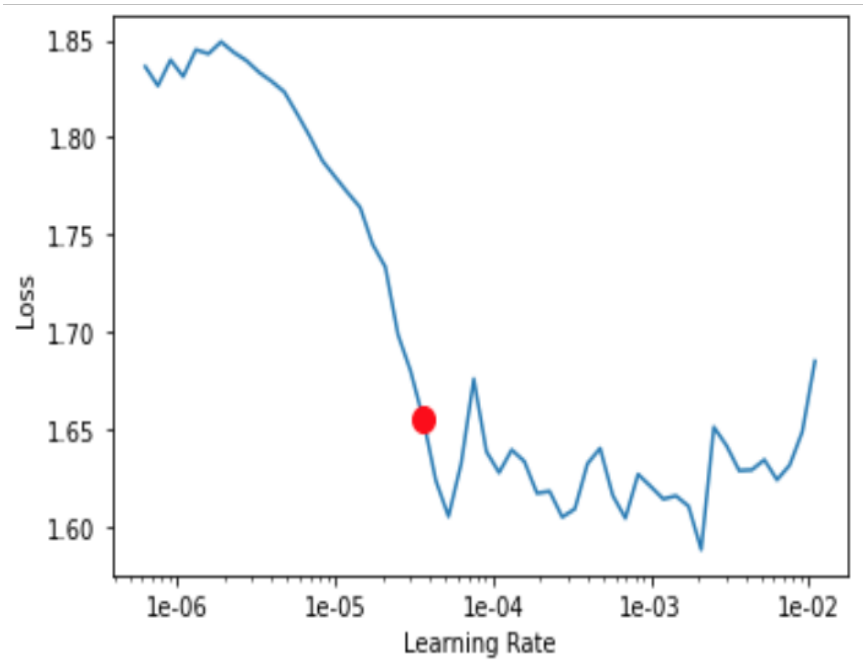

Рисунок 48 – Пошук оптимального коефіцієнту навчання

Виходячи з графіка, наведеного вище, було обрано коефіцієнт 1e-04. Після цього було запущено перший етап навчання нейронної мережі. На рисунку 48 наведено графік залежності train loss від кількості оброблених батчів [32].

Наступним кроком є повторення попередніх дій, але розморожується наступний з кінця шар моделі [35].

Як можна побачити на графіку мінімізації похибки (див. рис. 49), похибка менша ніж на попередньому етапі [35].

epoch	train_loss	valid_loss	accuracy	error_rate	time
0	1.135981	1.120599	0.546713	0.453287	1:57:24

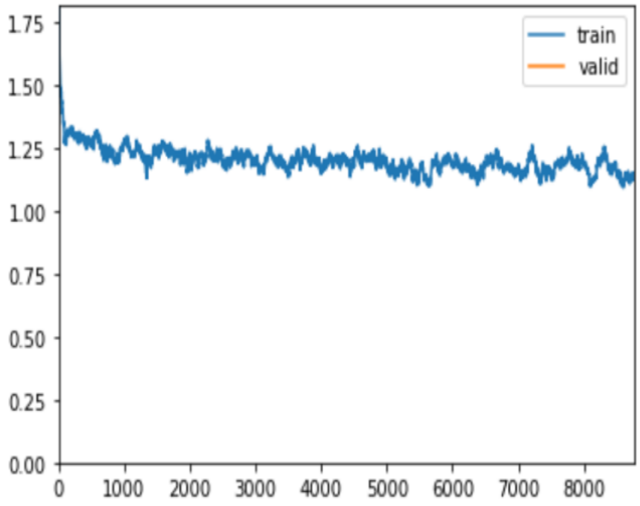

Рисунок 49 – Графік залежності похибки від кількості батчів на 1 етапі

Таким чином помітно зниження похибки з 1.13 до 1.05.

epoch	train_loss	valid_loss	accuracy	error_rate	time
0	1.052435	1.015872	0.589837	0.410163	1:51:16

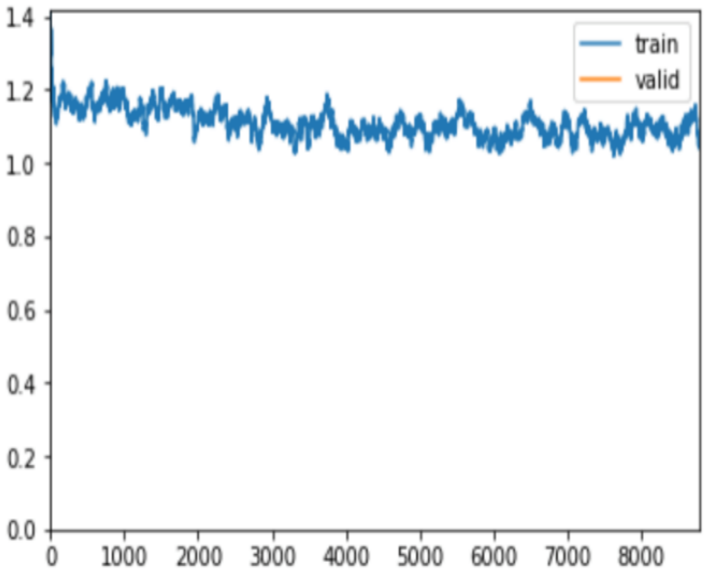

Рисунок 50 – Графік залежності похибки від кількості батчів на 2 етапі

Поступове розмороження відбувається далі і розморожується третій з кінця шар. Після цього виконується навчання. Графік похибки наведено на рисунку 50. Як видно, похибка зменшилася з 1.05 до 1.00 [35].

Після оптимізації останніх трьох шарів моделі виконується завершальне навчання моделі [35].

epoch	train_loss	valid_loss	accuracy	error_rate	time
0	1.003934	0.978852	0.591503	0.408497	2:09:48

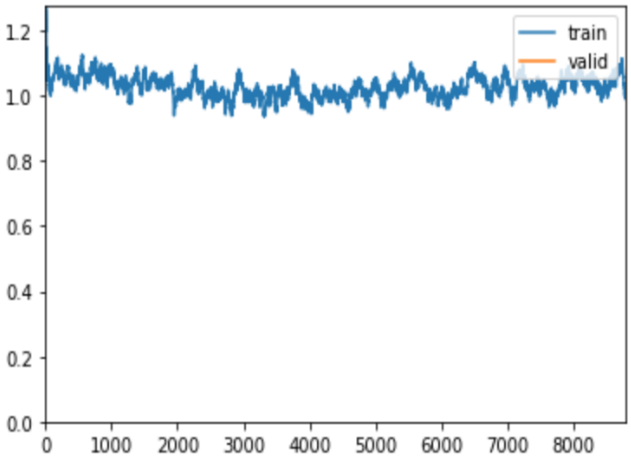

Рисунок 51 – Графік залежності похибки від кількості батчів на 3 етапі

Після повторного навчання рівень похибки зменшився з 1.0 до 0.73. Це можна побачити на рисунку 51 [35].

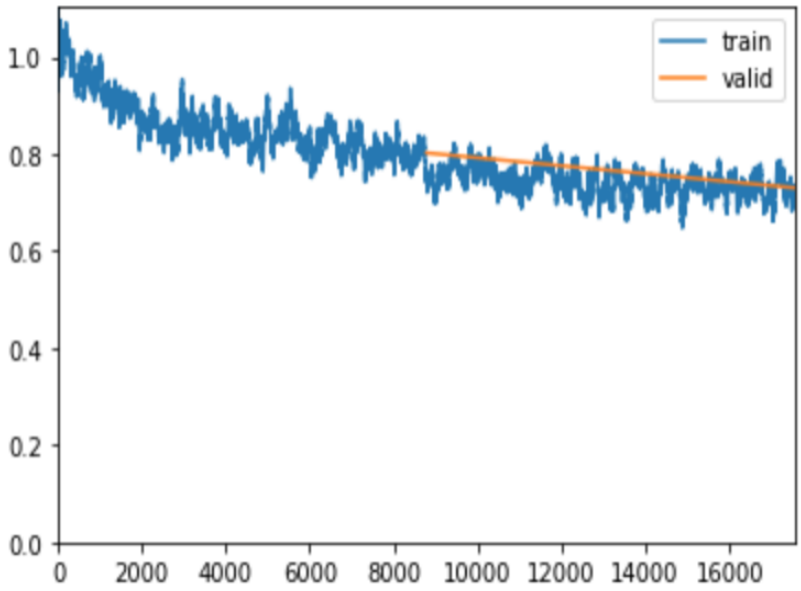

Рисунок 52 – Графік залежності похибки на фінальному етапі

В результаті, отримана модель на базі XLNet має точність 0.91. Цей показник буде використано для порівняння з іншими моделями [35].

3.6 Порівняння отриманих моделей

Оскільки метою роботи є пошук моделі, найбільш оптимальної за показником точності для вирішення задачі багатокласової класифікації сентиментів у тексті відгуків, далі буде порівнюватися саме цей показник, а також загальний час, витрачений на навчання та тестування кожної з моделей [36].

Отримані моделі було протестовано но тестовій вибірці з 31 тисячі відгуків користувачів [36].

Результати порівняння моделей наведено у таблиці 2 [36].

Таблиця 2 – Порівняння ULMFiT (LSTM), BERT та XLNet

Модель	Точність	Час навчання	Час тестування
ULMFiT (LSTM)	0.82	1 год. 3 хв.	16 с.
BERT	0.87	27 хв. 34 с.	12 с.
XLNet	0.91	34 хв. 11 с.	10 с.

Як видно з наведених результатів, найвищий показник точності показує модель на основі XLNet, що на 4% краще ніж BERT та на 8% краще ніж LSTM з ULMFiT [37].

Час навчання XLNet не є найменшим, так як він є на 6 хв. 23 с. більшим за час навчання моделі на основі BERT, просте час тестування є меншим, що відповідає результатам теоретичних досліджень у підрозділі 2.2.2, пов’язаній з XLNet. Також видно, що моделі на основі мереж-трансформерів мають помітно менший час навчання, порівняно з LSTM-моделлю, що пояснюється високою здатністю цієї архітектури до розпаралелювання, що також відповідає дослідженню с вихідних робіт про BERT та XLNet [37].

Як можна побачити з таблиці, час тестування XLNet менший за такий для BERT, що також відповідає висновкам у підрозділі 2.2.3 [27].

За результатом дослідження можна зробити висновок, що гіпотезу про те, що модель на основі XLNet буде мати найкращий результат доведено.

Таким чином, можна стверджувати, що моделі на основі XLNet є оптимальними для сентимент-аналізу відгуків [37].

Подальший шлях розвитку досліджень полягає у тонкій настройці кожної з моделей для максимальної адаптації під конкретний датасет, а також порівняння точності класифікації сентиментів цих моделей.

ВИСНОВКИ

В результаті роботи було проведено аналіз існуючих рішень для обробки природних мов, а саме пов’язаних з задачею багатокласової класифікації тексту за сентиментами.

Проведено огляд існуючих наукових робіт, пов’язаних з найкращими підходами до обробки природних мов. Було розглянуто такі методи:

– ULMFiT – метод тонкої настройки мовних моделей;
– BERT – попередньо тренована модель від Google;
– XLNet – модель, що демонструє state-of-the-art результати на більшості завдань з обробки природних мов.

Проаналізовано існуючі рішення, пов’язані з вирішенням задачі аналізу тональності тексту за допомогою вищевказаних методів.

В результаті аналізу було висунуту гіпотезу, що XLNet буде найбільш придатним методом для вирішення поставленої задачі: сентимент-аналіз відгуків споживачів про товари в інтернет магазинах.

Для доказу цієї гіпотези було проведено експериментальне дослідження. В результаті експериментального дослідження було розроблено такі моделі на базі ULMFiT, BERT та XLNet:

– LSTM з використанням методу ULMFiT;
– BERT (bert-base), натренована з виконанням поступового розмороження;
– XLNet (xlnet-base), натренована з поступовим розмороженням.

Кожну модель було натреновано на наборі даних, що містить 124 тисячі відгуків споживачів з пов'язаним з відгуком сентиментом – числовою міткою, що позначає рівень задоволеності користувача у градації «дуже негативно», «негативно», «нейтрально», «позитивно», «дуже позитивно».

За результатами аналізу розроблених моделей було проведено їх порівняння за такими метриками, як точність (accuracy), час тренування та час класифікації тестової вибірки (валідації).

Під час порівняння моделей було з'ясовано, що XLNet має найвищі показники точності серед трьох моделей (на 4% краще за BERT і на 8% краще за LSTM з

ULMFiT), але потребує більших обчислювальних потужностей для процесу тренування. Таким чином гіпотезу було підтверджено.

Подальшим розвитком роботи може бути вивчення процесу тонкої настройки (fine-tuning) моделі на базі XLNet для отримання вищого рівню точності на використаному наборі даних, або адаптація її для потреб певного інтернет магазину, який зможе використовувати її для автоматизації сентимент-аналізу відгуків споживачів.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Liddy E.D. Natural Language Processing. In Encyclopedia of Library and Information Science, 2nd Ed. NY. Marcel Decker, Inc. — 2001. — P.1.
2. COMPUTING MACHINERY AND INTELLIGENCE [Electronic resource]. – Access mode: academic.oup.com/(lastaccess:05.06.21). – Title from the screen.
3. Hutchins J. The history of machine translation in a nutshell. – 2005. – P.286.
4. Person-centeredtherapy [Electronic resource]. — Access mode: en.wikipedia.org/(lastaccess:15.11.22). – Title from the screen.
5. Transformational grammar [Electronic resource]. — Access mode: en.wikipedia.org/(lastaccess:15.11.22). – Title from the screen.
6. Turchin, Alexander; Florez Builes, Luisa F. (15.11.22). "Using Natural Language Processing to Measure and Improve Quality of Diabetes Care: A Systematic Review". Journal of Diabetes Science and Technology. 15 (3): 553–560.
7. Опис кафедри програмної інженерії // ХНУРЕ. URL: https://nure.ua/department/kafedra-programnoyi-inzheneriyi-pi (дата звернення: 03.05.2020).
8. I. Ivanov. Analysis of the phaunistic composition of Ukraine. Topical issues of the development of modern science // Abstracts of the 9th International scientific and practical conference. Софія, Болгарія: ACCENT, 2020. С. 21-27.
9. SEM1A5 A brief history of NLP. URL: https://www.cs.bham.ac.uk/~pjh/sem1a5/pt1/pt1_history.html (дата звернення: 14.11.2022).
10. Валенда Н.А., Самофалов Л.Д. Функционально-семантический анализ конструкций естественного языка на основе унификации // Системи обробки інформації. – 2016. – №7 (144). – С. 79-82.
11. Четвериков Г. Г. Формалізація принципів побудови універсальних k-значних структур мовних систем штучного інтелекту // Доповіді НАН України. – 2021. – №. 1. – С. 41.
12. Razavian Ali S., Azizpour H., Sul-livan J., Carlsson S. Cnn features off- the-shelf: an astounding baseline for recognition // In Proceedings of the IEEE conference on computer vision and pattern recognition. – 2019. – С. 806-813.

13. Peters M., Ammar W., Bhagavat-ula C., Power R. Semi-supervised sequence tagging with bidirectional language models // In Proceedings of ACL. – 2017. – C. 2.
14. Smith L. Cyclical learning rates for training neural networks. In Applications of Computer Vision (WACV) // IEEE Winter Conference. – 2017. – C. 464–472.
15. Yosinski J., Clune J., Bengio Y., Lipson H. How transferable are features in deep neuralnetworks? // Advances in neural information processing systems. – 2019. – C 3320-3328.
16. Howard J., Ruder S. Universal Language Model Fine-tuning for Text Classification. – 2018. – C. 2.
17. Radford A., Narasimhan K., Salimans T., Sutskever I. Improving language understanding with unsupervised learning // Technical report. – 2018. – C. 4.
18. Devlin J., Chang M., Lee K., Toutanova K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. – 2018. – C. 3.
19. Vaswani A., Shazeer N., Parmar N., Uszkoreit J., Jones L., Gomez A., Kaiser L., Polosukhin I. Attention is all you need // Advances in Neural Information Processing Systems. – 2017. – C. 6000-6010.
20. Open Sourcing BERT: State-of-the-Art Pre-training for Natural Language Processing // Google AI Blog. URL: https://ai.googleblog.com/2018/11/open-sourcing-bert-state-of-art-pre.html (дата звернення: 22.11.2022).
21. Dai A., Le Q. Semi-supervised sequence learning // Advances in neural information processing systems. – 2019. – C. 3079-3087.
22. Dai Z., Yang Z., Yang Y., Cohen W., Carbonell J., Le Q., Salakhutdinov R. Transformer-xl: Attentive language models beyond a fixed-length context // arXiv preprint. – 2019. – C. 3.
23. Uria B., Côté M., Gregor K., Murray I., Larochelle H.. Neural autoregressive distribution estimation // The Journal of Machine Learning Research. – 2019. – №17 (1). – C. 7184-7220.
24. Yang Z., Dai Z. Yang Y., Carbonell J., Salakhutdinov R., Le Q. XLNet: Generalized Autoregressive Pretraining for Language Understanding. – 2019. – C. 2.
25. Amazon product data. URL: http://jmcauley.ucsd.edu/data/amazon (дата звернення: 05.11.2022)

26. Dai Z., Yang Z., Yang Y., Cohen W., Carbonell J., Le Q., Salakhutdinov R. Transformer-xl: Attentive language models beyond a fixed-length context // arXiv preprint. – 2019. – С. 3.
27. Uria B., Côté M., Gregor K., Murray I., Larochelle H.. Neural autoregressive distribution estimation // The Journal of Machine Learning Research. – 2019. – №17 (1). – С. 7184-7220.
28. Yang Z., Dai Z. Yang Y., Carbonell J., Salakhutdinov R., Le Q. XLNet: Generalized Autoregressive Pretraining for Language Understanding. – 2019. – С. 2.
29. Amazon product data. URL: http://jmcauley.ucsd.edu/data/amazon (дата звернення: 05.11.2022)
30. Maas A. L., Daly R. E., Pham P. T., Huang D., Ng A. Y., Potts C. Learning word vectors for sentiment analysis // Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies. – 2021. – С. 142-150.
31. Tai K. S., Socher R., Manning C. D. Improved semantic representations from tree-structured long short-term memory networks // Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing. – 2019. – С. 1556-1566.
32. Munikar M., Shakya S., Shrestha A. Fine-grained Sentiment Classification using BERT. – 2019. – С. 2.
33. Straka M., Strakova J. Tokenizing, POS tagging, lemmatizing and parsing UD 2.0 with UDPipe // Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies, Vancouver. – 2017. – С. 43.
34. Barnes J., Ravishankar V., Øvrelid L., Velldal E. Hierarchical models vs. transfer learning for document-level sentiment classification. – 2020. – С. 3.
35. Myagmar B., Li J., Kimura S. Cross-Domain Sentiment Classification With Bidirectional Contextualized Transformer Language Models // IEEE Access. – 2019. – № 7. – С. 163219-163230.