Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
(повна назва)

Кафедра Штучного інтелекту
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

Дослідження методів обробки природної мови для генерації діалогів

(тема)

Виконав:
студент 2 курсу, групи СШІм-20-3
Воробйов Є. К.
(прізвище, ініціали)

Спеціальність 122 Комп'ютерні науки
(код і повна назва спеціальності)

Тип програми освітньо-наукова
(освітньо-професійна або освітньо-наукова)

Освітня програма Системи штучного інтелекту
(повна назва спеціалізації)

Керівник зав. каф. ІУС Петров К.Е.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ В.О. Філатов
(підпис) (прізвище, ініціали)

2022 р.

Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
(повна назва)

Кафедра Штучного інтелекту
(повна назва)

Рівень вищої освіти другий (магістерський)

Спеціальність 122 Комп'ютерні науки
(код і повна назва)

Тип програми освітньо-наукова
(освітньо-професійна або освітньо-наукова)

Освітня програма Системи штучного інтелекту (СШІ)
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«_____» _____ 20 ___ р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Воробйову Євгену Костянтиновичу
(прізвище, ім'я, по батькові)

1. Тема роботи «Дослідження методів обробки природної мови для генерації діалогів»

затверджена наказом університету від 24 березня 20 22 р. № 414Ст
2. Термін подання студентом роботи до екзаменаційної комісії _____ _____ 20 ___ р.
3. Вихідні дані до роботи Науково-технічні публікації, дані статей, результати досліджень, специфікація мови Python, номери профільних видань

4. Перелік питань, що потрібно опрацювати в роботі Аналіз науково-технічної літератури з питань обробки природної мови, дослідження існуючих підходів до класифікації діалогових актів, проведення експериментального моделювання та навчання моделі класифікації діалогових актів

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри)_____

6. Консультанти розділів роботи (п.6 включається до завдання за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Отримання завдання на кваліфікаційну роботу	28.03.2022	виконано
2	Аналіз предметної галузі і постановка задачі	26.03.2022-01.04.2022	виконано
3	Аналіз існуючих методів класифікації діалогових актів	02.04.2022-09.04.2022	виконано
4	Розробка нового моделі класифікації діалогових актів	10.04.2022-15.04.2022	виконано
5	Програмна реалізація	16.04.2022-21.04.2022	виконано
6	Обробка і оформлення результатів	22.04.2022-24.04.2022	виконано
7	Оформлення пояснювальної записки	25.04.2022-28.04.2022	виконано
8	Оформлення графічних матеріалів	29.04.2022-30.04.2022	виконано
9	Захист перед ЕК		

Дата видачі завдання 28 березня 20 22 р.

Студент _____
(підпис)

Керівник роботи _____ _____
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Записка пояснювальна: 66 с., 23 рис., 1 табл., 1 дод., 33 джерел.

ГЛИБОКЕ НАВЧАННЯ, ДІАЛОГ, КЛАСИФІКАЦІЯ, НЕЙРОННІ МЕРЕЖІ, ОБРОБКА ПРИРОДНОЇ МОВИ.

Об’єкт дослідження – процес автоматичної класифікації діалогових актів в задачі обробки природної мови та генерації діалогів.

Предмет дослідження – методи класифікації діалогових актів на основі нейронних мереж.

Метою роботи є дослідження та аналіз методів та алгоритмів автоматичної класифікації діалогових актів та розробка власної моделі на основі попередніх досліджень.

Дослідження, що проведені в роботі базуються на використанні теорії ймовірностей та математичної статистики, теорії графів, теорії та методів оптимізації, апарату штучних нейронних мереж для розв'язання задачі розпізнавання діалогових актів; методів математичного та комп’ютерного моделювання для верифікації точності запропонованої моделі.

У роботі проведено аналіз стану класифікації діалогових актів та порівняння наявних алгоритмів та методів. На основі отриманих результатів сформовані вимоги до моделі та запропонована власна модель класифікації актів. Проведено навчання та тестування моделі з подальшим аналізом результатів.

ABSTRACT

Explanatory note: 66 p., 23 fig., 1 tab., 1 app., 33 sources.

CLASSIFICATION, DEEP LEARNING, DIALOGUE, NATURAL LANGUAGE PROCESSING, NEURAL NETWORKS.

Object of study – process of automatic classification of dialogue acts in the task of natural language processing and dialogue generation.

The subject of the study is the dialogue acts classification methods based on neural networks.

The aim of the work is to study and analyze methods and algorithms for automatic classification of dialogue acts and develop our own model based on previous studies.

The research conducted in the work is based on the use of probability theory and mathematical statistics, graph theory, theory and optimization methods, the apparatus of artificial neural networks to solve the problem of recognizing dialogue acts; methods of mathematical and computer modeling to verify the accuracy of the proposed model.

The analysis of the state of classification of dialogue acts and comparison of existing algorithms and methods is carried out in this attestation work. Based on the results obtained, the requirements for the model are formed and our own model of dialogue act classification is proposed. The model was trained and tested with subsequent analysis of the results.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

DA – Dialog Act – діалоговий акт;

DAMSL – Dialog Act Markup in Several Layers –розмітка діалогових актів у кілька шарів;

NLP – Natural Language Processing – обробка природної мови.

ВСТУП

Проблема взаємодії людини з комп'ютером існує з моменту появи обчислювальної техніки. З самого початку з електронною обчислювальною машиною взаємодіяти мали змогу лише програмісти. У міру розширення сфери використання комп'ютера та збільшення масштабів їх застосування кінцеві користувачі стали втягуватись в процес безпосередньої взаємодії з комп'ютером, що призвело до появи масової категорії користувачів – прямих кінцевих користувачів, що працюють в діалоговому режимі. Таким чином, сформувалась проблема обробки природної мови та її транслювання у машинне подання.

Обробка природної мови, як загальний напрям інформатики та штучного інтелекту, бере свій початок ще наприкінці 1940-х років. Саме тоді вперше постала задача машинного перекладу тексту. З плином часу задачі цієї галузі змінювались та розширювались. На сьогоднішній день обробка природної мови одна з найбільш затребуваних галузей штучного інтелекту. Напрямів дослідження в середині цієї галузі досить багато, зокрема машинний переклад, інформаційний пошук, реферування та анотування текстів, класифікація та кластеризація текстів, в тому числі тематичне моделювання, створення чат-ботів, аналіз тональності текстів, видобування знань з текстів, автоматична генерація текстів тощо.

Ще одним популярним напрямом є дослідження генерації діалогів. У сучасному світі все більше досліджень та розробок спрямовано на автоматизацію дій або процесів, зокрема автоматизацію діалогів. Прикладам такої автоматизації можуть слугувати персональні мобільні асистенти, питально-відповідальні системи, чат-боти, тощо. Подібні системи дозволяють користувачу вирішувати певні задачі через спілкування з комп’ютером природною мовою у форматі діалогу. Саме на дослідженнях та розробці подібних систем зосереджений напрям генерації діалогів.

Однією з ключових задач цього напряму є класифікація діалогових

актів, де діалоговий акт (DA) – це висловлювання в контексті розмовного діалогу, яке виконує певну функцію в діалозі. Іншими словами, акт певної репліки – це мета, яку хочете досягти автор, висловлюючи її. Прикладом актів може бути запит інформації, твердження, згода та інші. Розпізнавання та класифікація DA підвищує якість та доцільність генерованих відповідей, що в свою чергу підвищує якість усього діалогу.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Опис предметної області

Обробка природної мови – один із найважливіших напрямів досліджень у галузі штучного інтелекту. Зародження цього напряму пов'язують з появою перших обчислювальних машин і з ідеєю необхідності використовувати машини для вирішення корисних завдань, пов'язаних з природною мовою, якою розмовляють і пишуть люди. Обробка природної мови спирається на багато дисциплін: від класичних мовних наук, як-от лінгвістика, морфологія та синтаксис, до суттєво більш технічних – інформатики та комп'ютерної лінгвістики.

Ще шість років тому NLP переважно вбирала у собі техніки та методи з інших галузей, але згодом вона почала експортувати їх. Методи, які розвинулись у сфері аналізу природних мов, почали успішно застосовуватись й в інших областях. Зараз при аналізі природних мов теж починають широко застосовувати глибокі нейромережі.

Однією з центральних проблем для IT-рішень штучного інтелекту є завдання «розуміння» тексту, тобто отримання сенсу з тексту природною мовою. Саме до неї, зрештою, зводяться практичні рішення розумних мовних технологій. Проте вона дуже далека від рішення у загальному вигляді, адже феномен чи специфікацію «розуміння» поки що не можуть пояснити чи змоделювати ні психологи, ні нейрофізіологи.

Особливо важливо підкреслити, що з точки зору аналізу інформації, природна мова є чи не найбільш неструктурованим та повним протиріччя видом вхідних даних. Людська мова досить складна і різноманітна для примітивної формалізації. Ми висловлюємо свої думки як усно, так і письмово. Окрім факту існування сотень мов та діалектів, варто зазначити факт унікальності граматичних та синтаксичних правил, термінів та сленгу

у просторі, навіть однієї мови. З огляду на це, перед аналізом природної мови постають наступні типові проблеми, а саме:

− синтаксична невизначеність: «Час – не кінь, не підженеш і не зупиниш» для програми може бути абсолютно не зрозуміло, про що саме йдеться у реченні, про коня чи про час;

− смислова невизначеність полягає у багатозначності елементів тексту. Розглянемо питальне речення «Де знайти ключ до того замку?» слово «замок» може мати два абсолютно різні значення, зважаючи на поставлений наголос або контекст;

− відмінкова невизначеність: у фразах «Усі були схвильовані перед концертом» та «Перед автівки був вщент розбитий» слово «перед» означає час або місце, що абсолютно змінює сенс фрази;

− референційна невизначеність: у фразі «Відкрий поличку та дістань мокру парасольку, я хочу її висушити» займенник «її» за смисловим значенням матиме відношення до мокрої парасольки, проте для машини, у якої повністю відсутнє розуміння реальності, даний займенник відноситиметься як до полички, так і до парасольки.

1.2 Огляд структури задач обробки природної мови

В цілому обробка природної мови є величезним спектром задач різного рівня.

Задачі рівня сигналу полягають у наявності певного типу вхідних даних, вигляд яких не дозволяє зробити його обробку. Такими даними можуть бути безпосередньо мова, рукопис або друкований відсканований текст. Суть задачі полягає в отриманні цифрового представлення вхідних даних, з яким може працювати програма. Отже, прикладами таких задач є машинне розпізнавання тексту, мови та їх синтез.

Рівень слова розглядає його багатомірність як лексичної одиниці, оскільки воно може бути описано багатьма характеристиками з боку

лінгвістичних наук, що впливає на його вживаність у тій чи іншій ситуації. Типовими задачами тут є морфологічний аналіз, корекція помилок, нормалізація словоформ за допомогою стемінгу чи лематизації.

Наступний рівень – рівень словосполучень. Тут ми вже оперуємо поняттями частин мови і їх співвідношеннями. Так як у деяких випадках визначення частини мови є неоднозначним без використання контексту вживання, то цю задачу неможливо помістити на рівень слова. Отже, на рівні словосполучень ми маємо задачі відокремлення слів, визначення частини мови та визначення іменованих сутностей.

Словосполучення формують речення, а тому і наступний рівень теж пов’язаний із простором речення. На ньому речення потребують виділення, синтаксичного аналізу членів речення, усунення неоднозначності слів, усунення референційної невизначеності.

Виходячи за рамки одного речення, ми потрапляємо на рівень абзаців. Сутність, описана в одному реченні, цілком закономірно може мати посилання у інших. Тому виникає питання розв’язання посилань і встановлення відносин між об'єктами, згаданими у різних реченнях. З абзацами, окрім розв’язання посилань і встановлення відношень, ми можемо вирішувати нові завдання: проаналізувати емоційну тональність тексту, визначити якою мовою він написаний.

Абзаци формують рівень документів. Документ є вже повністю самостійною одиницею інформації, з точки зору її розгляду під час аналізу. Тут варто згадати задачу семантичного аналізу тексту, тобто визначення змісту та автоматичного анотування. Окремим випадком задачі рівня документу є машинний переклад текстів.

Найбільшим та останнім рівнем задач обробки природної мови є рівень корпусу. Він містить задачі дедублікації та розгорнутого пошуку на великому корпусі документів [1].

При розв'язанні задач зазначених вище рівнів потрібно враховувати наступне:

1. кожний з рівнів задач обов’язково потребує використання деяких задач нижчих рівнів. Так, наприклад, без відокремлення слів, що є задачею рівня словосполучення, неможливо уявити розв’язання задачі синтаксичного аналізу членів речення, що є задачею рівня речення;
2. на всіх рівнях завдання фактично йдуть у два шляхи: пов'язані з розбором існуючої мови і з генерацією нового матеріалу.

1.3 Діалогові системи та генерація діалогів

Наявність віртуального помічника або системи супутника чату з достатнім інтелектом здавалось далеким від реальності і могло існувати лише в науково-фантастичних фільмах протягом тривалого часу. Останнім часом спілкування людини з комп’ютером привертає все більшу увагу через його перспективний потенціал і привабливу комерційну цінність. З розвитком «великих даних» і методів глибокого навчання мета створити автоматичну комп’ютерну систему мовлення, як нашого особистого помічника чи супутника чату, більше не є ілюзією. З одного боку, сьогодні ми можемо легко отримати доступ до «великих даних» на будь-яку тему, що дозволяє нам створювати корпуси для навчання та розробки систем розмови між людьми та комп’ютерами не прив’язуючись до домену. З іншого боку, методи глибокого навчання виявились ефективними у фіксації складних шаблонів у «великих даних», і вони створили багато сфер дослідження, такі як: комп’ютерний зір, обробка природної мови та системи рекомендацій. Таким чином, з’явилась велика кількість досліджень, які використовують величезну кількість даних та глибоке навчання для просування систем діалогу та вирішення задачі генерації діалогів.

За галузями застосування, діалогові системи можуть бути приблизно поділені на дві групи: системи, орієнтовані на завдання та не орієнтовані на завдання системи (також відомі як чат-боти).

Системи, що орієнтовані на завдання, мають на меті допомогти

користувачеві виконати певні задачі (наприклад, знайти продукти, забронювати помешкання або столик у ресторані). Широко застосовувані підходи до систем, орієнтованих на завдання, полягають у тому, щоб розглядати відповідь в діалозі як конвеєр. Системи спочатку намагаються зрозуміти повідомлення, передане людиною, представляють його як внутрішній стан, а потім виконують деякі дії відповідно до правил, враховуючи поточний стан діалогу, і, нарешті, дія перетворюється на свою поверхневу форму – природну мову. Хоча розуміння природної мови спирається на статистичні моделі, більшість розгорнутих систем діалогу все ще використовують ручні функції або створені вручну правила для представлення стану та простору дій, виявлення намірів та заповнення слотів. Це не тільки робить розгортання справжньої діалогової системи дорогим і трудомістким, але й обмежує її використання у інших доменах. Нещодавно було розроблено багато алгоритмів, заснованих на глибокому навчанні, щоб вирішити ці проблеми та досягти помітних покращень у цих аспектах. Крім того, є спроби побудувати наскрізні орієнтовані на завдання діалогові системи, які можуть розширити представлення простору станів у традиційних конвеєрних системах і допомогти узагальнити діалоги за межами анотованих корпусів, що стосуються конкретних завдань.

Системи, що не орієнтовані на завдання, взаємодіють з людиною, щоб забезпечити розумні відповіді та розваги. Зазвичай вони зосереджені на спілкуванні з людиною у відкритих доменах. Незважаючи на те, що системи, не орієнтовані на виконання завдань, на перший погляд, виконують розважальні функції, вони домінують у багатьох реальних застосунках. Загалом, для систем, не орієнтованих на завдання, розроблено два основних підходи – генеративні методи, такі як Seq2Seq [2], які генерують належні відповіді під час розмови; і методи на основі пошуку, які навчаються вибирати відповіді для поточної розмови з певної бази знань.

Обидва класи систем мають вміти генерувати коректні та змістовні відповіді на репліки співрозмовника. Першим та одним з головних кроків

цього процесу є розуміння та опрацювання попередньої репліки або декількох реплік. Під розумінням репліки мається на увазі її лексичний та синтаксичний розбір, знаходження її сенсу та віднесення до попередніх висловлювань у діалозі. Для покращення розуміння репліки також може буди застосована її класифікація – тобто віднесення репліки до певного класу в залежності від її мети та сенсу [3]. Групування висловлювань у класи дозволяє застосовувати певні шаблони генерації відповідей в залежності від цих класів.

1.4 Класифікація діалогових актів

Класифікація – проблема упорядкування за деяким принципом множини об'єктів, що мають подібні класифікаційні властивості (один або декілька властивостей), обраних для визначення подібності або різниці між цими об'єктами [4].

Сучасні методи класифікації можна поділити на дві окремі групи:

– допоміжна (штучна) класифікація, яка виконується на зовнішньому атрибуті та служить для надання множині об'єктів (процесів, явищ) бажаного порядку;

– природна класифікація, яка утворюється суттєвими ознаками, що характеризують внутрішню підмножину об'єктів та явищ. Це і є результатом і важливим засобом наукових досліджень, оскільки воно передбачає і закріплює результати вивчення моделей класифікованих об'єктів.

Під класифікацією розуміється віднесення об'єктів (спостережень, подій) до одного з попередньо відомих класів.

Класифікація відноситься до стратегії навчання зі вчителем, яке також називається контрольованим або керованим навчанням.

Завдання класифікації часто називають прогнозом категориальної залежності змінної (тобто залежної змінної, яка є категорією) заснованої на основі чисельних та/або категоріальних змінних.

Класифікація може бути одновимірною (використовується одна ознака) і багатовимірною (використовується дві або більше ознак).

Класифікація загальної мети висловлювання користувача в розмові, також відомого як діалоговий акт, наприклад, відкрите запитання, висловлювання думки або запит думки, є ключовим кроком у розумінні природної мови для розмовних агентів. Хоча класифікація DA була детально вивчена в розмовах між людьми, вона недостатньо вивчена для нових автоматизованих розмовних агентів з довільними темами. Більше того, незважаючи на значний прогрес у класифікації DA на рівні висловлювань, повне розуміння висловлювань діалогу вимагає контексту розмови.

При вирішені задачі класифікації діалогових актів необхідно розглянути два попередніх питання, перш ніж можна буде застосувати техніку класифікації: вибір набору тегів і визначення ознак, які використовуються класифікаторами.

1.5 Набори тегів для класифікації DA

Визначення набору тегів DA є важливим, але складним кроком, оскільки воно є результатом компромісу між трьома суперечливими вимогами:

1. теги DA повинні бути достатньо конкретними, щоб кодувати детальні характеристики цільового завдання;
2. теги DA повинні бути достатньо загальними, щоб бути корисними для різних завдань, або, принаймні, стійкими до мінливості та розвитку цільового застосування;
3. теги DA повинні бути чіткими та легко відокремлюваними, щоб максимізувати узгодження між людьми, що анотують репліки.

Дослідження діалогових актів призвело до кількох схем анотації. Серед них – розмітка в кількох шарах (DAMSL) [5], яка слугує основою для

анотації двох довідкових корпусів, SwDA та MRDA [6-7]. DAMSL розроблявся як універсальний набір. Теги в DAMSL мають декілька рівнів класифікації. На першому рівні всі теги поділяються на чотири основні класи: комунікативний статус (Communicative Status) – фіксує, чи є висловлювання зрозумілим і чи було його успішно завершено, інформаційний рівень (Information Level) – характеристика смислового змісту висловлювання, функції, орієнтовані на перспективу (Forward Looking Function) – як поточне висловлювання обмежує майбутні переконання та дії учасників і впливає на дискурс та функції зворотного огляду (Backward Looking Function) – як поточне висловлювання пов'язане з попереднім дискурсом. Загалом, ці класи вважаються ортогональними, і можна побудувати приклади для будь-якої можливої їх комбінації. Комунікативний статус визначає, чи є висловлювання неінтерпретованим, залишеним чи є саморозмовою. Ця функція не використовується для більшості висловлювань. Інформаційний рівень забезпечує абстрактну характеристику змісту висловлювання. Він складається з чотирьох категорій: завдання, управління завданнями, управління зв'язком та інший рівень. Функції, що орієнтовані на перспективу, організовані в таксономію, подібно до дій у традиційній теорії мовленнєвих актів. Функції зворотного огляду показують зв'язок між поточним висловлюванням і попередніми діями діалогу, такими як прийняття пропозиції чи відповідь на запитання.

Далі Forward Looking Function та Backward Looking Function теги поділяються на підкласи зазначені на рисунках 1.1 та 1.2 в залежності від змісту.

Forward Looking Function

- Statement
  - Assert
  - Reassert
  - Other-statement
- Influencing-addressee-future-action
  - Open-option
  - Action-directive
- Info-request
- Committing-speaker-future-action
  - Offer
  - Commit
- Conventional Opening Closing
- Explicit-performative
- Exclamation
- Other-forward-function

Рисунок 1.1 – Підкласи класу Forward Looking Function

Backward Looking Function

- Agreement
  - Accept
  - Accept-part
  - Maybe
  - Reject-part
  - Reject
  - Hold
- Understanding
  - Signal-non-understanding
  - Signal-understanding
    - Acknowledge
    - Repeat-rephrase
    - Completion
  - Correct-misspeaking
- Answer
- Information-relation

Рисунок 1.2 – Підкласи класу Forward Looking Function

Зовсім нещодавно схема DIT++, яка призвела до створення стандарту ISO 24617-2 [8], запропонувала організацію, засновану на різних вимірах

(наприклад, управління діалоговими ходами, управління соціальними зобов'язаннями та ін.), кожен із яких містить різні діалогові акти. Загалом DIT++ пропонує виділити більше 100 діалогових актів. DAMSL, з іншого боку, пропонує 226 діалогових актів, які зазвичай групуються в 42 мітки. З точки зору автоматичної класифікації, забагато класів не призводить до ефективних результатів. У значній кількості досліджень пропонують обмежити кількість класів, використовуючи або загальні метакласи, або найбільш часто використовувані. Наприклад, кілька робіт засновані на тегах DAMSL скорочено до 5 класів (твердження, запитання, зворотна відповідь, наповнювачі та зриви) [9]. У деяких інших роботах, як-от довідкова [10] або [11], використовується набір тегів, скорочений до 5 найпоширеніших тегів (твердження, зворотна відповідь, думка, залишення, згода).

Зменшення набору тегів також можна зробити для пристосування класифікації до потреб конкретної доменної діалогової системи. Це, наприклад, було зроблено у роботі [12], описуючи комунікацію в контексті реагування на катастрофи за допомогою роботів. У цій роботі запропоновано конкретний набір метакласів ISO, адаптованих до потреб системи, об'єднавши 20 найкорисніших діалогових актів у 8 метакласів (зв'язок, повідомлення, твердження, запит, запитання, підтвердження, скасування, заперечення).

1.6 Ознаки класифікації DA

Друге питання, на яке потрібно відповісти для класифікації DA – про властивості, які використовуються класифікаторами. Такі інформаційні властивості можна поділити на декілька типів.

Перший – це лексична інформація. Кожне висловлювання складається з послідовності слів. Як правило, DA висловлювання можна частково вивести зі списків слів, які утворюють це висловлювання. Наприклад, запитання часто містять питальне слово, яке рідко зустрічається в інших

класах DA. Лексичну інформацію зазвичай фіксують уніграми слів.

Другий тип – це синтаксична інформація. Вона пов’язана з порядком слів у висловлюванні. Наприклад, у французькій та чеській мовах відносний порядок появи підмета та дієслова може використовуватися для розрізнення декларацій та питань. N-грами слів часто використовуються при розпізнаванні діалогових актів для моделювання деякої локальної синтаксичної інформації. Іншим типом синтаксичної інформації, що активно використовується для розпізнавання DA, є «рекомендаційні фрази», які фактично відповідають підмножині конкретних n-грам, де n може варіюватися від 1 до 4. Вони вибираються на основі їхньої здатності передбачити певний діалоговий акт та частоти їх появи. Ці ключові фрази насправді відповідають загальним і типовим послідовностям слів. Оскільки вони не моделюють весь лексичний простір, їх можна інтерпретувати в контексті виявлення DA замість розпізнавання DA.

Інша тип інформації – це семантична інформація. Діалоговий акт також залежить від значень висловлювання та слів, які його складають. Проте існує багато різних визначень «семантичної інформації», починаючи від широких тематичних категорій, таких як «погода», «спорт», до точних інтерпретацій на основі фреймів, наприклад: «показати рейси з Лондона до Парижа 12 березня». Останній зазвичай використовується в програмах для розуміння розмовної мови, де діалоговий акт залежить від конкретної попередньо визначеної дії. Інший вид семантичної інформації, який використовується для розпізнавання DA, – це конкретні об’єкти, такі як іменовані об’єкти або об’єкти завдання. Наприклад, дата, місце або власні іменники, коли вони вимовляються, можуть бути важливими ознаками для з’ясування та розпізнования діалогового акту.

Ще одна корисна інформація для розпізнавання DA – це просодія, і, зокрема, мелодіка висловлювання. Зазвичай запитання мають зростаючу мелодіку в кінці висловлювання, тоді як висловлювання часто характеризуються дещо спадною мелодікою.

Остання ознака, згадана тут, є контекстом кожного DA. Отже, будь-який DA залежить від попередніх (і наступних) DA. Найважливішим контекстом є попередні вислови. Наприклад, відповідь «так» або «ні», швидше за все, буде слідувати за запитанням «так/ні». Послідовність DA також називається історією діалогу.

1.7 Постановка задачі

Метою даної кваліфікаційної роботи є дослідження наявних на сьогоднішній день методів класифікації діалогових актів та розробка власної моделі класифікації DA на їх основі.

Для досягнення поставленої мети необхідно розглянути наступні задачі:

– провести аналіз предметної області;

– дослідити існуючі методи та порівняти їх;

– розробити та описати власний підхід до розв’язання задачі класифікації DA;

– навчити та протестувати розроблену модель класифікації DA;

– проаналізувати отримані результати.

2 МЕТОДИ КЛАСИФІКАЦІЇ ДІАЛОГОВИХ АКТІВ

2.1 Постановка задачі класифікації діалогових актів

При вирішенні завдання класифікації DA розмову C можна розглядати як вхідну інформацію, яка являє собою послідовність висловлювань різної довжини $U = \{u_1, u_2, \ldots, u_L\}$. Кожне висловлювання $u_i \in U$, у свою чергу, є послідовністю слів різної довжини $\{w_i^1, w_i^2, \ldots, w_i^{N_i}\}$ і має відповідну цільову мітку $y_i \in Y$. Отже, кожна розмова (тобто послідовність висловлювань) відображається на відповідну послідовність цільових міток $Y = \{y_1, y_2, \ldots, y_L\}$, яка представляє діалогові акти, пов'язані з відповідними висловлюваннями.

2.2 Баєсівські підходи

Основні типи підходів автоматичного розпізнавання діалогових актів, що розглядаються в літературі, можна широко класифікувати на баєсівські та небаєсівські підходи. Баєсівські підходи ґрунтуються на баєсовом висновуванні. Вони відносяться до найбільш розповсюджених та досліджених підходів.

2.2.1 Лексичні n-грам моделі

Байєсівський формалізм довгий час вважався кращим підходом у сфері розпізнавання DA. У приведеному досліджені краща послідовність діалогових актів $\hat{C}$ знаходиться за допомогою максимізації апостеріорної ймовірності $P(C|O)$ серед усіх можливих послідовностей діалогових актів C за формулою 2.1 [13].

$$\hat{C} = \arg\ max_c P(C|O)\ =\ \arg\ max_c \frac{P(C)\cdot P(O|C)}{P(O)}\ = \\ = \arg\ max_c P(C)\cdot P(O|C) \tag{2.1}$$

де $\hat{C}$ – краща послідовність DA;

C – довільна послідовність актів;

O – множина усіх послідовностей.

Більшість популярних підходів моделюють $P(O|C) = P(W|C)$, де W – це послідовність слів у вимовленому висловлюванні зі статистичними моделями, такими як n-грами. Ці методи засновані на спостереженні, що різні класи DA складаються з окремих наборів слів. Наприклад, 92,4% «угу» зустрічаються у зворотних відповідях, а 88,4% триграм «<початок> чи ви» зустрічаються у запитаннях «так/ні» [11]. Можна також враховувати порядок і позиції слів у висловлюванні.

Першим з подібних підходів є розпізнавання DA з точних транскрипцій слів. Даний підхід ґрунтується на гіпотезі, що слова у висловлюваннях відомі. Тоді рівняння 2.1 перетворюється на:

$$\arg\ max_c P(C|W)\ =\ \arg\ max_c P(C)\cdot P(W|C) \tag{2.2}$$

Застосовуючи наївне припущення щодо незалежності змінних можна отримати рівняння 2.3:

$$\arg\ max_c P(C)\cdot P(W|C)\ =\ \arg\ max_c P(C)\cdot \prod_{i=1}^{T} P(w_i|W) \tag{2.3}$$

де w_i – i-те слово у висловлюванні;

T – загальна кількість слів у висловлюванні.

Це рівняння представляє модель уніграми, яку також іноді називають наївним баєсовим класифікатором. У цьому випадку використовується тільки лексична інформація. Моделі вищого порядку, такі як 2-грами, 3-

грами тощо, також враховують деяку локальну синтаксичну інформацію про залежності між сусідніми словами. Через обмежені розміри корпусу використання 4-грам і більш складних моделей є рідкістю.

Використання цього підходу дозволяє отримати точність класифікації у проміжку від 46% до 80% в залежності від корпусу, мови та кількості цільових класів.

Далі можна припустити, що всі класи DA є рівно ймовірними, і, таким чином, не враховувати $P(C)$:

$$\hat{C} = \arg\, max_c P(W|C) \tag{2.4}$$

Цей підхід називають уніфікованим наївним класифікатором Баєса [14].

У багатьох реальних випадках точна транскрипція слів невідома. Її можна приблизно отримати за допомогою автоматичного розпізнавання мови. У такому випадку модель відносять до розпізнування DA з автоматичної транскрипції слів. Нехай A – випадкова величина, яка представляє акустичну інформацію мовного потоку (наприклад, спектральні характеристики). Послідовність слів W тепер є прихованою змінною, і ймовірність спостереження $P(A|C)$ може бути обчислена як:

$$P(A|C) = \sum_W P(A|W,C) \cdot P(W|C) = \sum_W P(A|W) \cdot P(W|C) \tag{2.5}$$

де C – клас діалогового акту;

$P(A|W)$ – імовірність спостереження, обчислена під час розпізнавання мови для заданої послідовності слів W.

Більшість робіт з розпізнавання діалогів за допомогою баєсових моделей використовують цей підхід і апроксимують підсумовування лише за k-кращою послідовністю слів.

2.2.2 N-грам моделі на основі послідовності реплік

Історія діалогу також містить дуже важливу інформацію для прогнозування поточного DA на основі попередніх. Історія діалогу зазвичай моделюється за допомогою статистичної граматики дискурсу, яка представляє попередню ймовірність $P(C)$ послідовності атків C.

Нехай C_τ – випадкова величина, яка представляє поточний клас акту діалогу в момент часу τ. Історія діалогів H визначається як попередня послідовність DA: $H = (C_{1,} \dots C_{\tau-1})$. Зазвичай вона зводиться до останніх n актів: $H = (C_{\tau-n+1,} \dots C_{\tau-1})$. Найпоширенішими значеннями для n є 2 і 3, що призводить до моделей 2 і 3-грам. Для навчання таких моделей умовні ймовірності $P(C_\tau | C_{\tau-n+1,} \dots C_{\tau-1})$ обчислюються на проанатованому навчальному корпусі. Методи згладжування також можуть використовуватися для навчання n-грам високого порядку.

Поліграми – це суміші n-грам різного порядку: n можна вибрати довільно великим, і ймовірності n-грам вищого порядку інтерполюються ймовірностями нижчого порядку. Вони зазвичай дають кращу точність розпізнавання, ніж стандартні n-грами, і показані в [15].

2.2.3 Приховані марковські моделі

Приховані моделі Маркова також можна використовувати для моделювання послідовностей діалогових актів. Нехай O – випадкова величина, яка представляє спостереження, а C – послідовність класів DA. Тоді можна розглядати ПММ (прихована марковська модель) n-го порядку, що означає, що кожен діалоговий акт залежить від n попередніх DA (так само, як і для n-грам). Потім кожен стан ПММ моделює один DA, а спостереження відповідають ознакам рівня висловлювання. Імовірності переходу навчаються на проанатованому наборі даних.

Розпізнавання DA здійснюється за допомогою деякого алгоритму

динамічного програмування, такого як алгоритм Вітербі. Для моделювання історії діалогів успішно використовуються ПММ зі словами та просодичними ознаками. У наведеному досліджені [16] використовуться інтонаційні події та функції нахилу, такі як: F0 (падіння/підйом), енергія, тривалість, тощо. Запропанована модель досягає 64% точності на корпусі DCIEM [17] з 12 класами DA. Також існують приклади поєднання ПММ з нейронними мережами [18]. Було отримано близько 76% точності на іспанському корпусі CallHome.

2.2.4 Баєсові мережі

Байєсова мережа представлена орієнтованим ациклічним графом. Вузли та дуги представляють відповідно випадкові величини та відносини (залежності) між вузлами. У контексті завдання не відрізняються динамічні баєсові мережі (зі стохастичними змінними) від статичних баєсових мереж, оскільки більшість змінних є стохастичними, і коли статичні байєсові мережі малюються, вони представляють уривок динамічної баєсової мережі в певний момент часу. Стохастичні змінні умовно залежать від їхніх нащадків і не залежать від їхніх предків. Приклад баєсові мережі для розпізнавання діалогових актів показаний на рисунку 2.1.

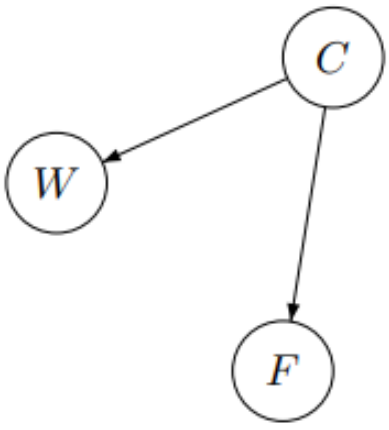

Рисунок 2.1 – Приклад баєсової мережі

Топологія графа моделює умовні незалежності між випадковими величинами. Вузол C представляє поточний акт діалогу. Ознаки висловлювання представлені вузлами W (послідовність слів у висловлюванні) і F (просодичні ознаки). Контекст діалогу не враховується. Твердження про умовну незалежність цієї мережі допускають наступну факторізацію:

$$P(C, W, F) = P(W|C) \cdot P(F|C) \cdot P(C) \tag{2.6}$$

Для побудови такої мережі необхідно визначити структуру мережі (умовні залежності) та умовні розподіли ймовірностей. Умовні ймовірності встановлюються шляхом навчання на навчальному наборі даних. Топологію мережі можна створити вручну або автоматично.

Інше застосування баєсових мереж для вирішення задачі розпізнавання діалогових актів показано в дослідженні [19]. Використовуються два типи ознак: ознаки висловлювання (слова у висловлюванні: w_i) та ознаки контексту (попередній діалоговий акт: $C_{\tau-1}$). Автори порівнюють дві різні баєсові мережі, що зображені на рисунку 2.2, щоб розпізнати DA.

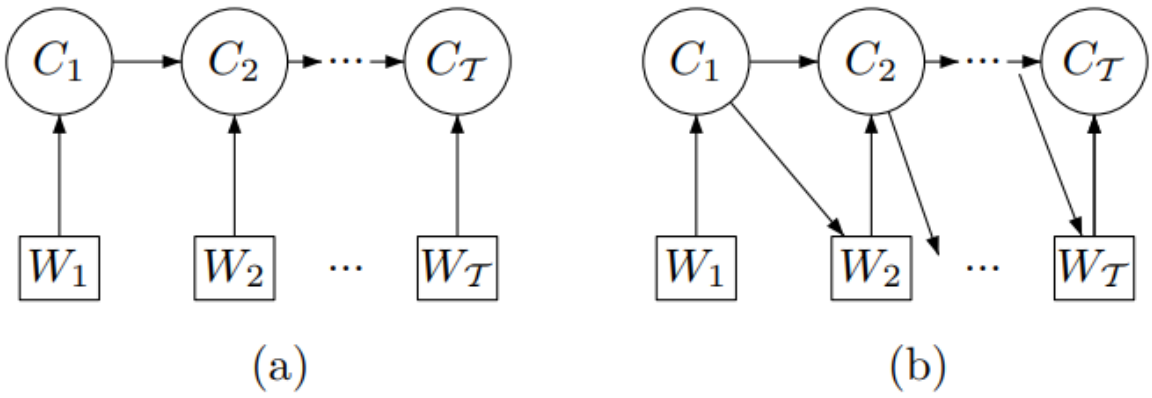

Рисунок 2.2 – Баєсові мережи з врахуванням контексту

Ці мережі побудовані вручну. У лівій моделі на рисунку 2.2 кожен акт діалогу розпізнається зі слів поточного висловлювання та з попереднього DA. У правій моделі автори далі розглядають додаткову залежність між кожним словом висловлювання та його попереднім діалоговим актом (діагональні дуги). Вони досягають приблизно 64% точності на підмножині корпусу MRDA і зі зменшеним набором класів.

Іншу баєсову модель – умовне випадкове поле з трикутним ланцюгом, яке спільно моделює дії діалогу та іменовані сутності, запропоновано в [20]. Ця модель показана на рисунку 2.3.

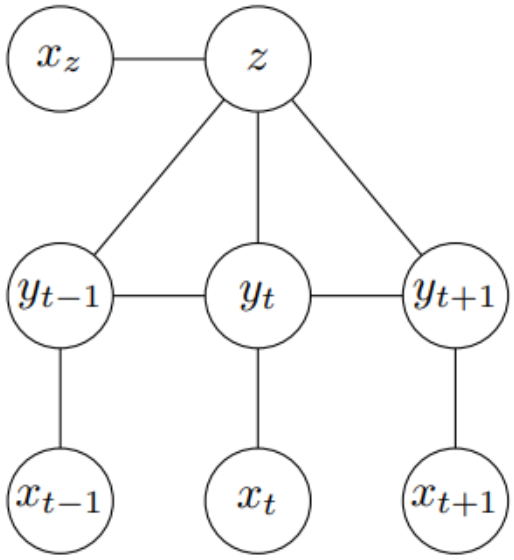

Рисунок 2.3 – Умовне випадкове поле

На малюнку z – це акти, y – іменовані сутності, а x – послідовність слів. У досліджені наведено, що ця спільна модель перевершує послідовні та каскадні моделі, в яких дії діалогу передбачаються незалежними від названих сутностей. У незалежному підході DA часто моделюються за допомогою багатовимірної логістичної регресії (або класифікатора максимальної ентропії). Як альтернатива, спільна модель поєднує як максимальну ентропію, так і підходи з умовними випадковими полями.

2.3 Небаєсівські підходи

Небаєсовські підходи також успішно використовуються в області розпізнавання DA, але вони не настільки популярні, як баєсовські. Прикладами таких підходів є нейронні мережі, такі як багатошаровий персептрон або мережі Кохонена, дерева рішень, навчання на основі пам’яті та навчання на основі трансформації.

2.3.1 Моделі на основі дерев рішень

Дерева рішень (або дерева класифікації та регресії) є інструментами генерації, які успішно використовуються в дослідженнях операцій та аналізі рішень. Зазвичай вони представлені орієнтованим ациклічним графом. Корінь дерева являє собою початкову точку рішення, кожен вузол містить набір умов для оцінки, а дуги показують можливі результати цих рішень.

У разі розпізнавання DA рішення зазвичай стосуються особливостей висловлювання. Під час кожного рішення порівнюється значення деякої ознаки з порогом. Наприклад, на рисунку 2.4 показано три різні просодичні ознаки (sf, ld і ldp) з відповідними порогами (T, T_1, T_2 і T_{12}). sf – це функція типу паузи, а ld і ldp – функції типу тривалості. В та А відповідають класам діалогових актів. Навчання дерева рішень виконується автоматично на навчальному корпусі. Результатом дерева є ймовірність DA з урахуванням особливостей висловлювання (лексичних і просодичних), тобто апостеріорна ймовірність $P(C|W,F)$. Основна перевага дерев рішень полягає в тому, що вони можуть поєднувати різні дискретні та безперервні функції.

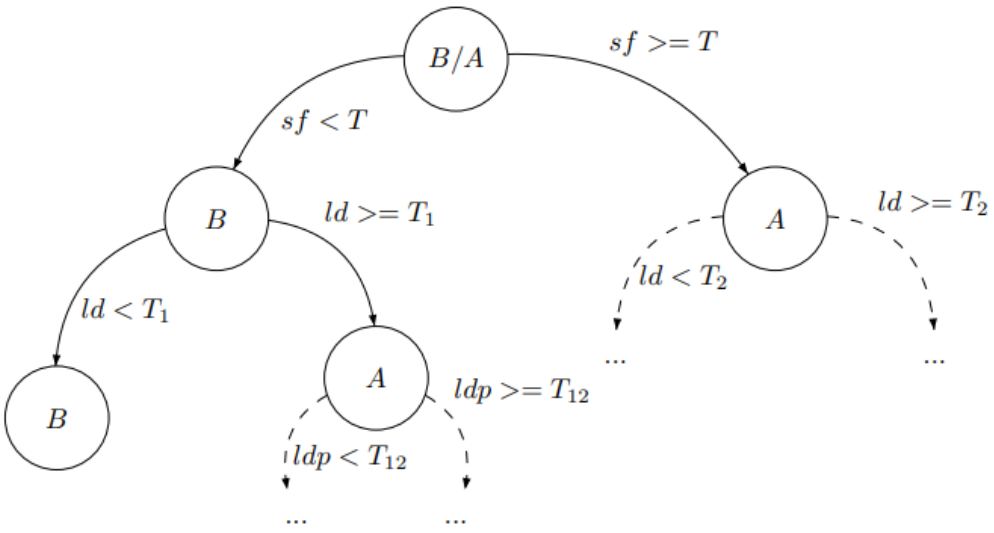

Рисунок 2.4 – Приклад дерева рішень для задачі класифікації DA.

2.3.2 Навчання на основі пам'яті

Навчання на основі пам’яті (MBL) є застосуванням теорії міркувань на основі пам’яті в області машинного навчання. Ця теорія заснована на припущенні, що можна обробляти новий зразок, порівнюючи його із збереженими уявленнями попередніх зразків. Отже, в MBL всі відомі зразки зберігаються в пам’яті для подальшого використання, а будь-який невідомий зразок класифікується шляхом порівняння з усіма збереженими зразками. Основна перевага MBL порівняно з іншими методами машинного навчання полягає в тому, що він успішно керує винятками та прихованими зв’язками в даних. Основним недоліком методу є його високі вимоги до пам’яті та обчислювальних процесів.

Для порівняння збережених і розпізнаних зразків можна використовувати кілька методів. Найпопулярнішим є k-найближчих сусідів (k-NN). Він полягає у визначенні міри відстані між зразками та пошуку k збережених екземплярів, які мають найменшу відстань до цільового зразка. Передбачається, що ці k зразків подібні до визнаного зразку, а визнаний зразок класифікується як домінуючий клас серед цих «сусідів».

Ротару в своєму досліджені [21] використовує MBL для автоматичної розмітки діалогових актів на корпусі Switchboard [6], який складається з спонтанної телефонних розмов між людьми. Характеристики висловлювання засновані на біграмах слів, обчислених для всього навчального корпусу. Ці біграми хешуються до заданої кількості ознак, оптимальне значення яких знайдено експериментально. Хеш-функція використовує літери, присутні в біграмах, і кількість ознак. Автор експериментує з різною кількістю сусідів. Найкращий результат – близько 72% точності з трьома сусідами.

2.3.3 Навчання на основі трансформацій

Основна ідея навчання на основі трансформацій (TBL) полягає в тому, щоб почати з деякого простого рішення проблеми і застосувати перетворення для отримання кінцевого результату. З огляду на проанотований навчальний корпус і набір можливих шаблонів перетворення в цьому корпусі, усі можливі перетворення генеруються з шаблонів, після чого перетворення вибираються ітераційно. Шаблони можуть мати наступний вигляд: якщо тег X стоїть після тегу Y та/або N попередніх висловлювань містять слово w, то змінити фактичний тег на Z. На кожному кроці вибирається «найкраща» трансформація (з найбільшою точністю) і застосовується до поточного рішення. Алгоритм зупиняється, коли вибране перетворення недостатньо змінює дані або коли перетворень більше не залишилося.

Загальна кількість усіх можливих перетворень може бути дуже великою. Таким чином, перевірка всіх перетворень часто є дорогою з точки зору обчислень, тим більше, що більшість з перетворень не покращують точність. Для вирішення цієї проблеми можна використовувати підхід Монте-Карло: лише фіксована кількість перетворень вибирається випадковим чином і використовується на наступних кроках. Хоча це може

виключити найкраще перетворення із збереженого набору, зазвичай залишається достатньо перетворень, щоб одне з них все ще принесло значне підвищення точності.

TBL можна застосувати до більшості завдань класифікації, в тому числі для автоматичного розпізнавання DA. Наприклад у наведеному досліджені використовується TBL зі стратегією Монте-Карло на корпусі VERBMOBIL. Автори використовують такі функції висловлювання для розпізнавання DA: ключові фрази, n-грами слів, ідентичність мовця, розділові знаки, попередній акт діалогу тощо. Отримана точність DA становить близько 71% [22].

2.3.4 Підходи на основі багатошарового перцептрону

Однією з найбільш часто використовуваних моделей нейронної мережі в області розпізнавання DA є багатошаровий перцептрон (MLP), який складається з набору вхідних вузлів, що утворюють вхідний рівень, одного або кількох прихованих шарів обчислювальних вузлів, і одного вихідного шару. Вхідний сигнал поширюється по мережі пошарово. MLP може представляти нелінійну функцію. Приклад багатошарового перцептрона зображено на рисунку 2.5.

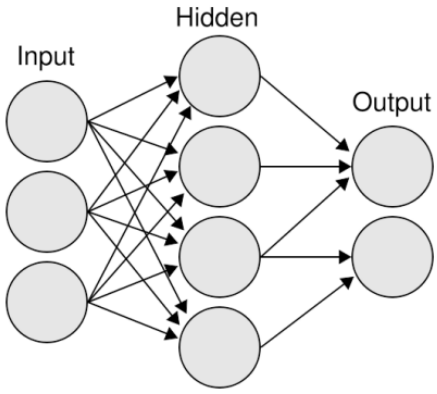

Рисунок 2.5 – Приклад багатошарового прецептрона

У своєму дослідженні [16] Райт описує підхід з одношаровим MLP. В якості вхідних даних використовуються 54 супрасегментні та тривалі просодичні ознаки. Вона досягає 62% точності на корпусі DCIEM з 12 класами DA. Риз успішно використовує в [18] MLP як окремо, так і в поєднанні з ПММ. Він отримує подібну точність (близько 76%) на іспанському корпусі CallHome з обома налаштуваннями. Левін та інші в своєму досліджені використовують набір бінарних функцій для навчання MLP [23]. Ці функції обчислюються автоматично шляхом поєднання фразового аналізу на основі граматики та методів машинного навчання. Вони отримують точність розпізнавання DA близько 71% для англійської та близько 69% для німецької мов для датасету NESPOLE.

2.3.5 Підходи на основі карт Кохонена

Іншим типом нейронної мережі, що використовується в області класифікації актів діалогу, є мережа Кохонена [24], також відома як самоорганізаційна карта (SOM). Вона визначає впорядковане відображення, свого роду проекцію з набору заданих елементів даних на, зазвичай, двовимірну сітку.

Топологія SOM – це одношарова мережа з прямим зв'язком, де дискретні виходи впорядковані в низькорозмірну (зазвичай 2D або 3D) сітку. Кожен вхід пов’язаний з усіма вихідними нейронами. Вектор вагів з такою ж розмірністю, як і вхідні вектори, приєднується до кожного нейрона. Кількість вхідних розмірів зазвичай набагато більше, ніж вихідна сітка. SOM в основному використовуються для зменшення розмірності.

Моделі мережі Кохонена оцінюються за алгоритмом SOM [25]. Елемент даних відображається на вузол, модель якого найбільш схожа на елемент даних, тобто має найменшу відстань до елемента даних, на основі деякої метрики.

Мережі Кохонена для розпізнавання діалогових актів

використовуються в [26]. Її топологія зображена на рисунку 2.6. Автори використовують сім поверхневих ознак висловлювання: мовець, режим речення, наявність чи відсутність слів-маркерів відкритих запитань, наявність чи відсутність знаку питання, тощо. Кожне висловлювання представлено шаблоном цих ознак, який кодується у двійковому форматі. Спочатку точна кількість класів DA не відома апріорі. Процес кластеризації переривається після того, як буде знайдено задану кількість кластерів.

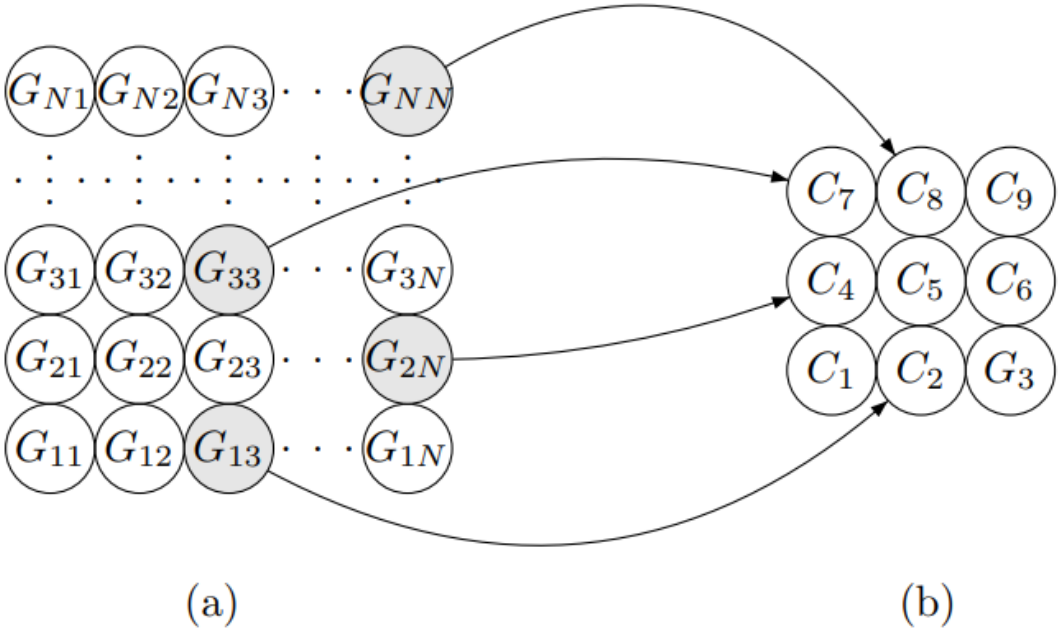

Рисунок 2.6 – Мережа Кохонена для класифікації DA

Для інтерпретації кластерів будується ще одна невелика мережа Кохонена (права модель на малюнку 2.6). Ця мережа містить стільки ж нейронів, скільки класів DA. Ці нейрони ініціалізуються значеннями векторів ваги репрезентативних нейронів великої мережі.

Як правило, методи з навчанням без вчителя, такі як мережі Кохонена, рідко використовуються для розпізнавання DA.

2.4 Підхід на основі рекурентних нейронних мереж

Рекурентні нейронні мережі або RNN, як їх коротко називають, є дуже важливим варіантом нейронних мереж, які активно використовуються в обробці природної мови. Концептуально вони відрізняються від стандартної нейронної мережі, оскільки стандартним введенням в RNN є слово, а не вся вибірка, як у випадку стандартної нейронної мережі. Це дає можливість мережі працювати з реченнями різної довжини, чого неможливо досягти в стандартній нейронній мережі через її фіксовану структуру. Це також надає додаткову перевагу спільного використання функцій, засвоєних у різних позиціях тексту, які неможливо отримати в стандартній нейронній мережі.

RNN розглядає кожне слово речення як окремий вхід, що відбувається в момент t, і також використовує значення активації в $t-1$, як вхід на додаток до введення в момент t. На рисунку 2.7 показано детальну структуру архітектури RNN.

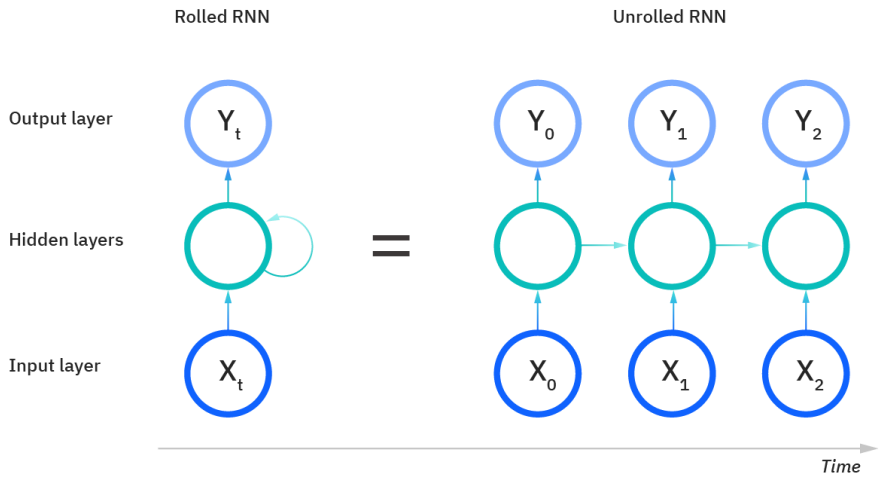

Рисунок 2.7 – Структура рекуретної нейроної мережі

Архітектура, зображена вище, також називається архітектурою «багато до багатьох» з $T_x = T_y$, тобто кількість входів дорівнює кількості

виходів. Така структура є досить корисною для моделювання послідовностей. Крім згаданої вище архітектури, є інші типи архітектур RNN, які зазвичай використовуються.

Архітектура «багато до одного» відноситься до архітектури RNN, де багато входів (T_x) використовуються для надання одного виходу (T_y). Приклад наведено на рисунку 2.8. Типовим прикладом використання такої архітектури буде завдання класифікації.

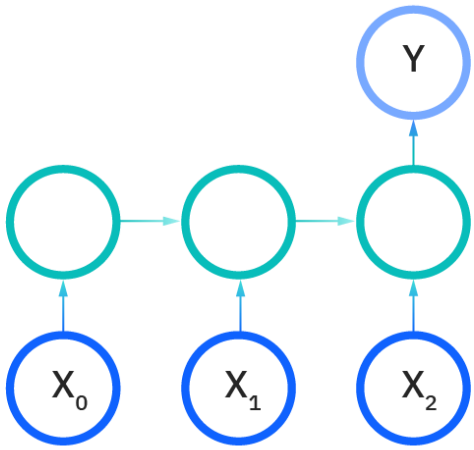

Рисунок 2.8 – RNN «багато до одного»

Архітектура «один до багатьох» відноситься до ситуації, коли RNN генерує серію вихідних значень на основі одного вхідного значення. Структура такої мережі наведена на рисунку 2.9.

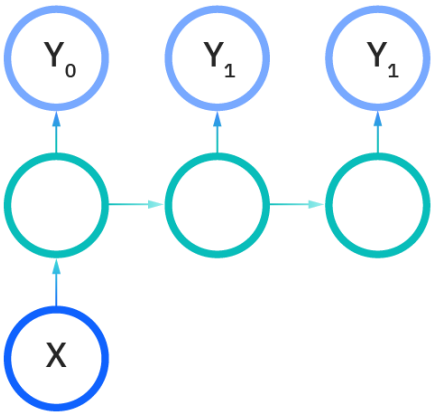

Рисунок 2.9 – RNN «один до багатьох»

Яскравим прикладом використання такої архітектури буде завдання генерації музики, де вхідним є жанр або перша нота.

Архітектура «багато до багатьох» де $T_x \neq T_y$ відноситься до того, де багато входів зчитуються для отримання багатьох виходів, але довжина входів не дорівнює довжині виходів. Яскравим прикладом використання такої архітектури є завдання машинного перекладу. Структра зображена на рисунку 2.10.

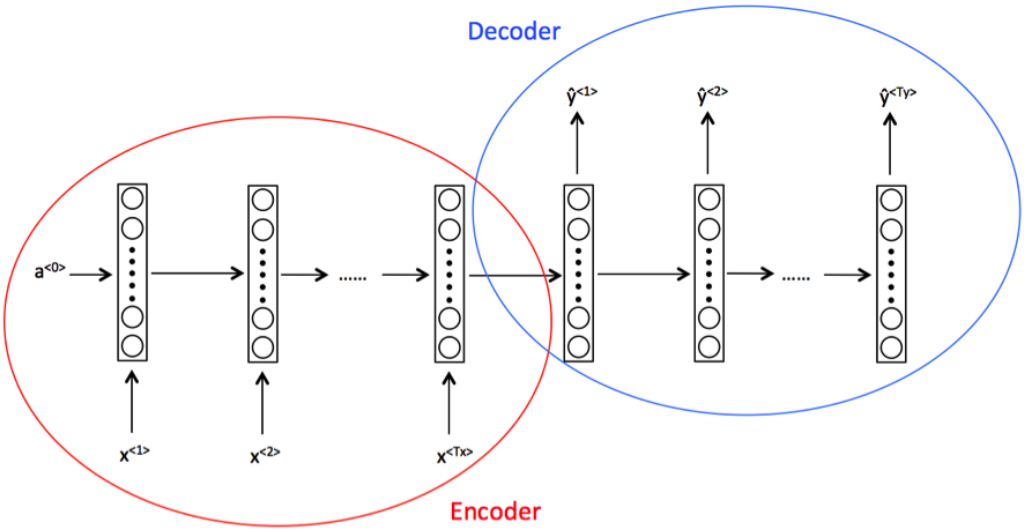

Рисунок 2.10 – Альтернативна RNN «багто до багатьох»

На ньому кодер (Encoder) відноситься до частини мережі, яка читає речення, яке потрібно перекладати, а декодер (Decoder)– це частина мережі, яка перекладає речення на бажану мову.

Незважаючи на усі переваги рекурентних нейроних мереж, вони мають певні недоліки. До основних можна віднести:

1. приклади архітектури RNN, наведені вище, здатні фіксувати залежності лише в одному напрямку мови. В основному, у випадку обробки природної мови передбачається, що слово, що йде після, не впливає на значення слова, що йде перед. Проте враховуючи структури мов, можна стверджувати, що це твердження не завжди правильне;

2. RNN також не дуже добре фіксує довгострокові залежності та

проблему зникаючих градієнтів, які знову з’являються в RNN.

Обидва ці обмеження породжують нові типи архітектур RNN, які обговорюються нижче.

2.4.1 Вентильний рекурентний вузол

Вентильний рекурентний вузол (GRU) – це модифікація основного рекурентного блоку, яка допомагає фіксувати залежності на великій відстані, а також дуже допомагає у вирішенні проблеми зникаючого градієнта [27].

GRU містить додатковий блоку пам’яті, який зазвичай називають шлюзом оновлення або шлюзом скидання. Крім звичайного нейронного блоку з сигмовидною функцією та нормованою експоненційною функцією (softmax) для виведення, він містить додатковий блок із гіперболічним тангенсом (tanh) як функцією активації. Tanh використовується, оскільки його вихід може бути як позитивним, так і негативним, отже, його можна використовувати як для збільшення, так і для зменшення. Вихід з цього пристрою потім об’єднується з вхідним сигналом активації для оновлення значення комірки пам’яті. Математично процес функціонування вентильного вузла можна подати наступним чином:

$$\begin{aligned} z_t &= \sigma_g(W_z \cdot x_t + U_z \cdot h_{t-1} + b_z) \\ r_t &= \sigma_g(W_r \cdot x_t + U_r \cdot h_{t-1} + b_r) \\ h_t &= z_t \circ h_{t-1} + (1 - z_t) \circ \sigma_h(W_h \cdot x_t + U_h \cdot (r_t \circ h_{t-1}) + b_h) \end{aligned} \tag{2.6}$$

де x_t – вектор входу;

h_t – вектор виходу;

z_t – вектор вузла уточнення;

r_t – вектор вузла скидання;

W, U, b – матриці та вектор параметрів;

σ_g – сигмоїдна функція активації;

σ_h – гіперболічний тангенс.

Архітектура вентильниного рекурентного вузла зображено на рисунку 2.11.

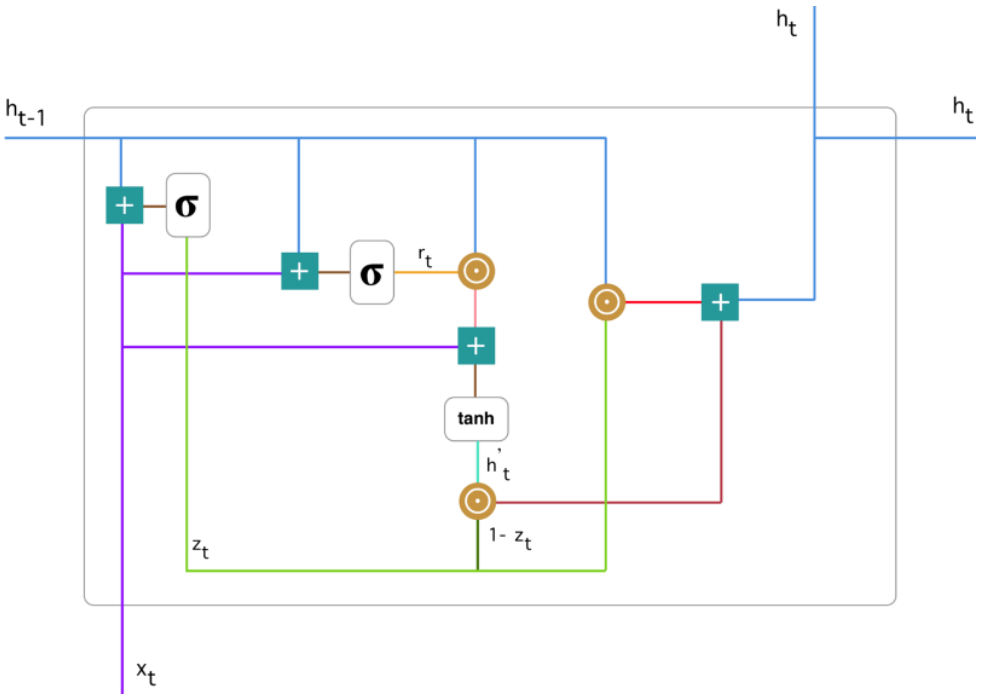

Рисунок 2.11 – Архітектура GRU

2.4.2 Двонаправлені рекурентні нейронні мережі

У наведених вище архітектурах RNN можуть бути враховані ефекти входжень лише за попередні часові позначки. У випадку NLP це означає, що модель враховує наслідки слова, написаного тільки перед поточним словом. Але це не так у мовній структурі, тому на допомогу приходить двонаправлена RNN (BRNN).

Двонаправлена RNN складається з прямої та зворотної рекуррентної нейронної мережі, і остаточне передбачення робиться, об’єднуючи результати обох мереж у будь-який момент часу t, як можна побачити на зображенні. 2.12.

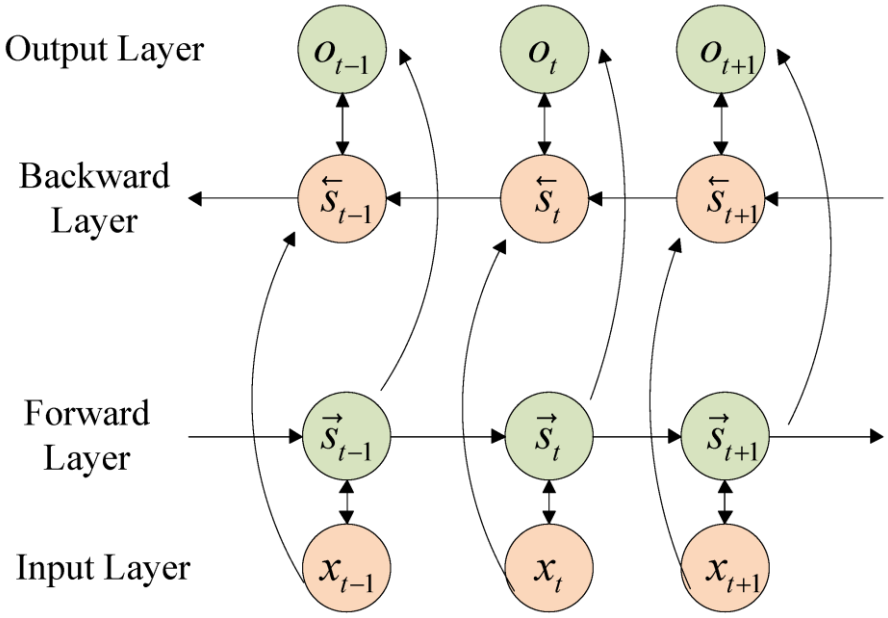

Рисунок 2.12 – Двонаправлена RNN

Навчання BRNN подібне до алгоритму зворотного поширення через час (BPTT). BPTT – це алгоритм зворотного поширення, який використовується під час навчання RNN. Типовий алгоритм BPTT працює наступним чином:

1. розгортання мережі та обчислення помилки на кожному кроці часу;
2. згортання мережі та оновлення ваги.

Однак у BRNN, оскільки прямі та зворотні проходи відбуваються одночасно, оновлення ваг для двох процесів може відбуватися в один і той же момент часу. Це призводить до помилкових результатів. Таким чином, для розміщення прямих і зворотних проходів окремо, для навчання BRNN використовується трохи інший алгоритм.

Для прямого проходу спочатку подаються подальші стани і попередні стани, а потім – вихідні нейрони. Для зворотного проходу спочатку проходять вихідні нейрони, після чого передаються подальші і попередні стани. Після проходження вперед і назад, ваги оновлюються.

2.4.3 Механізм уваги

Механізм уваги (attention) пропонується як рішення обмеження моделі кодер-декодер, яка кодує вхідну послідовність до одного вектора фіксованої довжини, з якого декодується вихідний результат на кожному кроці часу [28]. Вважається, що ця проблема з'являється під час декодування довгих послідовностей, оскільки нейронній мережі важко справлятися з довгими реченнями, особливо тими, які довші за речення в навчальному корпусі. При використанні уваги коли модель намагається передбачити наступне слово, вона шукає набір позицій у вихідному реченні, де зосереджена найбільш релевантна інформація. Потім модель прогнозує наступне слово на основі контекстних векторів, пов'язаних з цими вихідними позиціями та всіма попередніми згенерованими цільовими словами.

Замість того, щоб кодувати вхідну послідовність в один фіксований вектор контексту, модель уваги розробляє контекстний вектор, який фільтрується спеціально для кожного кроку вихідного часу.

Таким чином вхід декодера тепер отримує такі значення:

1. попередній прихований стан моделі декодера H_{k-1};
2. попередній вихід моделі декодера Y_{k-1};
3. вектор контексту C_k– зважена сума всіх прихованих станів кодера (h_j).

Занченя вектору контексту обчсилюється за формулою 2.7.

$$c_i = \sum_{j=1}^{T_x} a_{ij} \cdot h_j \tag{2.7}$$

де c_i – елемент вектору контексту;

h_j– прихований стан кодера;

a_{ij}– глобальні ваги вирівнювання;

T_x– кількість кроків у моделі кодера.

Схематичне функціонування моделі кодера-декодера з увагою зображено на рисунку 2.13.

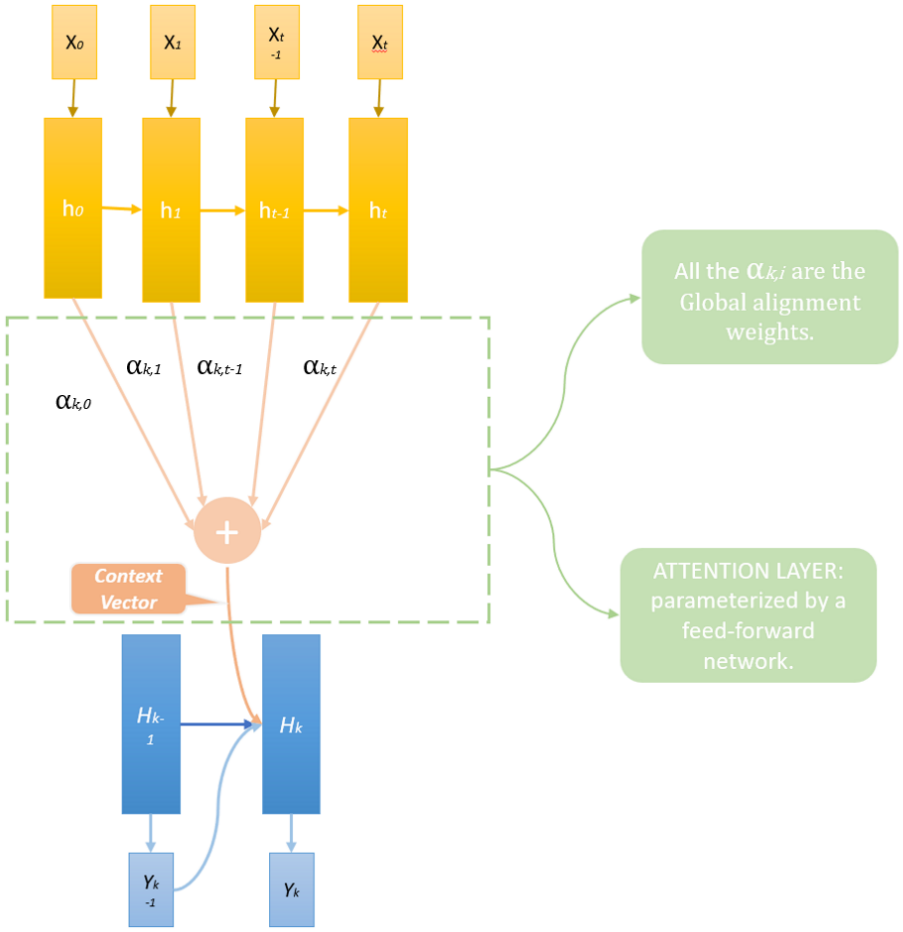

Рисунок 2.13 – Структура моделі кодера-декодера з увагою

Ще одинм видом уваги є самоувага. Вона визначє увагу тієї ж послідовності. Замість того, щоб шукати асоціацію/вирівнювання послідовності введення-виведення, ми тепер шукаємо оцінки між елементами послідовності, як показано на рисунку 2.14.

Модуль самоуваги приймає n входів і повертає n виходів. Механізм самоуваги дозволяє входам взаємодіяти один з одним і з'ясовувати, на кого вони повинні приділяти більше уваги. Вихідними результатами є сукупність

цих взаємодій і показників уваги.

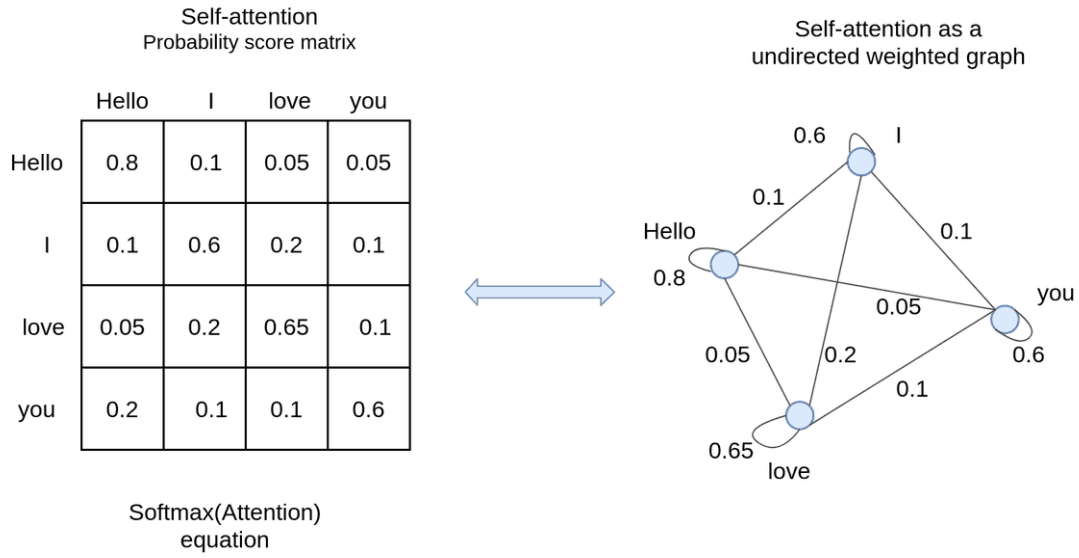

Рисунок 2.14 – Ілюстарція самоуваги

Використання уваги здатно значно покращити результати моделі. По-перше, це зазвичай усуває проблему зникаючого градієнта, оскільки увага забезпечують прямі зв’язки між станами кодера та декодера. Концептуально вона діє подібно до пропуску зв’язків у згорткових нейронних мережах.

Іншою перевагою є зрозумілість. Перевіряючи розподіл ваг уваги, ми можемо отримати уявлення про поведінку моделі, а також зрозуміти її обмеження.

Єдиним недоліком механізму уваги є те, що він займає дуже багато часу і його важко реалізувати в паралельний шлях .

2.4.4 Модель BERT

BERT (Bidirectional Encoder Representations from Transformers) – це модель представлення мови, опублікована дослідниками Google AI Language [29]. Вона викликав резонанс у спільноті машинного навчання, представивши найсучасніші результати в широкому спектрі завдань NLP, включаючи відповідь на запитання, виведення з природної мови та інші.

Ключовою технічною інновацією BERT є застосування двонаправленого навчання Transformer, популярної моделі, до мовного моделювання. Результати роботи показують, що мовна модель, яка навчається двосторонньо, може мати більш глибоке відчуття мовного контексту та потоку, ніж односпрямовані мовні моделі. У статті дослідники докладно описують нову техніку під назвою Masked LM (MLM), яка дозволяє двонаправлене навчання в моделях, у яких раніше це було неможливо.

На відміну від спрямованих моделей, які зчитують введений текст послідовно (зліва направо або справа наліво), кодер Transformer зчитує всю послідовність слів одночасно. Тому він вважається двонаправленим, хоча точніше було б сказати, що він ненаправлений. Ця характеристика дозволяє моделі вивчати контекст слова на основі всього його оточення (ліворуч і праворуч від слова).

Вхідними є послідовність токенів, які спочатку вбудовуються у вектори, а потім обробляються в нейронній мережі. Вихідним є послідовність векторів розміру H, в якій кожен вектор відповідає вхідному токену з тим самим індексом.

Під час навчання мовних моделей виникає проблема визначення мети передбачення. Багато моделей передбачають наступне слово в послідовності наприклад: «Дитина повернулася додому з ___». Це спрямований підхід, який за своєю суттю обмежує вивчення контексту. Щоб подолати цю проблему, BERT використовує дві стратегії навчання.

Перша має назву Masked LM. Перед подачею послідовностей слів у BERT 15% слів у кожній послідовності замінюються маркером [MASK]. Потім модель намагається передбачити вихідне значення замаскованих слів на основі контексту, наданого іншими, незамаскованими словами в послідовності. Рисунок 2.15 є абстрактним зображенням кодера Transformer.

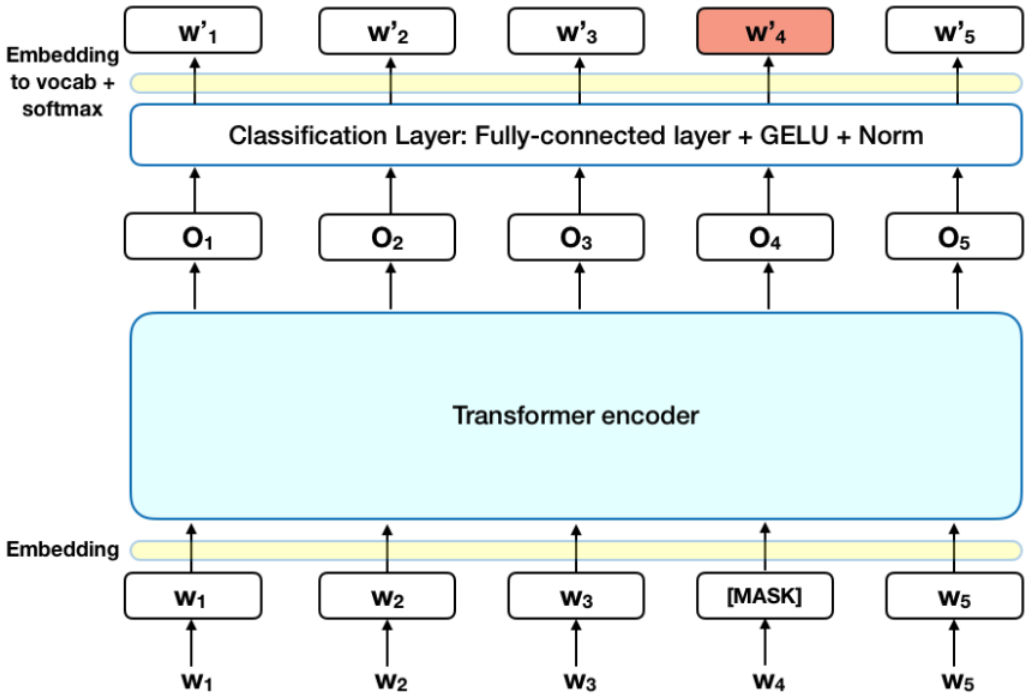

Рисунок 2.15 – Структура Transformer

Друга стратегія – прогноз наступного речення(NSP). У процесі навчання BERT модель отримує пари речень в якості вхідних даних і вчиться передбачати, чи є друге речення в парі наступним реченням у оригінальному документі. Під час навчання 50% введених даних є парою, в якій друге речення є наступним реченням в оригінальному документі, тоді як в інших 50% випадкове речення з корпусу вибирається другим реченням. Передбачається, що випадкове речення буде від'єднано від першого речення. Схема навчання надана на рисунку 2.16.

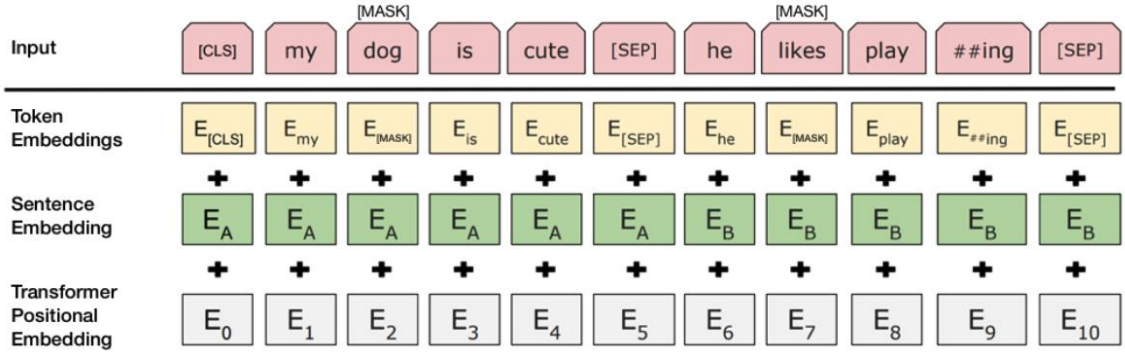

Рисунок 2.16 – Схема навчання BERT

Під час навчання моделі BERT, MLM і NSP навчаються разом, з метою мінімізації комбінованої функції втрат двох стратегій.

BERT можна використовувати для різноманітних мовних завдань, додаючи лише невеликий шар до основної моделі:

1. завдання класифікації, такі як аналіз настроїв, виконуються подібно до класифікації наступного речення, шляхом додавання шару класифікації поверх вихідних даних Transformer для маркера [CLS] ;

2. у завданнях «Відповідь на запитання» застосунок отримує запитання щодо текстової послідовності та має позначити відповідь у послідовності. Використовуючи BERT, модель запитань і відповідей можна навчити, вивчаючи два додаткові вектори, які позначають початок і кінець відповіді;

3. у розпізнаванні іменованих об’єктів (NER) застосунок отримує текстову послідовність і має позначати різні типи сутностей (особа, організація, дата тощо), які з’являються в тексті. Використовуючи BERT, модель NER можна навчити, подаючи вихідний вектор кожного токена в рівень класифікації, який прогнозує мітку NER.

3 ЕКСПЕРИМЕНТАЛЬНЕ МОДЕЛЮВАННЯ ТА НАВЧАННЯ МОДЕЛІ КЛАСИФІКАЦІЇ DA

3.1 Архітектура моделі

Як показано у попередньому розділі, рекурентні нейроні мережі та їх модифікації дуже добре зарекомендували себе в вирішені різноманітних задач NLP. Саме тому буде логічно обрати саме їх в якості бази для запропонованої моделі. Архітектура моделі зображена на рисунку 3.1.

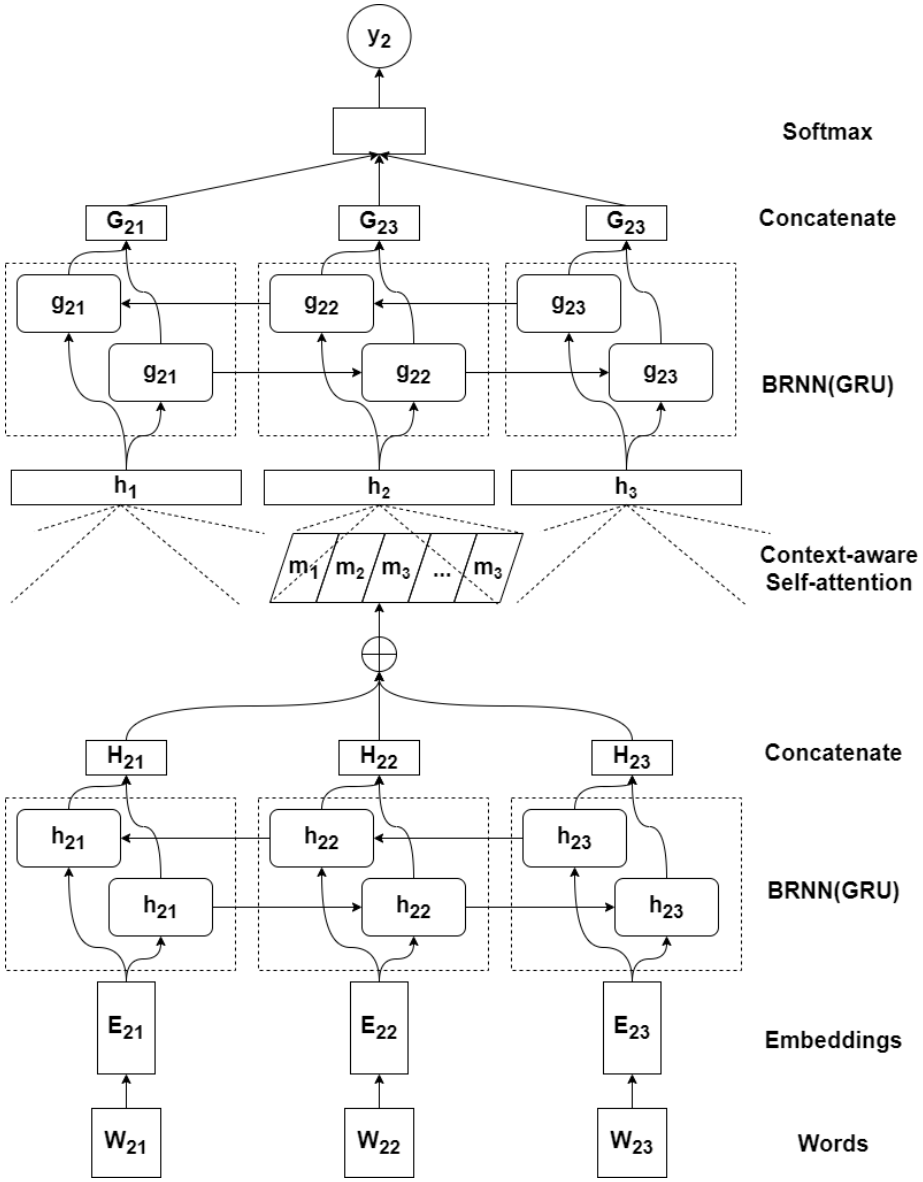

Рисунок 3.1 – Архітектура запропонованої моделі

Модель складається з трьох основних компонентів: RNN на рівні висловлювання, який кодує інформацію у висловлюваннях на рівні слова та символу, контекстно-свідомий механізм самоуваги, який об’єднує репрезентації слів у репрезентації висловлювань та RNN рівня розмови, за яким слідує шар класифікатора для визначення класу діалогових актів висловлювання.

3.1.1 RNN рівня висловлювання

На цьому шарі слова вхідної репліки кодуються в так звані ембедінги – векторні подання. Для цього використовується модель RoBERTa [30]. Ця модель є вдосконаленням моделі BERT командою Facebook AI Research. До основних відмінностей цієї моделі від BERT можна віднести спрощену систему навчання та значно збільшений корпус даних для навчання. RoBERTa демонструє від 2 до 20% покращення результату порівняно до BERT в залежності від завдання та застосування.

Таким чином кожне слово репліки кодується в ембедінг. Потім йде двонаправлений шар GRU (BGRU). Об’єднання прямих і зворотних вихідних даних BGRU генерує ембедінги висловлювання, які служить вхідним сигналом для контекстно-свідомого механізму самоуваги на рівні висловлювання, який вивчає остаточне представлення висловлювання.

3.1.2 Контекстно-свідомий шар з самоувагою

Самоуважні уявлення кодують послідовність змінної довжини у фіксований розмір, використовуючи механізм уваги, який враховує різні позиції в послідовності. Ми використовуємо попередній прихований стан з рівня RNN рівня висловлювання, який надає контекст розмови на даний момент, і об’єднуємо його з прихованими станами всіх складових слів у висловлюванні в самоуважний кодер, який обчислює 2D представлення

кожного вхідного висловлювання.

Репліка u_i, яку можна подати як послідовність слів $\{w_i^1, w_i^2, w_i^3, \dots, w_i^n\}$, за допомогою шару емебдінгів кодується в вектор розмірності d для кожного слова. Потім отримані вектори передаються до шару BGRU, приховані виходи якого об'єднуються для кожного кроку часу. Формально ці кроки можна зобразити формулами 3.1.

$$\begin{aligned} \overrightarrow{h_i^j} &= \overrightarrow{GRU}(w_i^j \cdot \overrightarrow{h_i^{j-1}}) \\ \overleftarrow{h_i^j} &= \overleftarrow{GRU}(w_i^j \cdot \overleftarrow{h_i^{j+1}}) \\ h_i^j &= concat\left(\overrightarrow{h_i^j}, \overleftarrow{h_i^j}\right) \\ H_i &= \{h_i^1, h_i^2, h_i^3, \dots, h_i^n\} \end{aligned} \tag{3.1}$$

де H_i – n-ий вихід BGRU.

Далі контекстні оцінки самоуваги S_i обчислюються за формулою 3.2.

$$S_i = W_{s2} \cdot (W_{s1} \cdot H_i^T + W_{s3} \cdot \overrightarrow{g_{i-1}} + b) \tag{3.2}$$

де W_{s1}, W_{s2}, W_{s3} – матриці вагів відповідних розмірів;

b – зміщення(баєс);

$\overrightarrow{g_{i-1}}$ – прихований стан RNN рівня розмови.

Рівняння 3.2 можна розглядати як двошаровий MLP зі зміщенням та W_{s1}, W_{s2} та W_{s3} в якості вагових параметрів. Оцінки S_i відображаються в матрицю ймовірностей A_i за допомогою функції софтмакс (формула 3.3).

$$A_i = softmax(S_i) \tag{3.3}$$

Потім ця матриця ймовірностей використовується для отримання двовимірного представлення M_i вхідного висловлювання, використовуючи

приховані стани GRU H_i відповідно до вагових коефіцієнтів уваги, наданих A_i за формулою 3.4.

$$M_i = A_i \cdot H_i \tag{3.4}$$

Це двовимірне уявлення потім проектується на одновимірний ембедінг (позначається як h_i), використовуючи повнозв'язаний шар. Потім за допомогою RNN рівня розмови цей ембедінг перетворюється на g_{i-1}за формулою 3.5.

$$\begin{aligned} \overrightarrow{g_i} &= \overrightarrow{GRU}(h_i \cdot \overrightarrow{g_{i-1}}) \\ \overleftarrow{g_i} &= \overleftarrow{GRU}(h_i \cdot \overleftarrow{g_{i+1}}) \\ g_i &= concat(\overrightarrow{g_i}, \overleftarrow{g_i}) \end{aligned} \tag{3.5}$$

Вектор g_i надає контекст розмови, який використовується для вивчення показників уваги та двовимірного представлення M_{i+1} для наступного висловлювання в розмові h_{i+1}.

3.1.3 RNN рівня розмови

Представлення висловлювання з попереднього кроку передається на рівень RNN рівня розмови, який є іншим двонаправленим шаром GRU, який використовується для кодування висловлювань у розмові. Приховані стани об'єднуються, щоб отримати остаточне уявлення G_i кожного висловлювання, яке далі поширюється на шар класифікатора. Він в свою чергу складається з трьох повноз'єднаних шарів з функцією активації нещільний випрямлений лінійний вузол. Перші два шари необхідні для зменшення розмірності мережі. Обраний датасет має 43 класи тому виход нейронної мережі має відповідну розмірність.

Розмірності ключових шарів наведено в таблиці 3.1.

Таблиця 3.1 – Розмірності шарів мережі

Назва шару	**Вхідна розмірність**	**Вихідна розмірність**
RoBERTa Embeddings	256	768
BRNN	768	768*2
ContextAwareAttention	1536	1
BRNN	1	768*2
Linear	1536	256
Linear	256	128
Linear	128	43

3.2 Навчальний корпус

В якості набору даних для навчання та тестування моделі було обарно SwDA (Switchboard Dialogue Act Corpus) [6]. Switchboard – це набір з близько 2400 телефонних розмов між 543 спікерами (302 чоловіки, 241 жінка) з усіх регіонів Сполучених Штатів. Автоматичний комп’ютеризований оператор обробляв дзвінки, надаючи абоненту відповідні записані підказки, вибираючи та набираючи іншу особу (абонента) для участі в розмові, вводячи тему для обговорення та записуючи промову двох суб’єктів на окремі канали. Було подано близько 70 тем, з яких близько 50 були найбільш розповсюджені. Вибір тем і тих, хто викликає, був обмежений таким чином, щоб: (1) два доповідачі не говорили разом більше одного разу і (2) ніхто не говорив більше одного разу на певну тему.

SwDA є модифікацією оригінального датасету, кожна розмова була проанотована з використанням набору тегів DAMSL. У кодуванні було використано 220 тегів; 130 з них зустрічалися менше ніж 10 разів кожен, тому 220 тегів було об’єднано у 42 більших класи.

Приклад діалогу з корпусу наведено на рисунку 3.2.

Speaker	Dialogue Act	Utterance
A	Backchannel	Uh-huh.
B	Statement	About twelve foot in diameter
B	Abandoned	and, there is a lot of pressure to get that much weight up in the air.
A	Backchannel	Oh, yeah.
B	Abandoned	So it's interesting, though.
		. . .
B	Statement-opinion	it's a very complex, uh, situation to go into space.
A	Agree/Accept	Oh, yeah,
		. . .
A	Yes-No Question	You never think about that do you?
B	Yes-Answer	Yeah.
A	Statement-opinion	I would think it would be harder to get up than it would be
B	Backchannel	Yeah.

Рисунок 3.2 – Приклад діалогу з SwDA

На рисунку 3.3 наведено розподіл класів у корпусі.

Tag	Example	%
STATEMENT	*Me, I'm in the legal department.*	36%
BACKCHANNEL/ACKNOWLEDGE	*Uh-huh.*	19%
OPINION	*I think it's great*	13%
ABANDONED/UNINTERPRETABLE	*So, -/*	6%
AGREEMENT/ACCEPT	*That's exactly it.*	5%
APPRECIATION	*I can imagine.*	2%
YES-NO-QUESTION	*Do you have to have any special training?*	2%
NON-VERBAL	*<Laughter>,<Throat_clearing>*	2%
YES ANSWERS	*Yes.*	1%
CONVENTIONAL-CLOSING	*Well, it's been nice talking to you.*	1%
WH-QUESTION	*What did you wear to work today?*	1%
NO ANSWERS	*No.*	1%
RESPONSE ACKNOWLEDGMENT	*Oh, okay.*	1%
HEDGE	*I don't know if I'm making any sense or not.*	1%
DECLARATIVE YES-NO-QUESTION	*So you can afford to get a house?*	1%
OTHER	*Well give me a break, you know.*	1%
BACKCHANNEL-QUESTION	*Is that right?*	1%
QUOTATION	*You can't be pregnant and have cats*	.5%
SUMMARIZE/REFORMULATE	*Oh, you mean you switched schools for the kids.*	.5%
AFFIRMATIVE NON-YES ANSWERS	*It is.*	.4%
ACTION-DIRECTIVE	*Why don't you go first*	.4%
COLLABORATIVE COMPLETION	*Who aren't contributing.*	.4%
REPEAT-PHRASE	*Oh, fajitas*	.3%
OPEN-QUESTION	*How about you?*	.3%
RHETORICAL-QUESTIONS	*Who would steal a newspaper?*	.2%
HOLD BEFORE ANSWER/AGREEMENT	*I'm drawing a blank.*	.3%
REJECT	*Well, no*	.2%
NEGATIVE NON-NO ANSWERS	*Uh, not a whole lot.*	.1%
SIGNAL-NON-UNDERSTANDING	*Excuse me?*	.1%
OTHER ANSWERS	*I don't know*	.1%
CONVENTIONAL-OPENING	*How are you?*	.1%
OR-CLAUSE	*or is it more of a company?*	.1%
DISPREFERRED ANSWERS	*Well, not so much that.*	.1%
3RD-PARTY-TALK	*My goodness, Diane, get down from there.*	.1%
OFFERS, OPTIONS & COMMITS	*I'll have to check that out*	.1%
SELF-TALK	*What's the word I'm looking for*	.1%
DOWNPLAYER	*That's all right.*	.1%
MAYBE/ACCEPT-PART	*Something like that*	<.1%
TAG-QUESTION	*Right?*	<.1%
DECLARATIVE WH-QUESTION	*You are what kind of buff?*	<.1%
APOLOGY	*I'm sorry.*	<.1%
THANKING	*Hey thanks a lot*	<.1%

Рисунок 3.3 – Розподіл класів в SwDA

3.3 Експериментальне моделювання

Для реалізації запропонованої моделі було обрано наступні технології: мова програмування Python, фреймворк машинного навчання з відкритим кодом PyTorch, легка обгортка для високопродуктивних досліджень PyTorch Lightning та бібліотека Pandas для попередньої обробки даних. Навчання моделі відбувалося з використанням платформ Kaggle та Google Colaboratory.

3.3.1 Технології моделювання

В якості мови програмування було обрано Python оскільки на теперішній час це головний та найбільш розповсюджений інструмент у сфері машинного навчання та нейронних мереж. До переваг цієї мови можна віднести:

– легкий синтаксис та синтаксичний цукор. Програмний код на Python підлягає розумінню краще ніж код на інших високорівневих мовах програмування;

– велику кількість інструментів. Через високу розповсюдженість та популярність мова дуже активно розвивається та отримує все більшу кількість бібліотек та розширень;

– менший об’єм коду. Завдяки синтаксичній простоті програмний код на Python для виконання певної задачі є значно компактнішим порівняно до інших мов програмування.

До недоліків зазвичай відзначають: динамічну типізацію, яка підвищує час виконання та ймовірність помилок під час виконання, відсутність гарної підтримки паралельних обчислень та не ефективну роботу з пам’яттю.

PyTorch – це пакет машинного навчання Python, заснований на Torch, який є пакетом машинного навчання з відкритим вихідним кодом на основі

мови програмування Lua. PyTorch має дві основні функції:

– тензорні обчислення (наприклад, NumPy) із сильним прискоренням графічного процесора;

– автоматичне диференціювання для побудови та навчання нейронних мереж.

Тензори PyTorch дуже схожі на масиви NumPy з додаванням того, що вони можуть працювати на графічному процесорі. Це важливо, оскільки це допомагає прискорити чисельні обчислення, які можуть збільшити швидкість нейронних мереж у 50 разів або більше.

PyTorch використовує техніку під назвою автоматичне диференціювання, яка чисельно оцінює похідну функції. Автоматичне диференціювання обчислює зворотні проходи в нейронних мережах. У навчальних нейронних мережах ваги випадковим чином ініціалізуються числами близькими до нуля, але не нулями. Зворотній прохід – це процес, за допомогою якого ці ваги коригуються від виходів мережі до входів, а прямий прохід – навпаки (від входів до виходів).

Ще однієї значної перевагою цього фреймворку є наявність пакетів, що спрощують створення та використання нейронних мереж. До таких можна віднести пакет, який optim абстрагує ідею алгоритму оптимізації та надає реалізації часто використовуваних алгоритмів оптимізації, таких як AdaGrad, RMSProp та Adam. Іншим прикладом є пакет nn. Це модуль для побудови нейронних мереж в PyTorch. Він містить реалізації для найбільш розповсюджених типів шарів у нейронних мережах. До того ж є можливість пообдувати власні шари та мережі за допомогою використання механізму успадкування.

В свою чергу, PyTorch Lightning – це високорівневий фреймворк для PyTorch, який абстрагує деталі реалізації, щоб дозволяє зосередитися на створенні моделей і забути про витрачання часу на тривіальні деталі. Він значною мірою скриває другорядні деталі процесу навчання таким чином спрощуючи його.

Kaggle та Google Colaboratory надають змогу користуватися Jupyter Notebook – інтерактивною обчислювальною веб-платформою, яка об’єднує виконання програмного коду та відображення результатів у реальному часі.

Ці сервіси надають обмежений безкоштовний доступ до Jupyter блокнотів з графічними процесорами, що дозволяє підвищити швидкість навчання моделі.

3.3.2 Програмна реалізація

Як було зазначено у раніше, модель можна поділити на три концептуальні частини. Перша – RNN рівня висловлювання. Її код поданий у лістингу 3.1.

Лістинг 3.1 – Реалізація RNN рівня висловлювання

```
class UtteranceRNN(nn.Module):

    def __init__(self, model_name="roberta-base",
hidden_size=768, bidirectional=True, num_layers=1):
        super(UtteranceRNN, self).__init__()
        self.base =
AutoModel.from_pretrained(pretrained_model_name_or_path=mo
del_name)
        for param in self.base.parameters():
            param.requires_grad = False
        self.rnn = nn.GRU(
            input_size=hidden_size,
            hidden_size=hidden_size,
            num_layers=num_layers,
            bidirectional=bidirectional,
            batch_first=True
        )

    def forward(self, input_ids, attention_mask, seq_len):
        hidden_states, _ = self.base(input_ids,
attention_mask)
        outputs, _ = self.rnn(hidden_states)

        return outputs
```

Далі йде шар який реалізую самоувагу з використанням котектсу, реалізація якого наведено у лістингу 3.2.

Лістинг 3.2 – Реалізація шару самоуваги

```
class ContextAwareAttention(nn.Module):

    def __init__(self, hidden_size=1536, output_size=768,
seq_len=128):
        super(ContextAwareAttention, self).__init__()
        # context aware self attention
        self.fc_1 = nn.Linear(in_features=hidden_size,
out_features=output_size, bias=False)
        self.fc_3 = nn.Linear(in_features=hidden_size //
2, out_features=output_size, bias=True)
        self.fc_2 = nn.Linear(in_features=output_size,
out_features=128, bias=False)
        # linear projection
        self.linear_projection =
nn.Linear(in_features=hidden_size, out_features=1,
bias=True)

    def forward(self, hidden_states, h_forward):
        S = self.fc_2(torch.tanh(self.fc_1(hidden_states)
+ self.fc_3(h_forward.unsqueeze(1))))
        A = S.softmax(dim=-1)
        M = torch.matmul(A.permute(0, 2, 1),
hidden_states)
        x = self.linear_projection(M)
        return x
```

Найбільш цікавим є останній шар, який об’єднує попередні та координує їх обчислення. Лістинг 3.3 демонструє його реалізацію.

Лістинг 3.3 – Реалізація останнього шару моделі

```
class ContextAwareDAC(nn.Module):
    def      __init__(self,      model_name="roberta-base",
hidden_size=768,                          num_classes=18,
device=torch.device("cpu")):
        super(ContextAwareDAC, self).__init__()
        self.in_features = 2 * hidden_size
        self.device = device
        self.utterance_rnn                                =
UtteranceRNN(model_name=model_name,
hidden_size=hidden_size)
        self.context_aware_attention                      =
ContextAwareAttention(hidden_size=2 * hidden_size,
```

Продовження лістинг 3.3

```
output_size=hidden_size, seq_len=128)
        self.conversation_rnn                                    =
ConversationRNN(input_size=1, hidden_size=hidden_size)
        self.classifier = nn.Sequential(*[
            nn.Linear(in_features=self.in_features,
out_features=256),
            nn.LeakyReLU(),
            nn.Linear(in_features=256, out_features=128),
            nn.LeakyReLU(),
            nn.Linear(in_features=128,
out_features=num_classes)
        ])
        self.hx   =   torch.randn((2,   1,   hidden_size),
device=self.device)
    def forward(self, batch):
        outputs                                                  =
self.utterance_rnn(input_ids=batch['input_ids'],
attention_mask=batch['attention_mask'],
seq_len=batch['seq_len'].tolist())
        features   =   torch.empty((0,   self.in_features),
device=self.device)
        hx = self.hx
        for i, x in enumerate(outputs):
            x = x.unsqueeze(0)
            m                                                    =
self.context_aware_attention(hidden_states=x,
h_forward=hx[0].detach())
            hx      =      self.conversation_rnn(input_=m,
hx=hx.detach())
            features  =  torch.cat((features,  hx.view(1,  -
1)), dim=0)
        self.hx = hx.detach()
        logits = self.classifier(features)
        return logits
```

3.4 Аналіз результатів дослідження

Запропонована модель навчалася протягом 10 поколінь. Приблизний час навчання одного покоління з використанням прискорених графічних процесорів складає 2,5 години. В результаті було отримано максимальну точність в 75,35% для тестової вибірки та 76,62% для валідаційної.

Графік зміни точності та функції втрати зображено на рисунках 3.4 та 3.5. Для відображення загальної тенденції було застосовано згладжування.

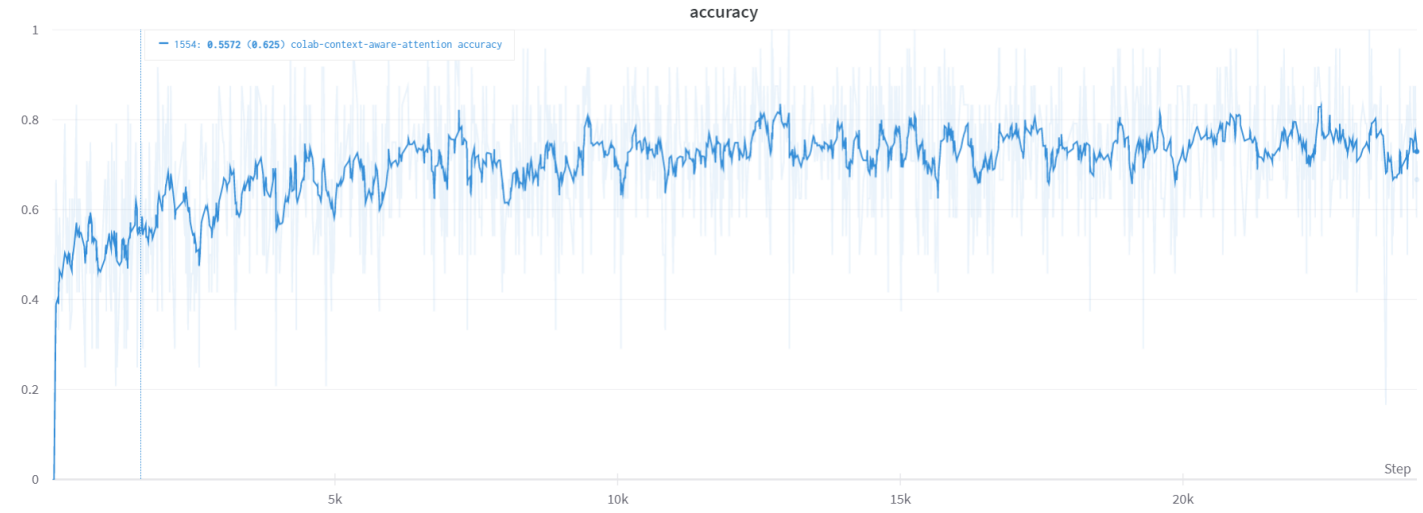

Рисунок 3.4 – Графік зміни точності моделі

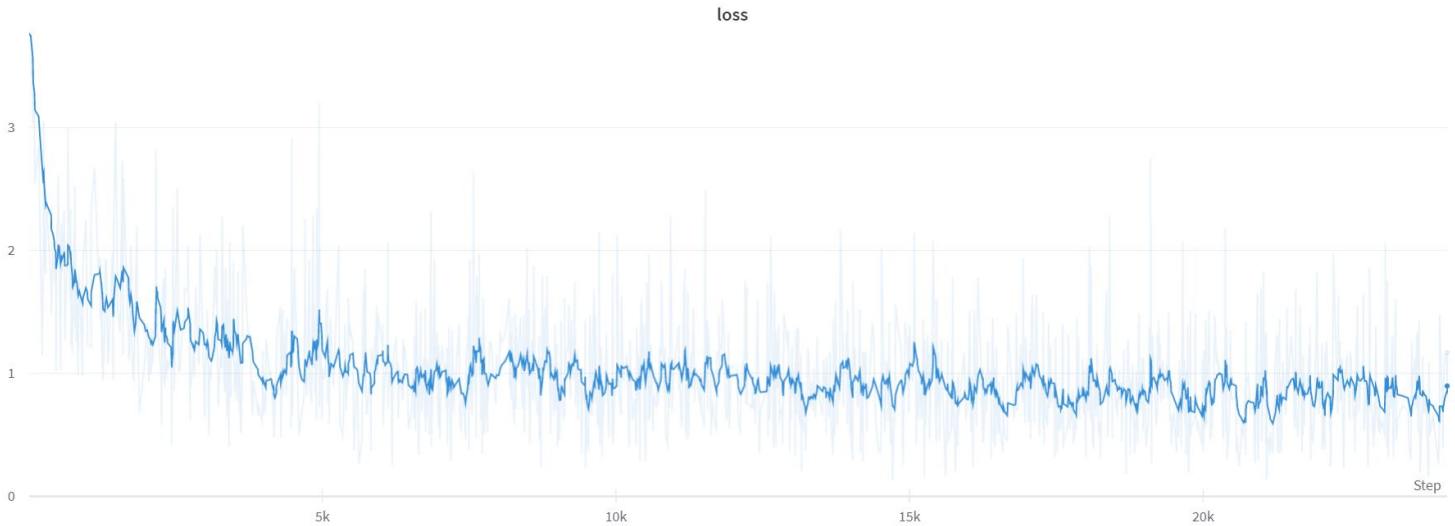

Рисунок 3.5 – Графік зміни функції втрат

Отримані результати не є найкращими на сьогоднішній день. Отримана точність перевершує результати отримані в [31] на 2,2%, про те все ще менша ніж в [32], де точність моделі складає 81,3%.

Таким чином постає питання про подальше вдосконалення моделі для отримання кращих результатів. Серед потенційних шляхів подальших досліджень можна розглянути застосування умовного випадкового поля в якості класифікатора. Ця модель добре зарекомендувала себе в галузі обробки природної мови.

Іншою популярної моделлю в NLP є довга короткочасна пам'ять (LSTM) [33]. Це є ще одним вдосконаленням RNN з використанням додаткових вентилів. LSTM – це ще один потенційний напрямок вдосконалення запропонованої моделі.

Також можна дослідити інші моделі уваги, що може вплинути на зміну ваги слів у кожному висловлюванні.

ВИСНОВКИ

У кваліфікаційній роботі було досліджено задачу класифікації у галузі NLP, що займається генерацією діалогів. Було визначено задачу такої класифікації та різні підходи до її розв’язання.

Основний фокус дослідження був спрямований на класифікацію діалогових актів. Під діалоговим актом можна розуміти певний намір або мету репліки у діалозі. Розпізнавання та опрацювання таких актів дає змогу значно покращити якість генерації відповідей у діалогових системах.

Було проведено аналіз більшості сучасних підходів до класифікації DA та запропоновано свій метод для реалізації поставленої задачі. Запропонований метод базується на рекурентних нейронних мережах та механізмі уваги. Його можна віднести до семантичних підходів, оскільки він працює лише з текстовим поданням діалогу та не бере до уваги просодичні ознаки.

Було розроблено та описано архітектуру моделі. Запропонована модель була реалізована засобами мови програмування Python та допоміжних бібліотек машинного навчання. Далі було проведено навчання моделі з використанням одного з найбільших розповсюджених для цієї задачі набору даних.

В результаті тестування модель показала досить великі показники точності класифікації. Отримані результати не перевищують поточний максимум, отриманий у інших дослідженнях, але все одно входять до топ 10 для обраного датасету.

Розроблена модель класифікації DA є новою, оскільки у роботі запропоновано її нову архітектуру, яка поєднує кращі підходи, що використовуються для вирішення інших завдань NLP.

До недоліків розробленої системи можна віднести досить довгий час навчання через складність моделі та велику кількість параметрів Ще одним недоліком є відсутність опрацювання просодичних ознак, що також може

вплинути на точність класифікації.

Також існує декілька шляхів подальшого розвитку проекту. До них можна віднести застосування інших методів класифікації на останньому шарі запропонованої мережі. Також можна розглянути інші механізми уваги для зміни важливості слів у репліці. Розмір контексту також може вплинути на точність класифікації. Та мабуть, найбільш перспективним є шлях об’єднання з іншими моделями NLP. Наприклад використання розпізнавання іменованих сутностей чи аналізу емоційного забарвлення разом з класифікацією.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Буданов М. Обработка текстов на естественных языках. https://habr.com/ru/company/vk/blog/358736/.

2. Learning phrase representations using RNN encoder-decoder for statistical machine translation / K. Cho et al. Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), Doha, Qatar. Stroudsburg, PA, USA, 2014. URL: https://doi.org/10.3115/v1/d14-1179.

3. Воробйов Є. К., Петров К. Е. Дослідження методів класифікації діалогових актів. Сучасні напрями розвитку інформаційно-комунікаційних технологій та засобів управління : Тези доп. дванадцатої міжнар. науково-техн. конф., м. Харків, 28 квіт. 2022 р. Харків, 2022. С. 14.

4. Data Mining, лекция 5: Задачи Data Mining. Классификация и кластеризация. URL: http://www.intuit.ru/department/database/datamining/5/datamining_5.html (дата звернення: 01.04.2022).

5. Core M. G., AlLen J. Coding dialogs with the DAMSL annotation scheme. AAAI fall symposium on communicative action in humans and machines. 1997. P. 28–35.

6. Jurafsky D., Shriberg E. Switchboard SWBD-DAMSL shallow-discourse-function annotation coders manual. 1997.

7. The ICSI meeting recorder dialog act (MRDA) corpus / E. Shriberg et al. Fort Belvoir, VA : Defense Technical Information Center, 2004. URL: https://doi.org/10.21236/ada460980.

8. Dialogue act annotation with the ISO 24617-2 standard / H. Bunt et al. Multimodal interaction with W3C standards. Cham, 2016. P. 109–135. URL: https://doi.org/10.1007/978-3-319-42816-1_.

9. Schmitt L. M., Nehaniv C. L., Fujii R. H. Linear analysis of genetic algorithms. Theoretical computer science. 1998. Vol. 200, no. 1-2. P. 101–134. URL: https://doi.org/10.1016/s0304-3975(98)00004-8.

10. Jeremy Ang, Yang Liu, Shriberg E. Automatic dialog act segmentation and classification in multiparty meetings. (ICASSP '05). IEEE international conference on acoustics, speech, and signal processing, 2005., Philadelphia, Pennsylvania, USA. URL: https://doi.org/10.1109/icassp.2005.1415300.

11. Dialogue act modeling for automatic tagging and recognition of conversational speech / A. Stolcke et al. Computational linguistics. 2000. Vol. 26, no. 3. P. 339–373. URL: https://doi.org/10.1162/089120100561737.

12. Dialogue act recognition via crf-attentive structured network / Z. Chen et al. SIGIR '18: The 41st International ACM SIGIR conference on research and development in Information Retrieval, Ann Arbor MI USA. New York, NY, USA, 2018. URL: https://doi.org/10.1145/3209978.3209997.

13. Berger J. O. Statistical decision theory and Bayesian analysis. 2nd ed. New York : Springer-Verlag, 1985. 617 p.

14. Dialogue act classification using a bayesian approach / S. Grau et al. 9th conference speech and computer, Saint-Petersburg, 20 September 2004. P. 495–499.

15. Automatic classification of dialog acts with Semantic Classification Trees and Polygrams / M. Mast et al. Connectionist, statistical and symbolic approaches to learning for natural language processing. Berlin, Heidelberg, 1996. P. 217–229. URL: https://doi.org/10.1007/3-540-60925-3_49.

16. Wright H. Automatic utterance type detection using suprasegmental features. Icslp’98, Sydney. 1998. P. 1403.

17. The DCIEM Map Task Corpus: spontaneous dialogue under sleep deprivation and drug treatment / E. G. Bard et al. Speech communication. 1996. Vol. 20, no. 1-2. P. 71–84. URL: https://doi.org/10.1016/s0167-6393(96)00045-3.

18. Ries K. HMM and neural network based speech act detection. 1999 IEEE international conference on acoustics, speech, and signal processing. proceedings. ICASSP99 (cat. no.99ch36258), Phoenix, AZ, USA, 19 March

1999. 1999. URL: https://doi.org/10.1109/icassp.1999.758171.

19. Gang Ji, Bilmes J. Dialog act tagging using graphical models. (ICASSP '05). IEEE international conference on acoustics, speech, and signal processing, 2005., Philadelphia, Pennsylvania, USA. URL: https://doi.org/10.1109/icassp.2005.1415043.

20. Jeong M., Lee G. Jointly predicting dialog act and named entity for spoken language understanding. 2006 IEEE spoken language technology workshop, Palm Beach, Aruba, 10–13 December 2006. 2006. URL: https://doi.org/10.1109/slt.2006.326818.

21. Rotaru M. Dialog act tagging using memory-based learning. 2002. P. 255–276.

22. Samuel K., Carberry S., Vijay-Shanker K. Dialogue act tagging with Transformation-Based Learning. The 17th international conference, Montreal, Quebec, Canada, 10–14 August 1998. Morristown, NJ, USA, 1998. URL: https://doi.org/10.3115/980432.980757.

23. Domain specific speech acts for spoken language translation / L. Levin et al. Proceedings of the fourth sigdial workshop of discourse and dialogue. 2003. P. 208–217.

24. Kohonen T. Self-Organizing maps. Berlin, Heidelberg : Springer Berlin Heidelberg, 1995. URL: https://doi.org/10.1007/978-3-642-97610-0.

25. Cottrell M., Fort J. C., Pagès G. Theoretical aspects of the SOM algorithm. Neurocomputing. 1998. Vol. 21, no. 1-3. P. 119–138. URL: https://doi.org/10.1016/s0925-2312(98)00034-4.

26. Andernach T., Poel M., Salomons E. Finding classes of dialogue utterances with kohonen networks. ECML/MLnet workshop on empirical learning of natural language processing tasks. 1997. P. 85–94.

27. On the properties of neural machine translation: encoder–decoder approaches / K. Cho et al. Proceedings of SSST-8, eighth workshop on syntax, semantics and structure in statistical translation, Doha, Qatar. Stroudsburg, PA, USA, 2014. URL: https://doi.org/10.3115/v1/w14-4012.

28. Hybrid computing using a neural network with dynamic external memory / A. Graves et al. Nature. 2016. Vol. 538, no. 7626. P. 471–476. URL: https://doi.org/10.1038/nature20101.

29. Bert: pre-training of deep bidirectional transformers for language understanding. / J. Devlin et al. 2018.

30. Roberta: a robustly optimized bert pretraining approach / Y. Liu et al. 2019. (Preprint. arXiv preprint arXiv:1907.11692).

31. Lee J. Y., Dernoncourt F. Sequential short-text classification with recurrent and convolutional neural networks. Proceedings of the 2016 conference of the north american chapter of the association for computational linguistics: human language technologies, San Diego, California. Stroudsburg, PA, USA, 2016. URL: https://doi.org/10.18653/v1/n16-1062

32. A context-based approach for dialogue act recognition using simple recurrent neural networks / C. Bothe et al. Lrec 2018. 2018.

33. Hochreiter S., Schmidhuber J. Long Short-Term Memory. Neural computation. 1997. Vol. 9, no. 8. P. 1735–1780. URL: https://doi.org/10.1162/neco.1997.9.8.1735.