МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ЛЬВІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ІВАНА ФРАНКА

Факультет прикладної математики та інформатики
(повне найменування назва факультету)

Кафедра прикладної математики
(повна назва кафедри)

Магістерська робота

РОЗУМІННЯ ТА ГЕНЕРУВАННЯ ПРИРОДНОЇ МОВИ ДЛЯ ПОБУДОВИ ДІАЛОГІВ ВІРТУАЛЬНОГО АСИСТЕНТА

Виконала: студентка групи ПМПм-22
спеціальності
113 Прикладна математика
(шифр і назва спеціальності)

Сидоряк В.Ю.
(підпис) (прізвище та ініціали)

Керівник доц. Музичук Ю.А.
(підпис) (прізвище та ініціали)

Рецензент доц. Недашковська А.М.
(підпис) (прізвище та ініціали)

Львів – 2022

ЛЬВІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ІВАНА ФРАНКА

Факультет прикладної математики та інформатики
Кафедра прикладної математики
Спеціальність 113 Прикладна математика
(шифр і назва)

«ЗАТВЕРДЖУЮ»
Завідувач кафедри _____

" ___ " _____ 20 ___року

З А В Д А Н Н Я

НА МАГІСТЕРСЬКУ РОБОТУ СТУДЕНТУ
Сидоряк Вірі Юріївні
(прізвище, ім'я, по батькові)

1. Тема роботи Розуміння та генерування природної мови для побудови діалогів віртуального асистента
керівник роботи Музичук Юрій Анатолійович, доцент кафедри обчислювальної математики, кандидат фізико-математичних наук
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджені Вченою радою факультету від "22" вересня 2021 року № 6

2. Строк подання студентом роботи 16 травня 2022 року

3. Вихідні дані до роботи
1. Chen, Q., Zhuo, Z., & Wang, W. (2019). BERT for Joint Intent Classification and Slot Filling. arXiv. https://doi.org/10.48550/ARXIV.1902.10909
2. Li, X., Chen, Y.-N., Li, L., Gao, J., & Celikyilmaz, A. (2017). End-to-End Task-Completion Neural Dialogue Systems. arXiv. https://doi.org/10.48550/ARXIV.1703.01008

4. Зміст магістерської роботи (перелік питань, які потрібно розробити)
1. Дослідити проблему розуміння природної мови в контексті віртуального асистента.
2. Побудувати і натренувати модель, яка класифікуватиме висловлювання та відповідні слоти.
3. Дослідити і представити оцінку точності моделі та вплив вхідних параметрів на результати класифікації.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)
1. Таблиці чисельних експериментів
2. Графічні зображення моделі машинного навчання
3. Графічні зображення матриць плутанини

6. Консультанти розділів роботи

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

7. Дата видачі завдання 02.09.2021р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	**Назва етапів магістерської роботи**	**Строк виконання етапів роботи**	**Примітка**
1.	Проведення теоретичних досліджень	Вересень - листопад 2021р	
2.	Практична реалізація алгоритмів	Грудень - березень 2022р.	
3.	Оформлення роботи	Квітень - травень 2022р	

Студент _____ **Сидоряк В.Ю.**
(підпис) (прізвище та ініціали)

Керівник роботи _____ доц. Музичук Ю.А
(підпис) (прізвище та ініціали)

ОЦІНОЧНИЙ ЛИСТ

магістерської роботи студентки Сидоряк Віри Юріївни
прізвище, ім'я, по-батькові
групи ПМПм-22 факультету прикладної математики та інформатики

Відгук наукового керівника

Тематика

X	Комп'ютерне моделювання
X	Чисельні методи
	Оптимізація процесів
	Системне програмування
	Бази даних
	Навчальні програми
	Веб-проектування
	Інше

Зміст

Максимальна кількість балів		
4	3	Складність, повнота розкриття дослідження
4	4	Наукова новизна, елементи творчості
4	3	Самостійність виконання, систематичність роботи
4	4	Якість та складність програм

Оформлення

4	3	Стиль, грамотність Ілюстративний матеріал Відповідність вимогам до роботи
Сума балів	17	

Коментарі

Задачі розуміння природньої мови при роботі з віртуальними асистентами в наш час є актуальними, як ніколи. Ми з ними стикаємось в наших мобільних телефонах, системах доповненої реальності чи навіть в автомобілях. З появою таких моделей як ELMO, BERT, GPT3, типові задачі, які виникають при роботі з текстом стало можливим розв'язати з високою точністю. Студентка Сидоряк В.Ю. продемонструвала використання моделі BERT, натренованої на українському датасеті, для розуміння намірів користувача та розпізнавання сутностей в тексті від нього. Отримані результати порівняно із результатами на класичному англійському наборі даних. Вважаю, що робота оформлена на належному рівні та заслуговує на оцінку Добре.

Науковий керівник: к.ф.-м.н. доцент Музичук Ю.А.

Оцінка рецензента _____б. Оцінка за захист _____б. СУМА БАЛІВ _____

Голова ЕК

РЕЦЕНЗІЯ

на магістерську роботу

студентки Сидоряк Віри Юріївни
прізвище, ім'я, по-батькові

групи ПМПм-22 факультету прикладної математики та інформатики

	Тематика
v	Комп'ютерне моделювання
v	Чисельні методи
	Оптимізація процесів
	Системне програмування
	Бази даних
	Навчальні програми
	Веб-проектування
	Інше

Максимальна кількість балів		**Зміст**
6	5	Складність
6	6	Наукова новизна
6	5	Повнота розкриття дослідження
6	5	Якість та складність програм
		Оформлення
2	2	Стиль, грамотність
2	2	Ілюстративний матеріал
2	2	Відповідність вимогам до роботи
Сума балів	27	

Коментарі

Робота студентки Сидоряк В. присвячена розумінню та генеруванню природної мови при побудові діалогів чат-ботів. Був створений свій набір даних, побудована модель машинного навчання і натренована на цьому наборі. На жаль, не завершеною є частина по генерації відповідей віртуального асистента, але й так робота є актуальною, грунтовною, грамотно оформленою та добре проілюстрованою. Тож заслуговує на оцінку відмінно.

Рецензент: к.ф.-м.н. доцент Недашковська А.М.

Львівський національний університет імені Івана Франка

Подання
голові екзаменаційної комісії
щодо захисту магістерської роботи

Направляється студент Сидоряк В. Ю.
(прізвище та ініціали)
до захисту магістерської роботи
за спеціальністю 113 Прикладна математика
(шифр і назва спеціальності)
на тему Розуміння та генерування природної мови
для побудови діалогів віртуального асистента
(назва теми)

Магістерська робота і рецензія додаються.
Декан факультету _____
(підпис)

Довідка про успішність

_____
(прізвище та ініціали студента)
за період навчання на факультеті прикладної математики та інформатики
з 20___ року до 20___ року повністю виконав навчальний план
з таким розподілом оцінок за:
національною шкалою:
відмінно_____%, добре_____%, задовільно_____%;
шкалою ECTS:
A_____%; B_____%; C_____%; D_____%; E_____%

Секретар факультету_____ _____
(підпис) (прізвище та ініціали)

Висновок керівника магістерської роботи

Студент (ка)__Сидоряк В.Ю. виконала всі завдання, які були поставлені перед нею на цю роботу. Вона дослідила задачі розуміння природньої мови у контексті роботи віртуального асистента, ознайомилася із наявними алгоритмами для розв'язування таких задач, реалізувала необхідну функціональність та натренувала модель машинного навчання на наборі даних, перекладеному на українську мову із наявного модельного датасету англійською мовою. Сидоряк В.Ю. продемонструвала самостійну роботу та отримала хороші результати чисельних експериментів. Для завершеної роботи віртуального асистента залишається доробити частину генерації природньої мови, але це можна відкласти на подальші дослідження. Робота заслуговує на добру оцінку.

Керівник роботи _____
(підпис)

"15" травня 2022 року

Висновок кафедри про магістерську роботу

Магістерська робота розглянута.

Студент(ка) _____
(прізвище та ініціали)

допускається до захисту даної роботи в Екзаменаційній комісії.

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

"_____"_____ 2022 року

Зміст

Вступ

Вперше термін «чатер-бот» (англ. ChatterBot) був використаний Майкл Молдін у 1994 році, щоб описати розмовні програми, які дозволяють спілкування між людиною і машиною зі штучним інтелектом. Існує багато визначень чат-бота або віртуального асистента. Відповідно до Кембриджського словника:

Чат-бот - це комп'ютерна програма, яка може відповідати на повідомлення користувачів переважно через інтернет. [2]

Основна ідея кожного чат-бота полягає в тому, щоб взаємодіяти з користувачем за допомогою текстових повідомлень і поводитися так, ніби чат-бот здатний зрозуміти розмову та відповісти користувачеві.

Чат-боти, як і багато інших технологій, з'явилися багато років тому, проте набрали популярності лише протягом останніх років. Концепцію чат-ботів представив Алан Тюринг ще у 1950 році. У його статті «Обчислювальна техніка та інтелект» (англ. Computing Machinery and Intelligence)[10] було запропоновано тест Тюринга як критерій штучного інтелекту. У цьому тесті кілька людей повинні ставити питання двом таємним співрозмовникам і на підставі відповідей визначати, хто з них машина, а хто людина. Якщо машину розкрити не вийде, це означає, що машина розумна. У 1966 році була представлена перша комп'ютерна програма під назвою ELIZA, яка могла спробувати пройти тест Тюринга. З розвитком обчислювальних можливостей чат-боти ставали якіснішими. А справжній бум наступив у 2016 році, коли Facebook дозволив створювати чат-боти на платформі Messenger. За перший рік було створено понад 100 000 ботів, оскільки велика кількість компаній на платформі Facebook прагнули використовувати нову технологію.

Загалом існує два типи чат-ботів, залежно від характеру розмови: загального характеру й орієнтовані на ціль. Віртуальні асистенти загального характеру можуть спілкуватись на будь-яку тему, тому їх важче реалізувати. Прикладами таких систем є Google Assistant, Alexa від Amazon та Siri від Apple.

Чат-боти з визначеною ціллю зосереджуються лише на декількох сфе-

рах і, як правило, мають заздалегідь визначений намір. Наприклад, це може бути система діалогу бронювання столиків у ресторанах, яка допомагає користувачам забронювати столик найбільш зручним способом, тобто шляхом розмови. Зазвичай, їх легше запрограмувати. Але варто зазначити, що орієнтованого на ціль чат-бота, дуже складно перевчити на іншу сферу застосування. Ця робота буде сфокусована на віртуальних асистентах орієнтованих на ціль.

Для побудови віртуального асистента потрібно опрацьовувати природню мову. Тому ознайомимось з важливими термінами в обробці природної мови.

NLP (англ. Natural language processing - обробка природної мови) — це загальний термін, що використовується для опису здатності машини сприймати те, що їй сказано, розуміти її значення, визначати відповідні дії та відповідати мовою, яку зрозуміє користувач.

NLU (англ. Natural language understanding - розуміння природної мови) — це підмножина NLP, яка має справу з обробкою неструктурованих вхідних даних та перетворення їх у структуровану форму, яку зможе зрозуміти машина. Наприклад, люди можуть легко справитись з неправильною вимовою, зміненими словами, скороченнями, розмовними висловлюванням, у той час як для машини це може бути надзвичайно складним завданням.

NLG (англ. Natural language generation - генерація природної мови) - це те, що відбувається, коли комп'ютери пишуть мову. Процеси NLG перетворюють структуровані дані на текст.

Поточні підходи до задачі NLU

Однією з найважливіших частин кожного віртуального асистента є модуль розуміння природної мови. Тому темою цієї роботи є розуміння природної мови для побудови віртуального асистента. Протягом останніх десятиліть системи NLU розвивались надзвичайно швидко. Загалом виділяють дві задачі NLU: ідентифікація намірів та заповнення слотів. Ці дві задачі зазвичай розглядають паралельно, але зараз стрімко розвиваються підходи, де одна задача має вплив на іншу. Тому підходи можна класифікувати як

незалежні та спільні.

Незалежний підхід. Розпізнавання намірів можна розглядати як проблему класифікації висловлювань. В попередніх роботах застосовували такі популярні класифікатори, як метод опорних векторів(SVM) [6] та різні модифікації нейронних мереж: згорткові нейронні мережі(CNN) у [15] та [16], довга короткочасна пам'ять(LSTM)[14] та інші. Заповнення слотів розглядали як задачу маркування послідовності. Популярними підходами до вирішення проблеми заповнення слотів були метод умовних випадкових полів(CRF) [8], рекурентні нейронні мережі(RNN)[7], довга короткочасна пам'ять(LSTM).

Спільний підхід. Спільні підходи полягають у побудові зв'язку між розпізнаванням слотів та класифікації намірів. Попередні праці часто використовували поєднання кількох методів: наприклад, згорткові нейронні мережі + метод умовних випадкових полів(CNN+CRF) [13]. На сьогодні, дуже ефективним є використання попередньо навчених моделей, наприклад у JoinBERT [3] було використано модель BERT.

Мета роботи

Об'єктом нашого дослідження виступає задача розуміння природної мови в контексті віртуального асистента.

Предметом дослідження є методи та засоби штучного інтелекту, що застосовуються в даній сфері та можуть бути покладені в основу моделі.

Метою дослідження є

- дослідити задачу NLU в контексті віртуального асистента;
- побудова українського датасету (на базі датасету ATIS);
- порівняння різних алгоритмів для вирішення задачі NLU на цьому наборі даних.

Зокрема буде розглянуто різні модифікації алгоритмів, побудованих на моделі BERT. За основу будуть взяті моделі представлені у [3] та [4].

1 Модель віртуального асистента

Задачею чат-бота є отримання повідомлень від користувача, аналіз їх в контексті цілого діалогу та генерація відповідей. Для вирішення цієї проблеми віртуальний асистент складається з трьох блоків:

- Блок розуміння природної мови (NLU)
- Менеджер діалогу (DM)
- Блок генерації природної мови (NLG)

Оскільки бот не розуміє людську мову, потрібен спосіб перекладу з нашої природної та неструктурованої мови на структуровані дані, які може обробити та зрозуміти машина. За цей крок відповідає модуль розумінням природної мови (NLU). На вході NLU модуль отримує повідомлення користувача, а на виході повертає намір, який хоче зробити користувач, у структурованому вигляді.

Наступним кроком є вибір найкращої дії. Цей процес виконується компонентою під назвою Dialog Manager (DM). На вході DM отримує структуровані дані з NLU, аналізує їх в контексті цілого діалогу, тобто попередніх повідомлень, та вибирає найкращу дію. Ця компонента діалогу часто спілкується з базою даних або API. На виході DM повертає певну дію, у структурованому форматі.

Останнім кроком є перетворення структурованої відповіді в природню мову. Цей процес відомий як генерація природної мови (NLG). На вході NLG компонент отримує певну дію, обробляє її і повертає користувачеві відповідь природною мовою (у вигляді повідомлення).

На рисунку 1.1 можна побачити змодельовану систему діалогу для бронювання столиків в ресторані.

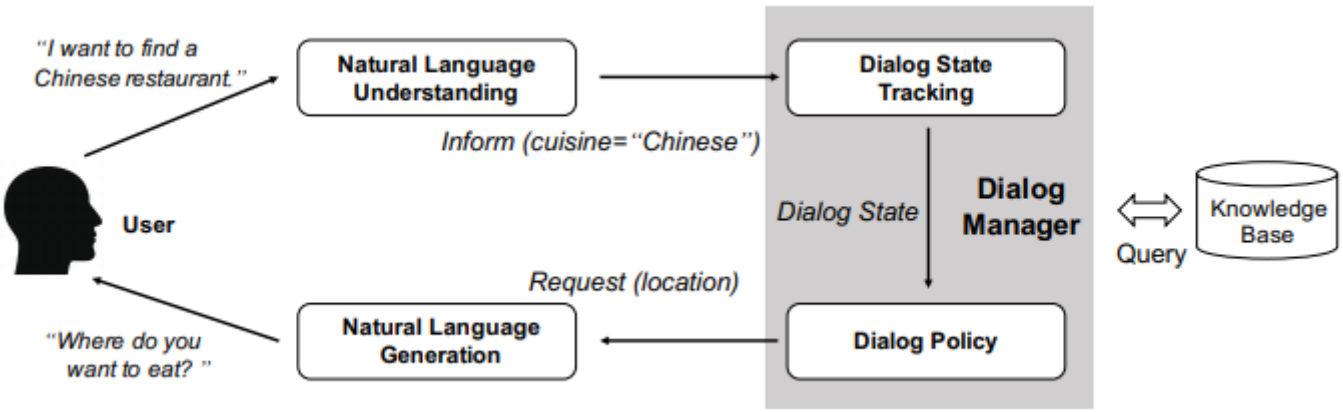

Рис. 1.1: Система діалогу [18]

1.1 Задача NLU

Розглянемо основні терміни. У NLP жаргоні ці терміни позначають ключові компоненти системи, які використовуються для класифікації, синтаксичного аналізу та іншої обробки природної мови.

Висловлювання (utterance) вводяться користувачем, і програма має інтерпретувати їх.

Намір (intent) — ціль користувача, яку він хоче досягти даним висловлюванням.

Сутність (entity) — це змінна, яка стосується наміру і дозволяє зрозуміти інформацію про запит.

Щоб навчити програму визначати наміри та сутності з них, важливо фіксувати різноманітні приклади висловлювань для кожного наміру. Висловлювання використовуються для побудови та навчання намірів і, тому не повинні включати кілька намірів або неоднозначних значень.

У цій роботі буде вважатись, що одне висловлювання містить один намір.

Приклад кожного терміну можна бачити на рисунку 1.2. Тому можна визначити такі задачі NLU:

- визначення наміру;
- заповнення слотів.

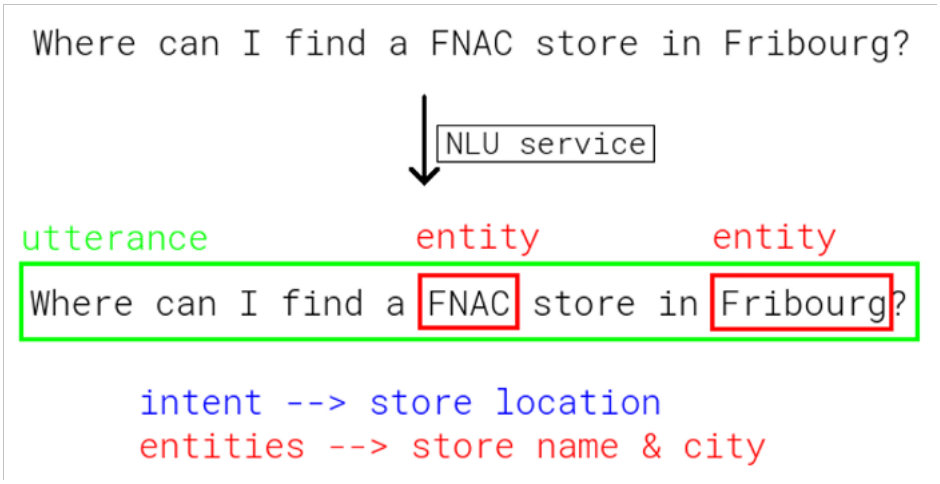

Рис. 1.2: Приклад аналізу виразу

1.2 Задача DM

Менеджер діалогу відповідає за вибір наступної дії. Як правило задачу DM вирішують шляхом навчання з підкріпленням (Reinforcement learning), а сам процес описують як процес прийняття рішень Маркова (MDP).

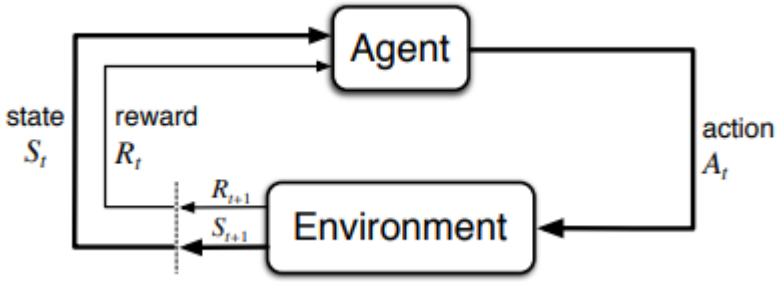

Рис. 1.3: Модель процесу прийняття рішень Маркова

MDP - це набір $< \mathrm{S}, \mathrm{A}, \mathrm{P}, \mathrm{R}, \gamma >$, де:

1. S - множина можливих станів, S_t - стан на кроці t;
2. A - множина можливих дій (A_s - множина можливих дій з стану s);
3. $\mathrm{P}(s' \mid s, a)$ - ймовірність того, що на дія a в стані s призведе до стану s' на $(t+1)$-кроці;
4. $\mathrm{R}(s, a)$ - винагорода отримана після переходу зі стану s в стан s';
5. γ - коефіцієнт знецінювання (англ. discount factor), задовольняє $0 \leq \gamma \leq 1$.

Мета агента - знайти таку стратегію π, яка максимізує суму його майбутніх винагород.

$$R_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1}, \tag{1.1}$$

Відповідно до описаного вище, менеджер діалогу складається з наступних двох компонентів: менеджера станів (Dialog State Tracking) і агента для вибору дії, назвемо його агент RL, це також можна побачити на рис. (1.1). DST відповідає за збереження станів, а агент RL за вибір наступної дії. [18]

Ця задача була розв'язана у моїй попередній роботі [20] з використанням Q-learning, яке представляє таблицю станів, та глибокого Q-learning, яке полягає у використанні нейронної мережі, замість таблиці станів.

1.3 Задача NLG

Блок NLG є сполучною частиною між системою і користувачем. Враховуючи реакцію системи як семантичний кадр, вона повертається до речення природною мовою, зрозумілого для кінцевого користувача.

Компонент NLG може бути базуватись на правилах або на моделі машинного навчання. У деяких випадках це може бути гібридна модель, тобто комбінація правил та моделі.

Система, яка базується на правилах, виводить користувачеві деякі заздалегідь визначені шаблонні речення для даного наміру. Загалом системи такого типу доволі обмежені. А блоки NLG, які базуються на моделях машинного навчання, є більш гнучкими і різноманітними. Зазвичай для навчання моделей використовується навчання з учителем, тобто є датасет з маркованими даними. Моделі, представлені в [11], [12], використовують архітектуру рекурентних нейронних мереж. І останнім кроком відповідне речення заповнюється актуальними даними по сутностях. Високорівнево система зображена на рисунку 1.4.

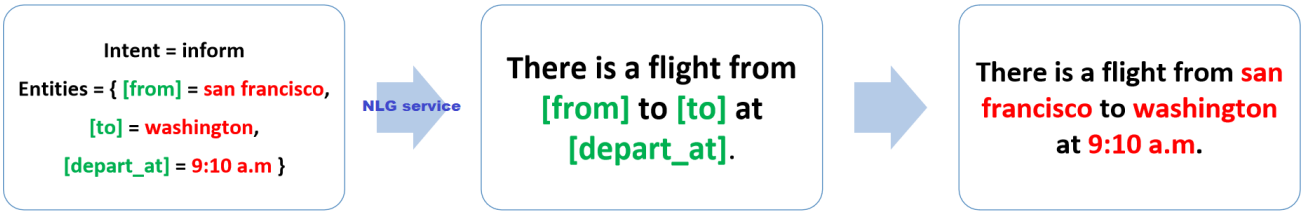

Рис. 1.4: Приклад аналізу виразу

2 Архітектура моделі

В цьому розділі буде сфокусована увага на теоретичній частині вирішення задачі NLU. Буде розглянуто спільні моделі, які базуються на попередньо навченій моделі BERT. Модель містить:

- частину, яка відповідає за визначення намірів
- частину, яка відповідає за визначення слотів
- механізм уваги, який дозволяє використовувати намір для визначення слотів.

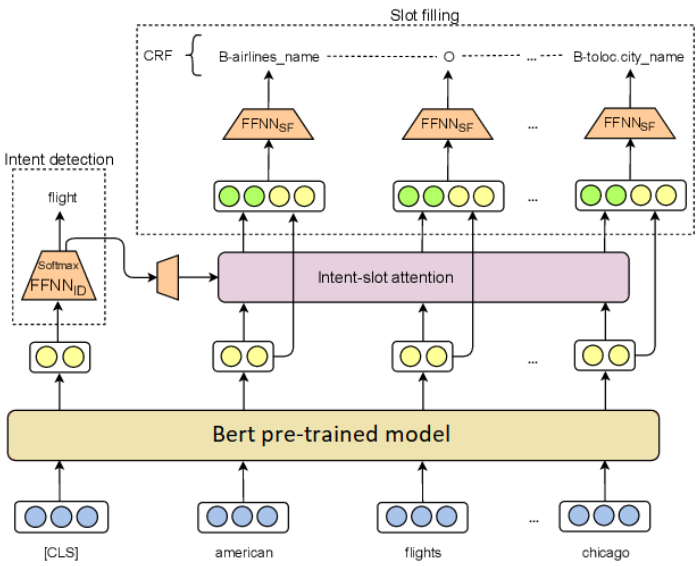

Рис. 2.1: Архітектура моделі

2.1 Токенізація

Для застосування обраної моделі BERT, потрібно провести попередню підготовку даних, а саме провести токенізацію.

Токенізація — це спосіб поділу фрагмента тексту на менші одиниці, які називаються токенами. Тут маркери можуть бути словами, символами або підсловами.

Для застосування BERT токенізація виконується алгоритмом WordPiece. На початок висловлювання додається спеціальний токен класифікації ([CLS]), а як кінцевий маркер додається спеціальний маркер ([SEP]).

2.2 Застосування BERT

Наступним кроком є застосування попередньо навченої моделі BERT до висловлювання. Тобто маємо висловлювання, яке складається з n токенів $w_1, w_2, ..., w_n$. Після додавання спеціальних маркерів отримане вхідне висловлювання буде мати вигляд: $w_0, w_1, w_2, ..., w_n$ (де w_0 — це «[CLS]»).

$$h_i = \text{BERT}\left(w_{0:n}, i\right) \tag{2.1}$$

BERT було створено й опубліковано 2018 року Джейкобом Девліним та його колегами з Google для вирішення задачі NLP [1]. BERT (англ. Bidirectional Encoder Representations from Transformers) — це методика машинного навчання, яка ґрунтується на механізмі трансформерів. Транформер містить кодер, який зчитує дані, та декодер, який генерує результат. На рис. (2.2) зображену архітектуру моделі transformers.

Модель BERT пропонує два наперед натреновані варіанти:

- модель $BERT_{BASE}$, нейромережна архітектура з 12 шарами, 768 прихованими, 12 головами, 110 мільйонами параметрів;
- модель $BERT_{LARGE}$, нейромережна архітектура з 24 шарами, 1024 прихованими, 16 головами, 340 мільйонами параметрів.

Обидві моделі треновано на BooksCorpus з 800 мільйонами слів, та статтях з Вікіпедії з 2 500 мільйонами слів.

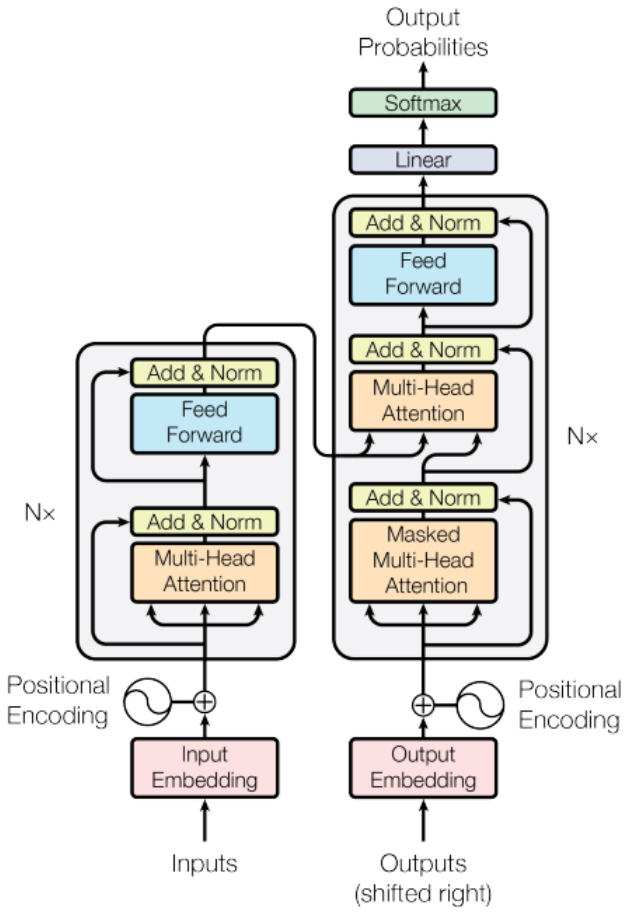

Рис. 2.2: Архітектура моделі transformers [1]

2.3 Визначення намірів

Модуль для визначення намірів представляє лінійну модель, яка накладається поверх визначеного h_0 з 2.1. На виході використаємо функцію softmax для перетворення у ймовірності.

$$\hat{p} = \text{softmax}\left(\mathbf{W}\boldsymbol{h}_0 + \boldsymbol{b}\right) \tag{2.2}$$

де $\hat{p} \in \mathbb{R}^k$, де k - кількість можливих намірів.

Функція Softmax - це узагальнення логістичної функції, що «стискує» K-вимірний вектор z із довільними значеннями компонент до K-вимірного вектора $\sigma(z)$ з дійсними значеннями компонент в області [0, 1], які дають в сумі одиницю.

Штрафну функцію визначимо як крос ентропію:

$$\mathcal{L}_{ID} = -\sum_{i=1}^{\text{k}} p_i \cdot \log \hat{p}_i \tag{2.3}$$

де $\hat{p}_i$ - наближене значення, а p_i - точне.

2.4 Механізм уваги

Механізм уваги потрібен, щоб узгодити інформацію між намірами і слотам. Спершу створюється залежність між намірами і слотами, за допомогою множення матриці ваг $W \in \mathbb{R}^{dxk}$ на вектор ймовірностей для намірів $p \in \mathbb{R}^k$, де d - кількість можливих слотів, k - кількість можливих намірів.

$$\mathbf{w} = \mathbf{W}\mathbf{p} \tag{2.4}$$

Після цього цей вектор використовується для створення послідовності векторів c_i, де $i = 1, ..., n$.

$$\alpha_i = \frac{\exp\left(\mathbf{w}_\top \mathbf{h}_i\right)}{\sum_{j=1\ n} \exp\left(\mathbf{w}_\top \mathbf{h}_j\right)} \tag{2.5}$$

$$c_i = \alpha_i \mathbf{w} \tag{2.6}$$

2.5 Визначення слотів

Для початку побудуємо послідовність вхідних векторів $v_{i:n}$, де v_i - кожне слово, для якого потрібно визначити слот. Для цього існує кілька підходів.

Модель з [4] на сьогодні дає найкращі результати. У ній вектор v_i визначається як

$$v_i = c_i \odot h_i \tag{2.7}$$

У моделі, представленій у [17], вектор v_i визначений як:

$$v_i = h_0 \odot h_i \tag{2.8}$$

Для базової архітектури з [3] вектор v_i має вигляд:

$$v_i = h_i \tag{2.9}$$

Варто також зазначити, що для 2.8 та 2.9 не потрібен механізм уваги. Ці різні імплементації вектора v_i буде порівняно в частині результатів.

До вектора v_i застосуємо лінійну модель з функцією softmax.

$$s_i = \text{softmax}(Wv_i + b) \tag{2.10}$$

Зауважте, що матриці вагових коефіцієнтів W та вільні члени b з 2.10 та 2.2 - різні.

Штрафну функцію визначимо, як крос ентропію, аналогічно до 2.3.

$$\mathcal{L}_{SF} = -\sum_{i=1}^{\text{k}} s_i \cdot \log \hat{s}_i \tag{2.11}$$

де $\hat{s}_i$ - наближене значення, а s_i - точне.

2.6 Ціль і штрафна функція об'єднаної моделі

Кінцева штрафна функція для тренування моделі – це зважена сума штрафних функції $\mathcal{L}_{\text{ID}}$ та $\mathcal{L}_{\text{SF}}$:

$$\mathcal{L} = \lambda\mathcal{L}_{\text{ID}} + (1-\lambda)\mathcal{L}_{\text{SF}} \tag{2.12}$$

де гіпер-параметр λ належить діапазону: $(0, 1)$.

2.7 Метод умовних випадкових полів (CRF)

У статті [19] Zhou and Xu показали як покращити результати для задачі розпізнавання іменованих сутностей додавши шар CRF до BiLSTM. Тому зараз більшість моделей [3], [4] також пробують покращити результати таким способом.

Умовні випадкові поля(англ. conditional random fields CRFs) — це клас методів статистичного моделювання, які часто застосовують в розпізнаванні образів та машинному навчанні, й використовують для структурового передбачування.

Спочатку визначимо деякі позначення:

- Навчальні дані: пари вхідного та вихідного векторів (X_i, y_i);
- і-ий вхідний вектор: $X_i = [x_1, x_l]$;

- і-ий вихідний вектор: $y_i = [x_1, x_l]$;
- l - довжина послідовності.

Оскільки вхідними параметрами для задачі класифікації слотів є текст (тобто послідовність слів), ми також можемо перетворити текст у лінійний ланцюжок (простий орієнтований граф, де кожне слово є вузлом, а між сусідніми словами є вершини). Використовуючи CRF, ми можемо змоделювати таку умовну ймовірність:

$$P(y \mid x) = \prod_{t=1}^{n} P(y_t \mid x_t) P(y_t \mid y_{t-1}) \tag{2.13}$$

де $P(y_t \mid x_t)$ - ймовірність мітки для токена x_t.

$P(y_t \mid y_{t-1})$ - ймовірність переходу (y_{t-1}, y_t).

Цю ймовірність можна переписати в експоненційній формі. Введемо позначення U, T, Z, так що умовна ймовірність:

$$P(\mathbf{y} \mid \mathbf{x}) = \frac{\exp\left(\Sigma_{t=1}^{n} U\left(y_t, x_t\right) + \Sigma_{t=1}^{n} T\left(y_t, y_{t-1}\right)\right)}{Z(\mathbf{x})} \tag{2.14}$$

де $U\left(y_t, x_t\right)$ оцінка викидів для токена x_t з значенням y_t;

$T\left(y_t, y_{t-1}\right)$ значення переходу y_{t-1} до y_t;

$Z(\mathbf{x})$ називається функцією розподілу і служить нормалізацією для отримання ймовірностей.

Щоб натренувати CRF потрібно штрафну функцію, яка оптимізує модель, таким чином, щоб ймовірність прямувала до 1. Тому штрафна функція буде виглядати наступним чином:

$$\text{loss} = -\log\left(\frac{\exp\left(\sum_{t=1}^{n} U(y \mid x) + \sum_{t=1}^{n} T\left(y_t \mid y_{t-1}\right)\right)}{Z(\mathbf{x})}\right) \tag{2.15}$$

Однак обчислення знаменника (функції розподілу) є набагато складнішим, оскільки існує експоненціальна кількість можливих послідовностей. Для вирішення цієї проблеми є два алгоритми «forward-backward» та Вітербі.[5]

$$Z(x) = \Sigma_{y'} \exp\left(\Sigma_{t=1}^{n} U\left(y'_t \mid x_t\right) + \Sigma_{t=1}^{n} T\left(y'_t \mid y'_{t-1}\right)\right) \tag{2.16}$$

3 Програмна реалізація

3.1 Датасет

Для тестування алгоритмів вибрано популярний датасет ATIS [9]. Набір даних ATIS — це стандартний контрольний набір даних, який широко використовується для класифікації намірів і заповнення слотів.

ATIS (Airline Travel Information Systems) розшифровується як система інформації про подорожі авіакомпаній. Його було перекладено на українську мову. Для того, щоб датасет був природнім, міста, назви аеропортів та авіакомпаній було замінено на такі, які є в Україні або всесвітньо відомі.

Базові статистики датасету:

	train	test
Кількість зразків	4978	893
Кількість слотів	129	129
Кількість намірів	26	26

Табл. 1: ATIS англійська версія датасету

	train	test
Кількість зразків	2000	500
Кількість слотів	129	129
Кількість намірів	26	26

Табл. 2: ATIS українська версія датасету

На рис.3.4 показано приклад даних.

На рисунку 3.2 і 3.3 можна побачити, які наміри і слоти представлені в датасеті.

```
покажіть мені рейси які прибувають до дніпра чотирнадцятого червня
які рейси відправляються з торонто в київ через мюнхен і прибувають до 21:00
які авіакомпанії здійснюють рейси з львова до вашингтона через інші міста
я шукаю рейс зі львова до барселони з зупинкою у варшаві сподіваюся на рейс ввечері які є
добре, а потім четвертого вересня з варшави я хотів би поїхати до мадриду
покажіть мені всі рейси з берліна до нью-йорка
добре я хотів би полетіти з києва в дубаї вдень що є в наявності
райнеїр зі львова до варшави вранці
які види наземного транспорту існують до аеропорту сан франциско
у наступні два дні я хочу полетіти з харкова в берлін або у відень
чи літає мау з києва в ужгород
одеса київ
```

Рис. 3.1: Приклад даних

```
abbreviation
aircraft
aircraft+flight+flight_no
airfare
airfare+flight
airfare+flight_time
airline
airline+flight_no
airport
capacity
cheapest
city
day_name
distance
flight
flight+airfare
flight+airline
flight_no
flight_no+airline
flight_time
ground_fare
ground_service
ground_service+ground_fare
meal
quantity
restriction
```

Рис. 3.2: Наміри з датасету

```
B-aircraft_code
B-airline_code
B-airline_name
B-airport_code
B-airport_name
B-arrive_date.date_relative
B-arrive_date.day_name
B-arrive_date.day_number
B-arrive_date.month_name
B-arrive_date.today_relative
B-arrive_time.end_time
B-arrive_time.period_mod
B-arrive_time.period_of_day
B-arrive_time.start_time
B-arrive_time.time
B-arrive_time.time_relative
B-booking_class
B-city_name
```

Рис. 3.3: Приклад слотів з датасету

На рис. 3.4 показано кількість виразів з відповідними намірами на train частині українського датасету, видно, що намір atis_flight суттєво переважає над іншими.

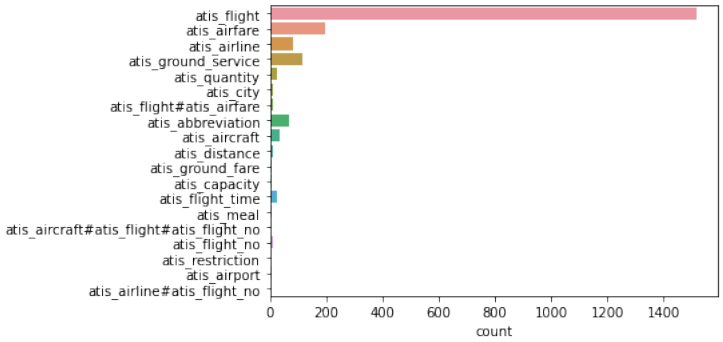

Рис. 3.4: Розподіл кількості намірів

У таблиці 3 приклад виразу з його наміром і відповідними слотами .

Intent	Entity	Slot
	покажіть	O
	мені	O
	рейси	O
	які	O
atis_fligth	прибувають	O
	до	O
	дніпра	B-toloc.city_name
	чотирнацятого	B-arrive_date.day_number
	червня	B-arrive_date.month_name

Табл. 3: приклад виразу з датасету

3.2 Метрики для оцінки точності

Для порівняння моделей означимо метрики точності. Введемо поняття матриці плутанини (confusion matrix). Кожен з рядків цієї матриці представляє зразки прогнозованого класу, тоді як кожен зі стовпців представляє зразки справжнього класу.

Для двох класів: правдивого(true) і хибного(false), матриця буде виглядати таким чином.

Актуальний \ Передбачений клас	True	False	total
True	TP	FN	T’
False	FP	TN	N’
total	T	N	

Табл. 4: Матриця плутанини для двох класів

Де

TP [True positive] = правдиві значення, які класифіковано правильно;

FN [False negative] = правдиві значення, які класифіковано як хибні;

FP [False positive] = хибні значення, які класифіковано як правдиві;

TN [True negative] = хибні значення, які класифіковано правильно.

Дані задачі (класифікація намірів та ідентифікація слотів) є задачами багатокласової класифікації, тому матриця плутанини для багатокласової класифікації буде розмірності NxN, де N - кількість класів. А значенням в кожній клітинці матриці (i, j) буде кількість значень, яким передбачили j-ий клас, а вони відносяться до i-го класу.

Приклад матриці для N=3 класів.

Актуальний \ Передбачений клас	Клас 1	Клас 2	Клас 3
Клас 1	C_{11}	C_{12}	C_{13}
Клас 2	C_{21}	C_{22}	C_{23}
Клас 3	C_{31}	C_{32}	C_{33}

Табл. 5: Матриця плутанини для 3-ох класів

Важливо зазначити, що при хорошій точності моделі, матриця плутанини буде мати суттєве діагональне переважання. Основні формули для **оцінки точності**:

$$\text{Загальна точність (accuracy)} = \frac{TP + FP}{TP + FP + FN + TN} \tag{3.1}$$

$$\text{Точність (precision)} = \frac{TP}{TP + FP} \tag{3.2}$$

$$\text{Повнота (recall)} = \frac{TP}{TP + FN} \tag{3.3}$$

$$\mathrm{F}_1 = 2 \cdot \frac{\mathrm{precision} \cdot \mathrm{recall}}{\mathrm{precision} + \mathrm{recall}} \tag{3.4}$$

Так як в цій роботі розглядаються дві задачі одночасно, то буде додаткова метрика, яка обраховує точність класифікації для цілого речення.

3.3 Програмна реалізація і налаштування параметрів

Код програми був реалізований на мові програмування python.

Для тренування на українському датасеті вибрано bert-base-multilingual-uncased, яка була натренована на статтях із Вікіпедії.

Для порівння на англійській версії датасету вибрано bert-base-uncased. В попередніх статтях використовували також моделі distilbert-base-uncased та albert-xxlarge-v.

Для Bert було використано такі параметри:

- max_seq_len = 50
- batch_size = 32
- dropout_rate = 0.1
- learning_rate_adam = 10^{-5}

max_seq_len - максимальна загальна довжина вхідної послідовності після токенізації.

dropout_rate - це параметр для методу регуляризації , який запобігає перенавчанню нейронної мережі. Цей метод регуляризації полягає у ігнорування певної кількості нейронів під час фази навчання. На кожному етапі навчання окремі вузли або вилучаються з мережі з імовірністю 1-p, або зберігаються з ймовірністю p, так що залишається зменшена мережа; вхідні та вихідні ребра до випавшого вузла також видаляються.

learning_rate_adam - це початкова швидкість навчання для оптимізатора адама.

Кількість епох = 10. У термінології нейронних мереж епоха (epoch) = одна ітерація у процесі тренування, тобто коли весь датасет пройде через нейронну мережу один раз. Навчання однієї епохи у середовищі Google Colab займає приблизно 35 хв.

На рис. 3.5 зображено графік зміни точності в залежності від ітерації(епохи).

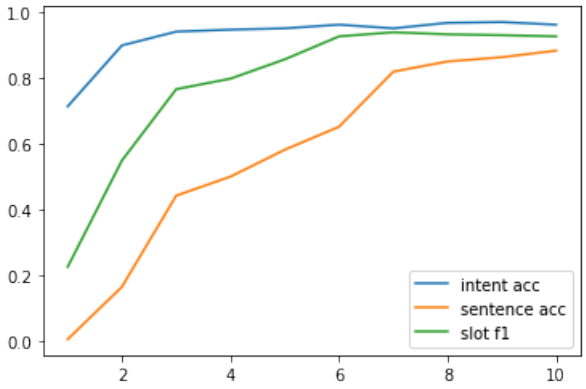

Рис. 3.5: Графік зміни точності в залежності від епохи

3.4 Результати

У таблиці 6 представлено результати роботи алгоритмів на двох датасетах. Можна побачити, що для англійського датасету результати є кращими, це може бути зумовлено більшим датасетом або якістю перекладу. Також видно, що CRF покращує модель для обох датасетів.

Для українського датасету найкращі результати показує $v_i = h_i$ з CRF, тому розглянемо детальніше цю модель.

На рис. 3.4 можна бачити матриці плутанини для намірів для англійської і української версій датасету. Видно, що atis_flight суттєво переважає над іншими і помилки зроблені в більшості саме в цьому класі.

Датасет	Метод	Intent Accuracy	Slot F1	Sentence Accuracy
Англійський	$v_i = h_i$	97.87	95.59	88.24
Англійський	$v_i = h_i$ з CRF	97.98	95.93	88.58
Англійський	$\mathrm{v}_i = h_0 \odot h_i$	98.35	96.78	88.85
Англійський	$\mathrm{v}_i = c_i \odot h_i$	98.45	97.03	89.55
Український	$v_i = h_i$	95.25	92.38	86.05
Український	$v_i = h_i$ з CRF	95.98	93.79	86.52
Український	$\mathrm{v}_i = h_0 \odot h_i$	96.92	93.45	86.37
Український	$\mathrm{v}_i = c_i \odot h_i$	95.62	92.91	85.98

Табл. 6: Таблиця результатів

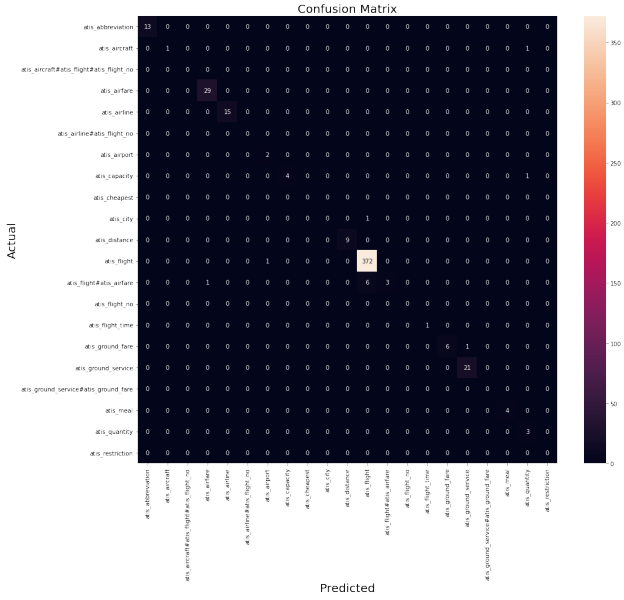

Рис. 3.6: Український датасет

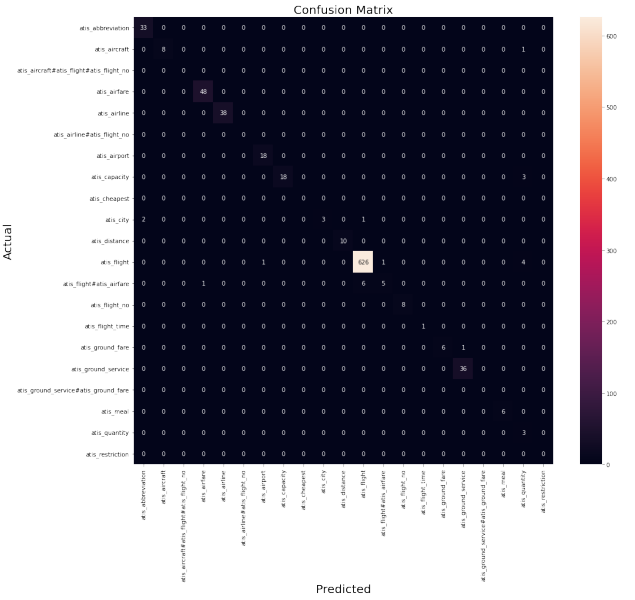

Рис. 3.7: Англійський датасет

На рис. 3.8, 3.9 можна бачити матриці плутанини для слотів для англійської і української версій датасетів відповідно. Ці матриці є дуже великими, тому їх нормалізовано, щоб можна було оцінити діагональне переважання.

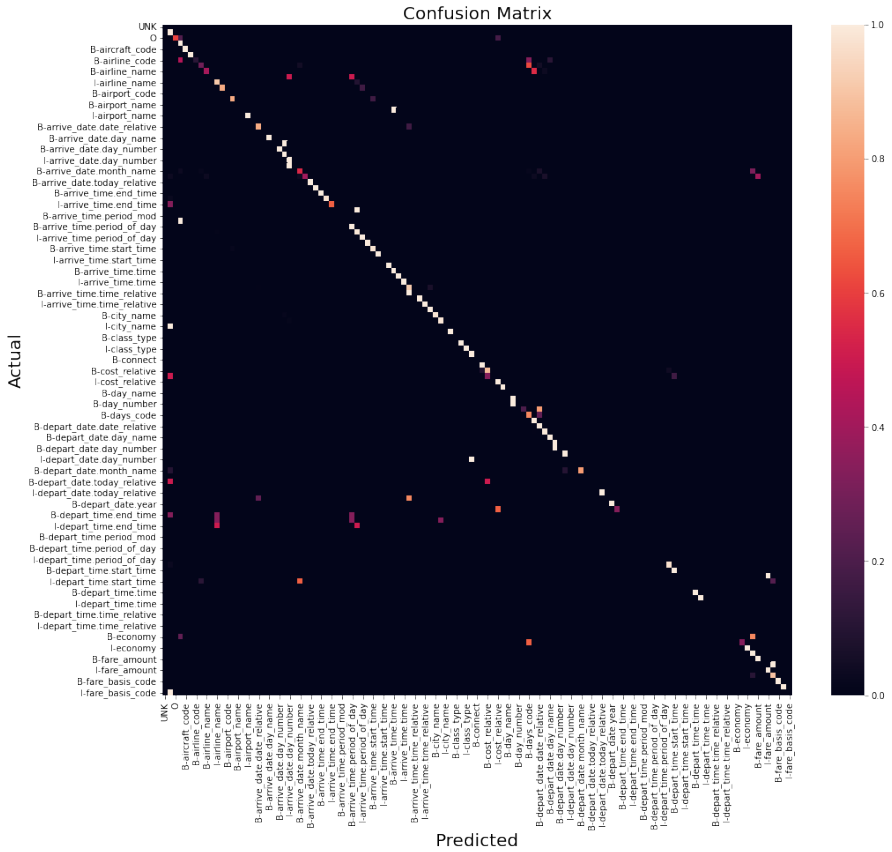

Рис. 3.8: Англійський датасет

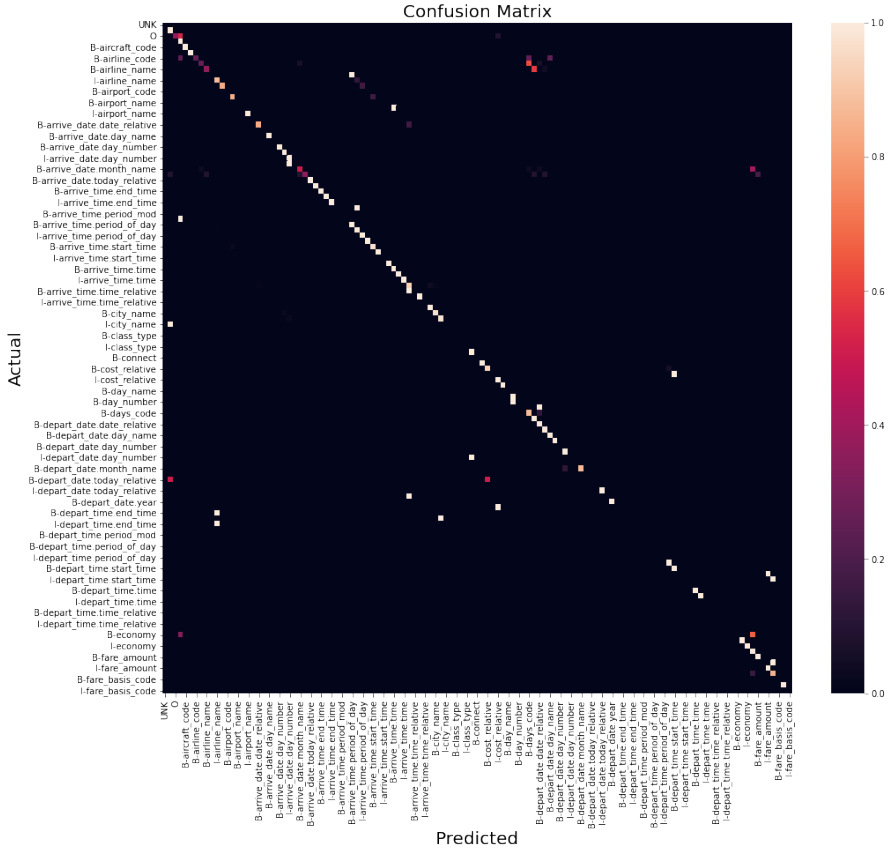

Рис. 3.9: Український датасет

Приклад повністю правильно класифікованого висловлювання:

sentence	будь ласка знайдіть рейс зі львова до берліну
slots	O O O O O B-fromloc.city_name O B-toloc.city_name
intent	atis_flight

Приклад помилково класифікованого висловлювання, можна бачити, що помилки не є суттєвими і практично все речення класифіковано правильно.

sentence	мені потрібен рейс з борисполя в одесу в один бік вилітаючи в середу ввечері або в четвер вранці
Predicted slots	O O O O B-fromloc.city_name O B-toloc.city_name O B-round_trip I-round_trip O O B-depart_date.day_name B-depart_time.period_of_day O O B-depart_date.day_name B-depart_time.period_of_day
Actual slots	O O O O B-fromloc.airport_name O B-toloc.city_name O B-round_trip I-round_trip O O B-depart_date.day_name B-depart_time.period_of_day O O B-depart_date.day_name B-depart_time.period_of_day
intent	atis_flight

Висновки

У даній роботі розглянуто задачу розуміння природної мови в контексті віртуального асистента. Зокрема, представлено роль NLU модуля в чат-боті, оглянуто актуальні підходи до вирішення задачі NLU, досліджено спільні підходи з використанням попередньо навченої моделі BERT, а також було описано метод умовних випадкових полів(CRF), який може використовуватись для покращення результатів. Результати роботи різних модифікацій алгоритмів порівняно на українській версії датасету ATIS, яка була реалізована спеціально для цієї роботи.

Програмна частина була реалізовна на мові програмування python з використанням бібліотек torch, transformers та інших. Код виконувався в блокноті Google Colaboratory.

Результати показують, що моделі на базі BERT добре справляються з задачею, як на українському, так і на англійському датасеті, а метод CRF справді покращує результати.

Майбутніми дослідженнями є реалізація компоненти генерування природної мови(NLG) і побудова цілої системи діалогу.

Література

[1] Arvindpdmn. *Bert (Language Model) [Electronic resource]*. Черв. 2021. URL: https://devopedia.org/bert-language-model.

[2] *Chatbot definition and meaning [Electronic resource]*. URL: https://dictionary.cambridge.org/dictionary/english/chatbot.

[3] Qian Chen, Zhu Zhuo та Wen Wang. *BERT for Joint Intent Classification and Slot Filling*. 2019. arXiv: 1902.10909 [cs.CL].

[4] Mai Hoang Dao, Thinh Hung Truong та Dat Quoc Nguyen. "Intent Detection and Slot Filling for Vietnamese". В: *Interspeech 2021* (серп. 2021). DOI: 10.21437/interspeech.2021-618. URL: http://dx.doi.org/10.21437/interspeech.2021-618.

[5] *Exploring conditional random fields for NLP applications [Electronic resource]*. Листоп. 2021. URL: https://hyperscience.com/tech-blog/exploring-crfs-for-nlp-applications/.

[6] P. Haffner, G. Tur та J.h. Wright. "Optimizing SVMs for complex call classification". В: *2003 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings. (ICASSP 03).* (2003).

[7] Bing Liu та Ian Lane. "Recurrent neural network structured output prediction for spoken language understanding". В.

[8] Andrew McCallum, Dayne Freitag та Fernando C. N. Pereira. "Maximum Entropy Markov Models for Information Extraction and Segmentation". В: *Proceedings of the Seventeenth International Conference on Machine Learning*. ICML '00. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2000, с. 591—598.

[9] *Papers with Code - ATIS Dataset [Electronic resource]*. URL: https://paperswithcode.com/dataset/atis.

[10] A. M. TURING. "I.—COMPUTING MACHINERY AND INTELLIGENCE". В: *Mind* LIX.236 (жовт. 1950), с. 433—460. ISSN: 0026-4423. DOI: 10.1093/mind/LIX.236.433. eprint: https://academic.oup.com/mind/

article - pdf / LIX / 236 / 433 / 30123314 / lix - 236 - 433 . pdf. URL: https://doi.org/10.1093/mind/LIX.236.433.

[11] Tsung-Hsien Wen та ін. "Conditional Generation and Snapshot Learning in Neural Dialogue Systems". В: *CoRR* (2016). URL: http://arxiv.org/abs/1606.03352.

[12] Tsung-Hsien Wen та ін. "Semantically Conditioned LSTM-based Natural Language Generation for Spoken Dialogue Systems". В: *CoRR* abs/1508.01745 (2015). arXiv: 1508.01745. URL: http://arxiv.org/abs/1508.01745.

[13] Puyang Xu та Ruhi Sarikaya. "Convolutional neural network based triangular CRF for joint intent detection and slot filling". В: *2013 IEEE Workshop on Automatic Speech Recognition and Understanding*. 2013, с. 78—83.

[14] Kaisheng Yao та ін. "Spoken language understanding using long short-term memory neural networks". В: *2014 IEEE Spoken Language Technology Workshop (SLT)*. 2014, с. 189—194.

[15] Xiang Zhang, Junbo Zhao та Yann LeCun. *Character-level Convolutional Networks for Text Classification*. 2015.

[16] Xiang Zhang, Junbo Zhao та Yann LeCun. "Character-level Convolutional Networks for Text Classification". В: *Advances in Neural Information Processing Systems*. За ред. С. Cortes та ін. Т. 28. Curran Associates, Inc., 2015. URL: https://proceedings.neurips.cc/paper/2015/file/250cf8b51c773f3f8dc8b4be867a9a02-Paper.pdf.

[17] Zhichang Zhang та ін. "A Joint Learning Framework With BERT for Spoken Language Understanding". В: *IEEE Access* 7 (2019), с. 168849—168858.

[18] Tiancheng Zhao та Maxine Eskenazi. *Towards End-to-End Learning for Dialog State Tracking and Management using Deep Reinforcement Learning*. 2016. URL: https://arxiv.org/abs/1606.02560.

[19] Jie Zhou та Wei Xu. “End-to-end learning of semantic role labeling using recurrent neural networks”. В: *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Beijing, China: Association for Computational Linguistics, лип. 2015, с. 1127—1137. URL: https://aclanthology.org/P15-1109.

[20] Сидоряк В. “Генерування діалогу віртуального асистента за допомогою навчання з підкріпленням [Курсова робота]”. В: (2021).