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 нейронного машинного перекладу в реальному часі на прикладі листування багатонаціональної групи
(тема)

Виконав:
студент 2 курсу, групи СШІм-19-2
Волков Є. Д.
(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 доцент Чала Л. Е.
(посада, прізвище, ініціали)

Допускається до захисту

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

2021 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 Волкову Євгенію Дмитровичу
(прізвище, ім'я, по батькові)

1. Тема роботи Дослідження методів нейронного машинного перекладу в реальному часі на прикладі листування багатонаціональної групи

затверджена наказом університету від 29 березня 20 21 р. № 390Ст

2. Термін подання студентом роботи до екзаменаційної комісії _____ травня 20 21 р.

3. Вихідні дані до роботи Науково-технічні публікації, дані Інтернет-джерел та відомих наукових проектів щодо розробки та дослідження нейронного машинного перекладу корпусів текстів, Python documentation,

4. Перелік питань, що потрібно опрацювати в роботі Аналіз науково-технічної літератури з питань нейронного машинного перекладу, вивчення особливостей та проблем газузі нейронного машинного перекладу, аналіз існуючих рішень означених проблем, проведення експериментального моделювання та навчання моделі, розробка моделі потокового машинного перекладу для листування багатонаціональної групи

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
Основна частина	доц. Чала Л. Е.		

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Аналіз предметної галузі	03.04.2021	виконано
2	Методи та алгоритми машинного перекладу	10.04.2021	виконано
3	Експериментальне моделювання та навчання моделі	17.04.2021	виконано
4	Апробація моделі нейронного машинного перекладу	21.04.2021	виконано
5	Написання пояснювальної записки	24.04.2021	виконано
6	Попередній захист	14.05.2021	виконано
7	Захист перед ЕК		

Дата видачі завдання 29 березня 2021 р.

Студент _____
(підпис)

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

РЕФЕРАТ

Записка пояснювальна: 91 с., 8 рис., 5 табл., 3 дод., 46 джерел.

ЛИСТУВАННЯ, МАШИННИЙ ПЕРЕКЛАД, НЕЙРОННИЙ МАШИННИЙ ПЕРЕКЛАД, ПОЛІЛОГ, ПОТОКОВИЙ ПЕРЕКЛАД, ЧАТ

Об'єкт дослідження – методи оптимізації нейронного машинного перекладу.

Предмет дослідження – методи нейронного машинного перекладу потоків текстової інформації.

Мета роботи – дослідження методів нейронного машиного перекладу текстової інформації стандартного та потокового виду та створення моделі машинного перекладу здатної до перекладу групового листування багатонаціональної групи.

Методи дослідження – методи нейронного машинного перекладу, переклад з увагою, переклад розмов, методи збереження контексту підчас перекладу.

РЕФЕРАТ

Записка пояснительная: 91 с., 8 рис., 5 табл., 3 прил., 46 источников.

МАШИННЫЙ ПЕРЕВОД, НЕЙРОННЫЙ МАШИННЫЙ ПЕРЕВОД, ПЕРЕПИСКА, ПОЛИЛОГ, ПОТОКОВЫЙ ПЕРЕВОД, ЧАТ.

Объект исследования – методы оптимизации нейронного машинного перевода.

Предмет исследования – методы нейронного машинного перевода потоков текстовой информации.

Цель работы - исследование методов нейронного машинного перевода текстовой информации стандартного и потокового вида и создание модели машинного перевода способной к переводу групповой переписки многонациональной группы.

Методы исследования – методы нейронного машинного перевода, перевод с вниманием, перевод разговоров, методы сохранения контекста во время перевода.

ABSTRACT

Explanatory note: 91 p., 8 fig., 5 tabl., 3 ann., 46 sources.

CORRESPONDENCE, MACHINE TRANSLATION, NEURAL MACHINE TRANSLATION, POLYLOGUE, STREAM TRANSLATION, CHAT.

The object of the research – methods of optimization of neural machine translation.

The subject of the research – methods of neural machine translation of text streams.

The aim of the work – to study the methods of neural machine translation of standard and streaming text information and to create a machine translation model capable of translating joint correspondence of a multinational group.

Research methods – methods of neural machine translation, translation with attention, translation of conversations, methods of preserving context during translation.

ЗМІСТ

Перелік умовних позначень, символів, одиниць, скорочень та термінів 8

Вступ 9

1 Аналіз предметної галузі 10

1.1 Переклад текстів 10

1.2 Чати, мессенджери, групові переписки 14

1.3 Проблеми галузі МП 15

1.3.1 Переклад потоків текстових даних 15

1.3.2 Контекст у моделях машинного перекладу. Тематика 16

1.3.3 Мова як елемент моделі перекладу 17

1.4 Постановка задачі 18

2 Методи та алгоритми машинного перекладу 19

2.1 Підходи до створення моделей МП 19

2.1.1 Рекурентні нейронні мережі 19

2.1.2 Згорткові нейронні мережі 20

2.2 Вирішення проблеми збереження контексту 21

2.3 Вирішення проблеми перекладу потоків даних 24

3 Експериментальне моделювання та навчання моделі 28

3.1 Структура та склад моделі 28

3.2 Збереження контексту за допомогою кешу 29

3.2.1 Зчитування з кешу 30

3.2.2 Запис у кеш 33

3.2.3 Налаштування кешу 34

3.3 Алгоритм перекладу розмов на основі реплік 35

3.3.1 Історія на стороні джерела 36
3.3.2 Історія на стороні цілі 39
3.3.3 Подвійна історія розмови 40
3.3.4 Інтеграція історії у базову модель 41
3.3.5 Використання комбінованої уваги для декількох кодерів 43
3.4 Підготовка даних 46
3.4.1 Набори для навчання моделей лінійного перекладу 47
3.4.2 Набори даних для навчання розмовної моделі 49
4 Апробація моделі в реальних умовах 52
4.1 Базова структура моделі 52
4.2 Навчання моделі з кешем 55
4.3 Навчання моделі обробки декількох кодерів 55
4.4 Навчання моделі перекладу розмов 56
4.5 Результати навчання моделі 57
Висновки 61
Перелік джерел посилання 62
Додаток А Приклади навчальних даних 69
Додаток Б Лістинги елементів моделі 85
Додаток В Відомість кваліфікаційної роботи 91

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ ТА ТЕРМІНІВ

МП – машинний переклад;

НМП – нейронний машинний переклад;

СМП – статистичний машинний переклад;

Bi-MSMT – Bilingual Multi-Speaker Machine Translation – двомовний машинний переклад для декількох співрозмовників;

CNN – Convolutional Neural Network – згорткова нейронна мережа;

GLU – Gated Linear Unit – лінійний елемент з обмеженням;

GRU – Gated Recurrent Unit – рекурентний елемент з обмеженням;

IWSLT – International Conference on Spoken Language Translation – міжнародна конференція з перекладу розмовної мови;

LDC – Linguistic Data Consortium – Консорціум з лінгвістичних даних;

LSTM – Long Short-Term Memory – довга короткострокова пам’ять;

NLP – Natural Language Processing – обробка природних мов;

RNN – Recurrent Neural Network – рекурентна нейронна мережа;

WMT – Conference on Machine Translation – конференція з машинного перекладу.

ВСТУП

Обчислювальна лінгвістика – це міждисциплінарна галузь, що займається обчислювальним моделюванням природної мови, а також вивченням відповідних обчислювальних підходів до лінгвістичних питань. Машинний переклад є однією з галузей машинного навчання, що активно набирають темпи розвитку. Ця галузь спрямована на створення моделей, здатних до автоматичного перекладу природномовних текстів з однієї мови на іншу.

Поточні моделі машинного перекладу часто налаштовуються за доменом або професією (наприклад, звіти про погоду), покращуючи результати за допомогою обмеження допустимих перекладів. Цей прийом ефективний у сферах, де використовується формальна або формульна мова. Машинний переклад державних та юридичних текстів легше дає корисний результат, ніж переклад розмов або менш стандартизованих текстів. Основним напрямом перекладу до недавніх пір був переклад текстів фіксованого розміру, зазвичай певних документів. Такий підхід є задовільним для спеціалізованих комерційних продуктів, які не спрямовані на швидкий переклад значних об’ємів інформації, проте в умовах споживчого використання швидкість обробки та допустимі об’єми мають бути вище.

Однією з потенційних форм споживчого використання може бути переклад чатів у соціальних мережах, де багато користувачів можуть одночасно вести декілька тематичних листувань. В подібних умовах у методів, спрямованих на обробку фіксованих об’ємів текстів, можуть виникати значні проблеми. Ця кваліфікаційна робота направлена на пошук, інтеграцію та розробку моделі машинного перекладу, здатної частково їх подолати.

1 АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ

1.1 Переклад текстів

Машинний переклад (МП) – підгалузь обчислювальної лінгвістики, яка досліджує використання програмного забезпечення для перекладу тексту або мовлення з однієї мови на іншу.

На базовому рівні МП виконує механічну заміну слів однією мовою словами іншою. Переклади, отримані за допомогою таких моделей зазвичай мають низьку якість, оскільки для якісного перекладу необхідне розпізнавання цілих фраз та їх найближчих аналогів цільовою мовою. Крім того, не всі слова однієї мови мають еквівалентні слова іншою мовою, і багато слів мають більше одного значення.

Витоки машинного перекладу можна знайти в роботах Аль-Кінді, арабського криптографа 9-го століття, який розробив методи системного перекладу мов, включаючи крипто аналіз, частотний аналіз, а також ймовірність і статистику, які використовуються в сучасному машинному перекладі. Пізніше, у 1629 році Рене Декарт запропонував універсальну мову, у якій рівнозначні ідеї з різних мов мають спільний символ.

Ідея перекладу природних мов за допомогою цифрових комп’ютерів була одночасно запропонована англійськими вченими А. Д. Бутом та Уорреном Вівером з Фонду Рокфеллера ще в 1946 році. Останнім у 1949 році було написано меморандум, який деякі історики галузі вважають однією з найвпливовіших публікацій в перші дні машинного перекладу. У 1954 році на машині APEXC у коледжі Біркбек (Лондонський університет) було проведено демонстрацію елементарного перекладу з англійської мови на французьку. Аналогічний додаток, також вперше запущений у коледжі Біркбек, зчитував та складав тексти Брайля. В той же час було опубліковано кілька наукових робіт на цю тему і навіть статей в популярних журналах.

Період активного розвитку галузі тривав до шістдесятих років, коли

дослідження машинного перекладу почали приносити малі результати, а ціна та тривалість дослідних проектів значно зросла. Новий виток розвитку розпочався в другій половини сімдесятих років, з появою комерційного інтересу у машинному перекладі, а також зі збільшенням обчислювальної потужності комп’ютерів. В центрі уваги був статистичний машинний переклад. Ця динаміка тривала до початку двадцять першого століття.

Ранні спроби використання нейронних мереж у МП мали на меті замінити n-грамові мовні моделі на нейромережеві [1, 2]. Пізніше нейронні мережі зворотного зв'язку використовувались для вдосконалення систем, що базуються на фразах, шляхом повторної оцінки ймовірності перекладу фраз [3]. Можливість використовувати вхідні дані змінної довжини досягалась за допомогою рекурентних нейронних мереж (RNN), які пропонували принциповий спосіб представлення послідовностей завдяки прихованим станам. Одну з перших «безперервних» моделей, тобто тих, що не використовують явно збережені перекладені фрази, запропонували Кальчбреннер та Блунсом [4], з RNN для моделювання цільової мови та моделі вихідного речення на основі згорткової мережі (або n-грамової). Щоб вирішити проблему зникаючого градієнта у RNN, в моделях послідовність-до-послідовності використовувались одиниці довгої короткочасної пам’яті (LSTM) [5], які в подальшому були спрощені до закритих рекурентних одиниць (GRU) [6, 7]. Такі блоки дозволяли мережам фіксувати довгострокові залежності між словами завдяки спеціалізованим блокам, що дозволяють запам'ятовувати та забувати попередні вхідні дані.

Такі моделі послідовність-до-послідовності застосовувались до МП з кодером та RNN- декодером [6], але мали серйозні труднощі у відображенні довгих речень як єдиного вектору [8], хоча використання двонаправлених RNN та об'єднання їх відображень для кожного слова могло б частково вирішити це обмеження. Однак ключовим нововведенням був механізм уваги, запроваджений [9], що дозволяє декодеру на кожному кроці вибирати, яку частину вхідного речення корисніше врахувати для

передбачення наступного слова. Увага – контекстний вектор – зважена сума за всіма прихованими станами кодера – який можна розглядати як модель вирівнювання між вхідними та вихідними положеннями. Ефективність моделі була додатково покращена, що мало незначний вплив на якість перекладу [10, 11]. Пропозицію розрізнювати моделі локальної та глобальної уваги з роботи [10] ще не включили в основні моделі.

Демонстрація того, що НМП з орієнтованою на увагу кодером-декодером RNN перевершила СМП на основі фраз, відбулася в завданні перекладу новин 2016 року в оцінюванні WMT [12]. Система, представлена Единбурзьким університетом у роботі [13], отримала найвищий рейтинг, зокрема завдяки двом додатковим вдосконаленням загальної моделі. Перший з них полягав у використанні зворотного перекладу одномовних цільових даних із ультрасучасного механізму СМП на основі фраз, щоб збільшити кількість паралельних даних, доступних для навчання [14]. Другий – використання кодування пар байтів, дозволяючи трансляцію n-грам символів і, таким чином, долаючи обмежений словниковий запас кодування та вбудованих декодерів [15]. Було показано, що мовні позначки низького рівня приносять невеликі додаткові переваги у якості перекладу [16]. Незабаром Единбурзька система була видана у відкритий доступ під назвою Nematus [17].

Дослідницькі та комерційні системи МП швидко прийняли НМП, починаючи з пар мов, які мали найбільшу кількість надійних перекладів, таких як англійська та інші європейські мови та китайська. Приблизно в кінці 2016 року онлайн МП, запропонований Bing, DeepL, Google або Systran, забезпечувався все глибшими RNN (інформація з глибини мереж недоступна). У випадку з DeepL, хоча інформації про системи опубліковано небагато, її видима якість може бути частково пояснена використанням високоякісних паралельних даних Linguee.

RNN з увагою дозволяють досягти максимальної продуктивності, але за ціною великих обчислювальних витрат. Наприклад, найбільша система

Google НМП з 2016 року [18], з її 8-ма рівнями кодера та декодера по 1024 вузла LSTM кожен, вимагала навчання на 96 графічних процесорах nVidia K80 протягом 6 днів, незважаючи на масове розпаралелювання (наприклад, запуску кожного шару на окремому графічному процесорі). Більш перспективним підходом до зменшення обчислювальної складності є використання згорткових нейронних мереж для моделювання послідовностей до послідовностей, як пропонується [19] у моделі ConvS2S від Facebook AI Research. Ця модель перевершила систему By та співавторів [18]. у перекладах WMT 2014 EN / DE та EN / FR «на порядок швидше, як на графічному, так і на центральному процесорі». Опублікована в травні 2017 року модель була перевершена наступного місяця Transformer [20].

Модель Transformer НМП [20] усуває послідовні залежності (повторюваність) у кодері та декодері, а також потребу в згортаннях, і використовує мережі з увагою до себе для позиційного кодування. Наприклад, кодер складається з шести пар 512-мірних шарів; у кожній парі перший шар реалізує увагу до себе на багатьох заголовках, тоді як другий – повністю з'єднаний шар прямого руху. У декодері додатковий шар у кожній парі реалізує увагу з декількома заголовками через вихідні дані кодера. Як результат, навчання на графічних процесорах може бути повністю розпаралелено, що істотно скорочує час навчання і трохи перевершує моделі RNN.

З цих причин Transformer був швидко прийнятий дослідницьким співтовариством: його використовували практично всі системи для перекладу новин WMT 2018 [21]. Зараз модель реалізована в більшості наборів інструментів НМП. Хоча на момент написання статті Transformer залишається найсучаснішим, декілька його авторів продемонстрували, що, використовуючи деякі ідеї з попередньої моделі, архітектури RNN можна вдосконалити щоб досягти ефективності, яка перевершує результати Transformer, і що гібридні архітектури, засновані на RNN, CNN та

Transformer підвищують оцінки на наборах даних WMT'14 EN / DE та EN / FR ще вище [22]. В кінці 2018 року була представлена модель більш глибокої уваги для МП, яка фільтрувала увагу від нижчих до вищих рівнів у понад п’яти шарах [23], із обнадійливими результатами.

1.2 Чати, мессенджери, групові переписки

Мессенджер (англ. Messenger – посланець, instant messaging – обмін миттєвими повідомленнями) – це програма для миттєвого обміну повідомленнями, яка встановлюється на комп'ютер, ноутбук, смартфон або планшет. Дозволяє обмінюватися між користувачами текстовими, голосовими повідомленнями, а також прикріплювати фото, відео, різні документи. Перші версії месенджерів могли обмінюватися тільки текстовими повідомленнями. Сучасні системи, крім перерахованих вище функцій, дозволяють здійснювати аудіо та відео дзвінки, створювати групові чати, відеоконференції, відправляти автоматичні повідомлення, зберігати дані в «хмарі» і багато іншого.

Найбільш цікавою в контексті МП є функція групового чату, яка дозволяє обмінюватись текстовими повідомленнями з багатьма людьми одночасно. В групових чатах присутні ті ж можливості, що й в персональних, проте кількість співбесідників зростає, а за рахунок цього і кількість можливих бесід, при цьому кожна бесіда може мати будь-яку кількість учасників. Таким чином виникають досить специфічні умови для виконання перекладу, оскільки для забезпечення якості необхідно подолати ряд проблем, з якими й досі не до кінця впорались в галузі.

1.3 Проблеми галузі МП

1.3.1 Переклад потоків текстових даних

Потік текстових даних – безперервна, часто швидка, впорядкована послідовність текстів; текстова інформація, що надходить безперервно з часом у вигляді потоку даних. Приклади включають в себе, але не обмежуються:

– новини та аналогічні регулярні звіти: статті новин, онлайн-коментарі до них, звіти про трафік, внутрішні звіти компаній, результати веб-пошуку, наукові статті, патенти;

– соціальні медіа: форуми для обговорень (наприклад, Twitter, Facebook), короткі повідомлення на телефонах або комп’ютері, чат, стенограми телефонних розмов, блоги, електронні листи.

Проблема перекладу потоків текстових даних з’являється у випадках, коли необхідно обробляти масиви текстових даних, які не мають остаточного закінчення. Стандартний машинний переклад спрямований на опрацювання кінцевих блоків даних у вигляді речень, абзаців, або документів. Вважається, що для перекладу цих блоків вся необхідна інформація знаходиться або безпосередньо всередині блоків, або в моделі, натренованій їх перекладати. Переклад потоків даних в цьому випадку не гарантує, що вся необхідна для перекладу інформація буде присутня на тому чи іншому етапі.

Прикладами потокових даних, які потребують перекладу, можна вважати діа- та полілог, в яких кожна наступна репліка може мати непередбачуваний контекст, а також пряму мову безпосередньо, оскільки під час прямої мови модель-перекладач не здатна заздалегідь передбачити, що саме скаже людина.

1.3.2 Контекст у моделях машинного перекладу. Тематика

Поняття контексту в галузі машинного перекладу пов’язано з двома основними факторами: тематикою документу та специфікою природної мови.

Тематика задає глобальний контекст перекладу. Вона задає обмеження на значення слів та словосполучень, а також вимагає використання специфічних термінів, яких може не бути в навчальному корпусі документів. Тематика потребує від моделі можливості забувати та запам’ятовувати контекст.

Одна з ключових проблем більшості моделей машинного перекладу – втрата контексту оброблюваних даних в процесі перекладу. Вона впливає на якість отриманих перекладів і має більш значний вплив при обробці великих блоків текстових даних, таких як багатосторінкові документи. Особливо гостро ця проблема стає за умови перекладу потоку пов’язаних один з одним контекстом реплік, що сильно впливає на якість отриманих результатів. Процес втрати відбувається за рахунок того, що обробка великих масивів тексту відбувається послідовно протягом великої кількості ітерацій. RNN здатні забезпечити збереження певних частин контексту протягом декількох ітерацій, проте більш близькі за часом ітерації матимуть більший вплив на переклад. Це призводить до того, що елементи, вжиті на початку перекладу, в кінці більше не впливають на процес, що призводить до відхилень у сприйнятті значень термінів.

Проблема зміни контексту в потоці даних характерна для розмов, коли учасники можуть у будь-який момент перейти від однієї теми до іншої. Модель повинна мати змогу швидко адаптуватись до зміни контексту. В контексті перекладу повідомлень у груповому чаті ця проблема особливо загострюється, оскільки підчас групової комунікації різні співбесідники можуть відповідати на різні репліки інших учасників, формуючи таким чином потік невпорядковано пов’язаних одне з одним блоків даних. Трохи

поліпшити визначення контексту може широко розповсюджена у мессенджерах функція «відповісти на повідомлення», яка дозволяє поєднати дві, або більше репліки у ланцюжок бесіди, з якого буде простіше виділити контекст.

1.3.3 Мова як елемент моделі перекладу

Специфіка природної мови виражена наявністю у мові слів зі спорідненим написанням, але різним значенням, використанням різних форм займенників для зменшення повторень згадок певних об’єктів. Певні мови також мають більш складні випадки, коли глобальний контекст визначає окремий сенс кожного окремого слова. Окрім цього, кожна мова має певний набор замінників, які можуть використовуватись заміть тих чи інших об’єктів у тексті. Інколи, використання замінників може відбуватись протягом декількох послідовних частин документу. Це накладає на модель обмеження у вигляді необхідності виділяти та пов’язувати між собою об’єкти та замінники, які для них використовуються.

Окремо також можна висунути проблему омонімів – слів, які пишуться однаково, проте мають відмінні значення. Основна складність для алгоритму перекладу полягає в необхідності за допомогою зовнішнього контексту визначити, який сенс має омонім. Так, наприклад, для української мови характерна наявність як омографів («брати́» і «бра́ти»), так і омоформ («руда» копалина і «руда» колір), в яких сенс слова можна вловити з речення, тоді як для японської ситуація складніше, оскільки один і той же ієрогліф може мати кілька прочитань, але також і кілька ієрогліфів можуть мати однакове прочитання. На рисунку 1.1 представлений приклад подібної ситуації – кандзі для слова «гарячий». Слово має один й той самий вимов «атсуі» та написання у фонетичній формі, проте, залежно від контексту (хвороба, а бо клімат), використовується інший ієрогліф. У подібному випадку складніше буде визначити значення слова, якщо воно

написано у фонетичній формі, а не у ієрогліфічній.

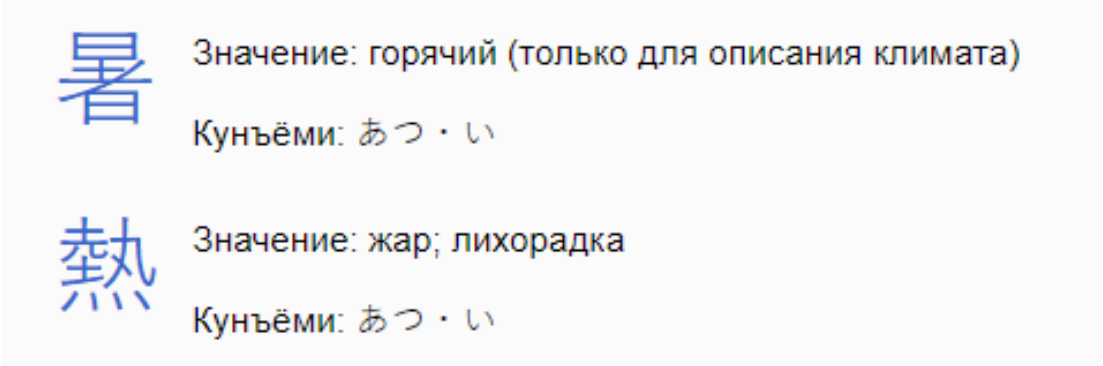

Рисунок 1.1 – Кандзі для слова «гарячий»

1.4 Постановка задачі

Таким чином, мета даної кваліфікаційної роботи – аналіз існуючих методів машинного перекладу для потоків текстових даних, а також методів вирішення проблем, пов’язаних з контекстом перекладу, та створення з їх допомогою моделі, здатної виконувати переклад багатомовного групового чату користувачів.

Згідно з метою роботи були поставлені наступні задачі:

– аналіз існуючих наукових джерел за темою дослідження;

– дослідження методів та етапів машинного перекладу;

– розробка моделі МП, здатної виконувати переклад багатомовного групового чату користувачів;

– навчання та апробація моделі.

2 МЕТОДИ ТА АЛГОРИТМИ МАШИННОГО ПЕРЕКЛАДУ

2.1 Підходи до створення моделей МП

2.1.1 Рекурентні нейронні мережі

В джерелах [24 – 28] розглядаються рішення, засновані на рекурентних нейронних мережах (RNN), які є однією з найбільш популярних основ для моделей МП. Особливість даного типу мереж полягає в здатності попередніх обчислень впливати на поточне за допомогою операції затримки сигналу. Це дозволяє під час обробки фраз або речень сформувати впорядковану залежність між усіма їх елементами, що призводить до підвищення якості перекладу.

В рамках НМП структура кодера-декодера є досить популярною архітектурою RNN. Ця архітектура складається з двох компонентів: мережі кодера, яка споживає вхідний текст, та мережі декодера, яка генерує перекладений вихідний текст. Завдання кодера – витягти щільне представлення фіксованого розміру вхідних текстів різної довжини. Завдання декодера – генерувати відповідний текст призначеною мовою на основі щільного представлення від кодера (рисунок 2.1). Зазвичай обидві мережі є RNN, часто LSTM.

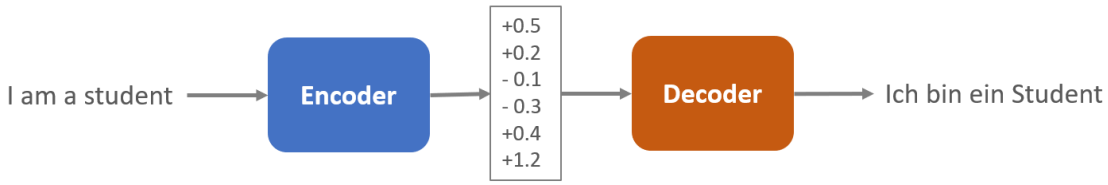

Рисунок 2.1 – Нейронна архітектура «кодер-декодер»

Типовим способом представлення RNN є розгортання їх у послідовність копій тієї самої статичної мережі A, кожна з яких подається прихованим станом попередньої копії $h_{(t-1)}$ та поточним входом x (t). Потім розгорнуту RNN можна навчити за допомогою алгоритму зворотного поширення у часі (BPTT) (рисунок 2.2).

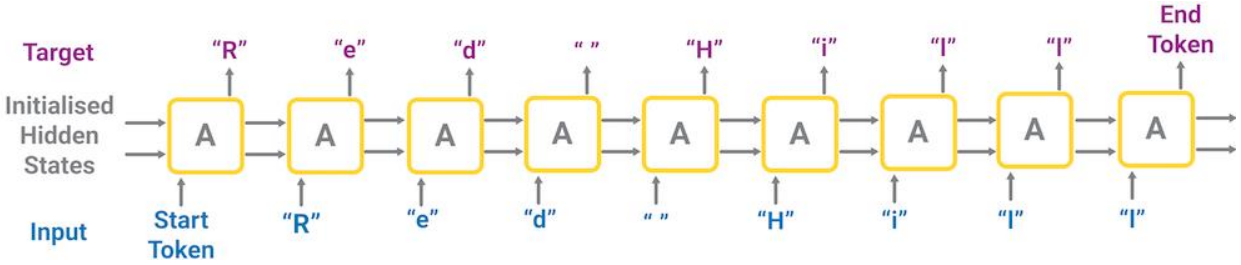

Рисунок 2.2 – LSTM RNN мережа, представлена у вигляді послідовності копій статичної мережі

2.1.2 Згорткові нейронні мережі

В джерелах [29 – 31] розглядаються рішення, засновані на згорткових нейронних мережах, які характеризуються спільною структурою кодера і декодера, яка складається з блоків та обчислює проміжні стани на основі фіксованої кількості вхідних елементів. Вихід l-го блоку позначається як $h^l = (h_1^l, \dots, h_n^l)$ для мережі декодера, та $z^l = (z_1^l, \dots, h_m^l)$ для мережі кодера. Кожен блок містить одновимірну згортку з наступною нелінійністю. Для мережі декодера з одним блоком і шириною ядра k кожен результуючий стан h_i^l і містить інформацію про k елементів введення. Складання декількох блоків один на одного збільшує кількість вхідних елементів, представлених у стані. Наприклад, складання 6 блоків з k = 5 призводить до поля введення з 25 елементів, тобто кожен вихід залежить від 25 входів. Нелінійність дозволяє мережам використовувати повне поле введення або, якщо потрібно, зосередитись на меншій кількості елементів.

Кожне ядро згортки забезпечується параметрами $W \in \mathbb{R}^{2d \times kd}$, $b_w \in \mathbb{R}^{2d \times kd}$ і приймає вхідні дані у форматі $X \in \mathbb{R}^{k \times d}$, що є конкатенацією k вхідних елементів, вбудованих у d виміри, і відображає їх в один вихідний елемент $Y \in \mathbb{R}^{2d}$, який має подвійну розмірність вхідних елементів; наступні шари працюють над k вихідними елементами попереднього шару. Ми обираємо лінійні елементи з обмеженням (GLU, з роботи [32]) як нелінійну функцію, які реалізують простий механізм затвора над виходом згортки $Y = [A\ B] \in \mathbb{R}^{2d}$:

$$v([A\ B]) = A \otimes \sigma(B), \tag{2.1}$$

де $v([A\ B]) \in \mathbb{R}^{2d}$ – результат функції, вдвічі менший за розмір Y;

$A, B \in \mathbb{R}^{d}$ – вхідні дані до нелінійної функції;

$\otimes$ – точкове множення.

Затвори контролюють, які вхідні значення A поточного контексту є релевантними.

2.2 Вирішення проблеми збереження контексту

Відображення символу у формі безперервного вектору кодує декілька вимірів подібності, що є еквівалентним кодуванню декількох значень слова. Отже, НМП потрібно витратити значну частину своєї потужності на однозначне визначення початкових та цільових слів на основі контексту, що визначається реченням-джерелом [33]. Послідовність – ще одна важлива проблема перекладу на рівні документа, де повторюваний термін повинен зберігати однаковий переклад у всьому документі [34, 35]. Тим не менше, нинішні моделі НМП все ще обробляють документи, перекладаючи кожне речення окремо, страждаючи від непослідовності та неоднозначності, що виникають через використання одного речення-джерела. Ці проблеми важко

усунути, використовуючи лише обмежений контекст окремих речень. Вчені й досі працюють над різними інструментами для подолання проблеми контексту.

Перехресний контекст між реченнями, або глобальний контекст, виявився корисним для кращого відображення значення або наміру в послідовних завданнях, таких як рекомендація запитів [36] та моделювання діалогу [37, 38]. Однак переваги глобального контексту для НМП отримали порівняно мало уваги дослідницького співтовариства. У роботі [26] автори пропонують модель НМП, яка враховує перехресний контекст речень, що враховує вплив попередніх речень-джерел у тому самому документі.

Зокрема, вони використовують ієрархію RNN, щоб узагальнити перехресний контекст речень з попередніх речень джерела, який використовує додаткову RNN на рівні всього документа разом з RNN на рівні окремих речень [36]. Для інтеграції глобального контексту в переклад речень розроблені наступні стратегії:

– ініціалізація – використовує представлення історії як початковий стан кодера, декодера або обох;

– допоміжний контекст – використовує представлення історії як статичний перехресний контекст речень, що працює разом із динамічним внутрішнім контекстом речення, створеним моделлю з увагою;

– обмежуючий допоміжний контекст – визначає обсяг глобального контексту, що використовується для генерації наступного цільового слова на кожному кроці декодування.

Окрім збереження контексту між реченнями та фразами, проблема контексту також пов’язана зі значеннями слів. Як зазначено у розділі 1.3.2, на якість перекладу також можуть впливати омоніми і займенники, які здатні приховувати сенс слова. Існує ряд методів, пов’язаних як з моделлю безпосередньо, так і з вхідними даними.

Як приклад роботи з вхідними даними для навчання, можна розглянути роботу [29]. В ній пропонується два способи розширення

моделей НМП, здатних покращити переклад:

– підхід, заснований на явній вказівці сенсу слова в тексті за допомогою анотацій;

– підхід, заснований на лексичних ланцюжках, які в стислому вигляді формують необхідний для перекладу документа контекст.

Перший підхід спирається на сенсові вкладення для анотування. Сенсові вкладення – це векторні представлення сенсів слів у векторному просторі, але на відміну від вкладень слів, де кожна форма слова отримує векторне представлення, за допомогою сенсових вкладень ми отримуємо окремі векторні подання для кожного сенсу даного слова. Для обчислення сенсових вкладень використовується метод SenseGram [39]. Після того, як сенсові вкладення вивчені, всі змістові слова в даних позначаються відповідними сенсами, а інформація про сенси включається як додаткові властивості.

SenseGram дозволяє розрізнити слово на основі контексту, в якому воно зустрічається. На основі слів з визначеним сенсом стає можливим виявити лексичні ланцюги, тобто ланцюжки семантично подібних слів у документі. Для обчислення семантичної подібності між двома значеннями слів автори обчислюють подібність косинусів між їх сенсовими вкладеннями. Чим ближче до 1.0 отримане значення, тим вище їх семантична подібність. Щоб розрізнити подібні та не схожі сенси був встановлений поріг 0,85, який автори роботи обирали вручну, дивлячись на те, як різні значення впливають на отримані лексичні ланцюги: нижчий поріг створює лексичні ланцюги, що містять чуттєві слова, які недостатньо пов’язані між собою, тоді як вищий поріг призводить до семантично сильних, але, можливо, неповних лексичних ланцюжків, які не охоплюють усіх слів, що належать до ланцюга. Засновуючись на оцінці подібності сенсів, алгоритм вибудовує сенсові ланцюжки для окремих документів, проводячи створення, розширення та, якщо необхідно, об’єднання ланцюжків.

2.3 Вирішення проблеми перекладу потоків даних

Як було зазначено у розділі 1.3.1, існує суттєва різниця між перекладом текстових документів та потоку текстів. У випадку перекладу документів можна зробити припущення, що вся необхідна інформація є обмеженою та знаходиться всередині тексту, або безпосередньо в моделі. Переклад потоків тексту передбачає, що текстові дані можуть не мати повного контексту для перекладу, а інформація є потенційно нескінченною. В цьому розділі розглянуті декілька рішень, які були розроблені для покращення перекладу.

Розглянемо рішення, надане в роботі [24], спрямоване на оптимізацію перекладу розмов у чатах на різних мовах. В рамках задачі перекладу використовується набір даних, який містить паралельні розмови, і кожна розмова складається з черг. Кожен хід розмови складається з речень, виголошених одним мовцем, позначених x або y, якщо речення відповідно англійською або іноземною мовою. Мета полягає в тому, щоб підготувати модель, яка може використовувати змішану історію розмов для створення високоякісних перекладів.

Автор виділяє три типи контексту, з якими можна зіткнутися в поточній двомовній розмові з кількома ораторами (рисунок 2.3). Він включає:

- попередньо завершені англійські черги;
- раніше завершені іноземні черги;
- поточну чергу (англійська чи іноземна).

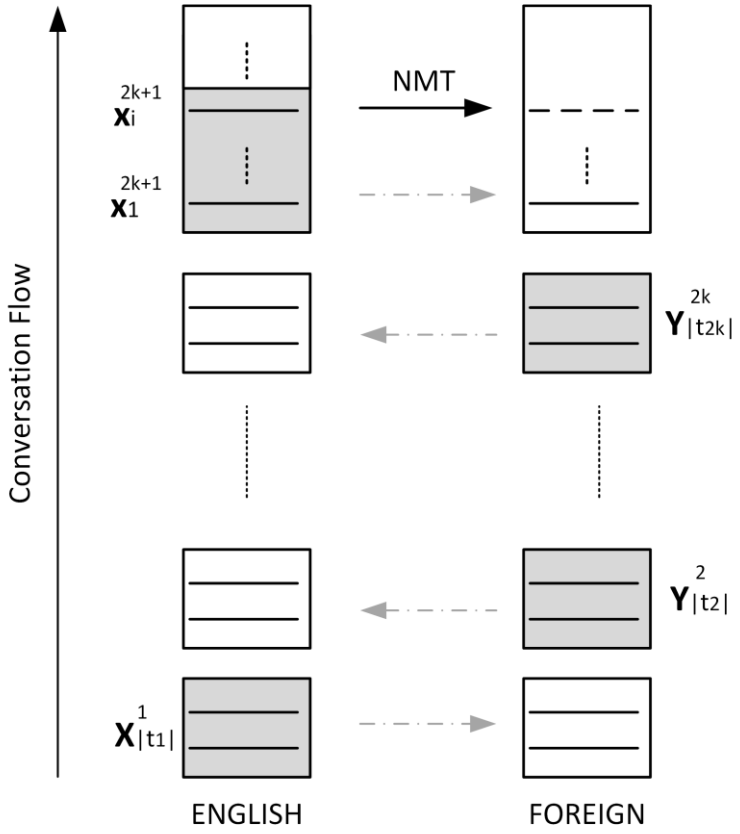

Рисунок 2.3 – Огляд тривалої розмови під час перекладу і-го речення у 2k + 1-й хід

Автор пропонує розмовну модель Bi-MSMT, яка може включити всі три типи контексту, використовуючи історію розмови джерела, цілі або і те, і інше, у базову модель. Базова модель забезпечує перехід мови мовця, маючи в основі речення модель НМП (описану раніше) для кожного напрямку перекладу, з англійської на іноземну та навпаки.

Проблема збереження інформації для подальшого перекладу розглядається в роботі [25].

Запропонований підхід доповнює моделі НМП за допомогою кеш-пам'яті, яка виявилась корисною для збору довшої історії при моделюванні мов [40, 41]. Кеш-пам'ять (кеш) є, по суті, пам'яттю «ключ-значення» [42], яка являє собою масив слотів у формі пар ключ-значення. Етап

встановлення відповідності базується на записах ключів, тоді як етап читання використовує записи значень.

Оскільки сучасні моделі НМП генерують переклад у форматі від слова до слова, інформація про переклад, як правило, зберігається на рівні слів, включаючи вхідний контекст, що містить контент, який треба перекласти, та цільовий контекст, який відповідає сформованому слову. Щоб кеш мав змогу запам'ятовувати історію перекладів, ключ повинен бути розроблений з властивостями, які допоможуть узгодити його з контекстом джерела, тоді як значення має бути розроблено з властивостями, які допоможуть узгодити його з цільовим контекстом. З цією метою слоти кешу визначаються як пари векторів $\{(c_1; s_1), \dots, (c_i; s_i), \dots, (c_I; s_I)\}$, де c_i і s_i – вектор контексту уваги та відповідний йому стан декодера на і-тому етапі з попередніх перекладів. Два типи векторів представлення добре відповідають контексту джерела та цільової сторони [43].

Рисунок 2.4 ілюструє архітектуру моделі з роботи [25].

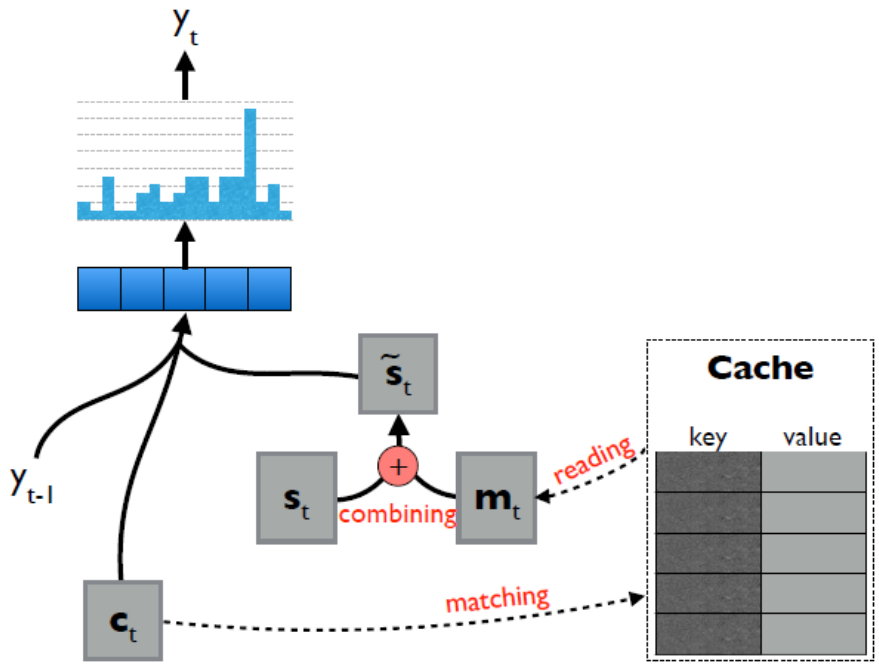

Рисунок 2.4 – Архітектура моделі з кешем

На кожному кроці декодування t поточний контекст уваги c_t служить запитом, який використовується для зіставлення та зчитування з кешу в пошуках відповідної інформації для формування цільового слова. Отриманий вектор m_t, який містить цільові контексти генерування подібних слів в історії перекладу, поєднується з поточним станом декодера s_t, щоб згодом утворити цільове слово y_t. Коли генерується повний переклад, контексти декодування зберігаються в кеші як історія майбутніх перекладів.

Таким чином, у розділі 2 розглянуто найпоширеніші сучасні методи та алгоритми машинного перекладу. Серед розглянутих методів були обрані три, які будуть використані для створення комбінованої моделі:

- модель лінійного перекладу з кеш-пам’яттю з роботи [25];
- модель поєднання контекстів декількох кодерів;
- модель, здатна до перекладу речень з роботи [24].

3 ЕКСПЕРИМЕНТАЛЬНЕ МОДЕЛЮВАННЯ ТА НАВЧАННЯ МОДЕЛІ

3.1 Структура та склад моделі

В рамках даної кваліфікаційної роботи запропонована базова модель, яка складається з двох архітектур НМП на основі послідовностей [9], по одній для кожного напряму перекладу. Кожна з них містить кодер для читання вихідного речення та декодеру з увагою для генерації перекладу по одному елементу за раз.

Кодер відображає кожне вихідне слово x_m у розподілене представлення h_m, яке є об'єднанням відповідних прихованих станів двох RNN, що працюють у протилежних напрямках над вихідним реченням [46]. У цій роботі в якості прямих та зворотних RNN виступають GRU [6].

Генерування кожного цільового слова y_n зумовлене всіма раніше створеними словами $y_{<n}$ через стан s_n декодера, та вихідне речення через динамічний вектор контексту c_n:

$$y_n \sim softmax\left(W_y \cdot u_n + b_y\right), \tag{3.1}$$

$$u_n = \tanh(s_n + W_{uc} \cdot c_n + W_{un} \cdot E_T[y_{n-1}]), \tag{3.2}$$

$$s_n = GRU(s_{n-1}, E_T[y_{n-1}], c_n), \tag{3.3}$$

$$c_n = \sum_m \alpha_{nm} h_m, \tag{3.4}$$

де s_n – стан декодера;

c_n – динамічний вектор контексту;

$y_{<n}$ – попередньо перекладені слова;

$W_{(\cdot)}$, b_y – навчальні параметри;

$E_T[y_{n-1}]$ – вбудовування попереднього цільового слова y_{n-1};

h_m – розподілене представлення вхідного слова;

α_{nm} – пропорціональний показник користі інформації з вхідного речення.

Описана в рівняннях (3.1 – 3.4) базова модель розширюється за допомогою методів, описаних у подальших підрозділах.

3.2 Збереження контексту за допомогою кешу

Оскільки переклад тексту відбуватиметься в умовах багатостороннього листування, важливо, щоб модель мала доступ до найближчих відомих контекстів розмов. Механізм, запропонований у роботі [25], надає змогу зберігати для декодера кеш з нещодавно перекладених термінів. Це, потенційно, може допомогти моделі з подоланням проблеми займенників.

Запропонований у роботі підхід доповнює моделі НМП за допомогою кеш-пам'яті, що виявилося корисним для збору довшої історії для завдання моделювання мови [40, 41]. Кеш-пам'ять є, по суті, пам'яттю-словником [42], яка являє собою масив слотів у формі пар (ключ, значення). Етап встановлення відповідності базується на ключах слотів, тоді як етап зчитування використовує їх значення. В подальшому для позначення кеш-пам'яті використовуватиметься позначення «кеш».

Оскільки сучасні моделі НМП генерують переклад від слова до слова, інформація про переклад, як правило, зберігається на рівні слів, включаючи контекст джерела, що містить текст, який перекладається, і цільовий контекст, який відповідає сформованому слову. З метою запам’ятати історію перекладів, ключ повинен мати властивості, які допоможуть узгодити його з контекстом джерела, тоді як значення має мати властивості, які допоможуть узгодити його з цільовим контекстом. З цією метою слоти кеш-пам'яті визначаються як пари векторів $\{(c_1, s_1), ..., (c_i, s_i), ..., (c_I, s_I)\}$, де

c_i та s_i – вектор контексту уваги та відповідний стан декодера на етапі часу і з попередніх перекладів. Два типи векторів відображення добре відповідають контексту джерела та цільової сторони.

Огляд структури моделі з кешем та візуальне зображення базового принципу дії наводиться у розділі 2.3 (рисунок 2.4). Процес взаємодії моделі з кешем можна розділити на три частини:

– запис в кеш;
– навчання кешу;
– зчитування з кешу.

3.2.1 Зчитування з кешу

Зчитування з кешу складається з трьох етапів:

– знаходження кешу;
– зчитування значення;
– модифікація стану декодера.

Метою пошуку ключа є отримання подібних записів у кеші. З цією метою використовується контекст уваги c_t для визначення розподілу ймовірностей між записами в кеші. Використовуючи представлення контексту як ключів у кеші, оператор пошуку кешу може бути реалізований за допомогою простої операції softmax для добутку векторів ключів і контексту:

$$P_m(c_i|c_t) = \frac{\exp(c_t^{\mathrm{I}} c_i)}{\sum_{i'=1}^{I} \exp(c_t^{\mathrm{I}} c_i')}, \tag{3.5}$$

де c_t – контекст уваги на поточному кроці t;

c_i – збережений контекст в і-тому слоті кешу;

I – кількість слотів у кеші.

На відміну від існуючих доповнених пам’яттю нейронних мереж, запропонований кеш уникає необхідності вивчати параметри для пошуку в пам’яті, такі як параметричні моделі уваги [41], перетворення між запитом та ключами [42], [25], або визначені людиною скалярних значень для контролю рівності розподілу [40].

Значення кешу зчитуються шляхом взяття суми над збереженими значеннями, зваженими за імовірностями збігу з ключем, генеруючи вектор m_t:

$$m_t = \sum_{(c_i,s_i)\in cache} P_m(c_i|c_t)s_i, \tag{3.6}$$

де c_t – контекст уваги на поточному кроці t;

c_i – збережений контекст в і-тому слоті кешу, ключ слоту;

s_i – збережений стан декодеру в і-тому слоті кешу, значення слоту;

$P_m(c_i|c_t)$ – ймовірність збігу контексту з ключем.

З точки зору доповнених пам’яттю нейронних мереж, ймовірність відповідності $P_m(c_i|c_t)$ може бути інтерпретована як ймовірність отримання подібної цільової інформації m_t з кешу з урахуванням контексту джерела c_t, де бажаною відповіддю є контексти, пов’язані з подібними цільовими словами, створеними в минулих перекладах.

Остаточний стан декодера, який використовується для генерації розподілу наступного слова, обчислюється з лінійної комбінації вихідного стану декодера st та вихідного вектору m_t, отриманого з кешу:

$$\tilde{s}_t = (1 - \lambda_t)\otimes s_t + \lambda_t \otimes m_t, \tag{3.7}$$

де $\tilde{s}_t$ – оновлений стан декодера;

s_t – поточний стан декодера;

m_t – згенерований вектор стану з кешу;

$\otimes$ – операція по-елементного множення;

$\lambda_t \in \mathbb{R}^d$ – динамічний вектор коефіцієнтів, що вираховується для кроку t.

Ця стратегія заснована на концепції воріт, що оновлюються, з GRU [6], яка бере лінійну суму між попереднім прихованим станом та можливим новим прихованим станом. Початковою точкою для цієї стратегії є спостереження: генерування цільових слів на різних етапах має різні потреби в історії перекладу. Наприклад, відображення історії перекладів є більш корисним, якщо подібний слот знаходиться в кеш-пам'яті, тоді як інші – в інших випадках. З цією метою динамічний вектор коефіцієнтів обчислюється за наступною формулою:

$$\lambda_t = \sigma(Us_t + Vc_t + Wm_t), \tag{3.8}$$

де σ(·) – функція логістичної сигмоїди;

$\{U \in \mathbb{R}^{d\times d}, V \in \mathbb{R}^{d\times l}, W \in \mathbb{R}^{d\times d}\}$ – матриці параметрів;

d, l – кількість параметрів стану декодера та контекстного вектору відповідно.

Параметр λ_t має однакову розмірність, як s_t і m_t, тому кожен елемент у двох векторах має різну вагу інтерполяції. Таким чином можна отримати більш точний контроль для поєднання представлень, оскільки різні елементи зберігають різну інформацію.

Використовуючи модифікований стан декодера, отримання вектору перекладу виконується за формулою:

$$P(y_t|y_{<t}, x) = g(y_{t-1}, c_t, \tilde{s}_t), \tag{3.9}$$

де $P(y_t|y_{<t}, x)$ – ймовірність для наступного слова в залежності від попереднього перекладеного слова та інформації з кодеру;

y_{t-1}– попередній переклад;

c_t – поточний контекст уваги;

$\tilde{s}_t$ – модифікований стан декодеру;

$g(\cdot)$ – операція GRU.

Додавання безперервного кешу до моделі НМП успадковує переваги схожих на кеш форм пам’яті: розподіл ймовірностей за згенерованими словами оновлюється в реальному часі залежно від історії перекладу, і послідовні переклади можуть бути сформовані, якщо вони були знайдені в історії. Нейронний кеш також успадковує здатність прихованих станів декодера моделювати довготривалі контексти перехресних речень замість внутрішнього контексту речення, що дозволяє більш точно моделювати контекст на рівні документа.

3.2.2 Запис у кеш

Компонент кешу – це зовнішня структура пам’яті ключ-значення, яка зберігає I елементів нещодавніх історій, де ключ у позиції $i \in [1, M]$ – k_i, а його значення – v_i. Для кожної пари ключ-значення також зберігається відповідне цільове слово y_t як індикатор для наступного оператора оновлення.

Робота [25] зосереджується на навчанні запам’ятовувати та використовувати історію перекладів з перехресних речень. Відповідно, на відміну від роботи [40], де кеш оновлюється після кожної генерації цільового слова, в даній моделі запис в кеш відбувається після повної генерації перекладу речення. Враховуючи, що сформований переклад речення $y = \{y_1, ..., y_t, ..., y_T\}$, його відповідна векторна послідовність контекстів уваги $\{c_1, ..., c_t, ..., c_T\}$, а послідовність станів декодера – $\{s_1, ..., s_t, ..., s_T\}$. Кожний триплет $\langle c_t, s_t, y_t \rangle$ записується в кеш наступним чином:

– якщо y_t не існує в кеш-пам’яті, вибирається порожній слот або перезаписується найбільш пізніше використаний слот, де ключем є c_t, значенням – s_t, а індикатором – y_t;

– якщо y_t вже існує в кеші в і-тому слоті, ключ та значення

оновлюються: $k_i = \frac{k_i + c_t}{2}$ та $v_i = \frac{v_i + s_t}{2}$.

З точки зору «загальної політики кешування», це можна розглядати як свого роду експоненціальний спад, оскільки при кожному оновленні попередні ключі та значення зменшуються вдвічі. З іншого боку, з точки зору безперервного кешування, інтуїція, що лежить в основі цього, полягає в моделюванні часового порядку одного й того самого слова – новіші історії виступають у якості більш важливих ролей.

3.2.3 Налаштування кешу

Дві стратегії проходження були визначені корисними для полегшення труднощів у навчанні, коли модель порівняно складна [24, 26]. Кеш додається до попередньо навченої моделі НМП з підготовкою лише нових параметрів, безпосередньо пов'язаних з ним.

В першу чергу, попередньо навчається стандартна модель НМП, яка здатна генерувати достатньо якісні переклади (c_t та s_t) для взаємодії з кешем. Формально параметри θ моделі навчаються, щоб максимізувати ймовірність набору навчальних прикладів $\{[x^n, y^n]\}_{n=1}^N$:

$$\hat{\theta} = \arg\max_{\theta} \sum_{n=1}^{N} \log P(y^n | x^n; \theta), \tag{3.10}$$

де $\hat{\theta}$ – приблизна оцінка параметрів θ;

$P(y^n | x^n; \theta)$ – оцінка ймовірності перекладу y^n для репліки x^n.

Підчас стандартного навчання моделі оцінка ймовірності визначається за наступною формулою:

$$P(y_t | y_{<t}, x) = g(y_{t-1}, c_t, s_t), \tag{3.11}$$

де y_{t-1} – попереднє слово;

c_t – поточний контекст уваги;

s_t – стан декодеру;

$g(\cdot)$ – операція GRU.

На наступному етапі, треновані параметри $\hat{\theta}$ фіксуються і тренуються нові параметри γ = {U, V, W}, пов'язані з кешем:

$$\hat{\gamma} = \arg\max_{\gamma} \sum_{n=1}^{N} \log P\left(y^n \middle| x^n; \hat{\theta}, \gamma\right), \quad (3.12)$$

де $\hat{\gamma}$ – оцінка параметрів кешу;

$\hat{\theta}$ – приблизна оцінка параметрів θ;

$P\left(y^n \middle| x^n; \hat{\theta}, \gamma\right)$ – оцінка ймовірності перекладу y^n для репліки x^n з використанням кешу, заснована на формулі (3.9).

Під час навчання вектори c_t та s_t залишаються однаковими для даної пари речень із фіксованими параметрами НМП, таким чином кеш можна явно навчити, коли використовувати історію перекладів, щоб максимізувати загальну продуктивність перекладу.

3.3 Алгоритм перекладу розмов на основі реплік

В основі кваліфікаційної роботи стоїть задача перекладу листувань між двома, або більше учасниками. З цього виходить, що контекст для перекладу поточної репліки може знаходитись у попередніх репліках на мовах, відмінних від поточного джерела. Робота [24] надає потенційний механізм для ефективного вилучення контексту незалежно від аспекту мови, що й послужило приводом для її інтеграції в поточну модель.

Для поліпшення сприйняття інформації, автори при описі методів в якості мови джерела та цільової мови використовують англійську та французьку відповідно. Ці ж мови використовуватимуться для опису в рамках даної кваліфікаційної роботи.

В рамках роботи автори визначають три типи контексту, з якими можливо зіткнутися в тривалій двомовній розмові з кількома ораторами. Вони включають:

– попередні репліки англійською мовою;
– попередні репліки іноземною мовою;
– поточна репліка (англійська чи іноземна).

В роботі [24] розглядається розмовна модель Bi-MSMT, яка може поєднати всі три типи контексту в базовій моделі, використовуючи історію розмов на стороні джерела, цілі або подвійну. База моделі забезпечує перехід на мову мовця, маючи на основі речення модель НМП (описану в розділі 2.3) для кожного напрямку перекладу, з англійської на іноземну та навпаки.

3.3.1 Історія на стороні джерела

Якщо припустити, що відбувається переклад поточної розмови з чергуванням англійської та іноземної мов. Поточною є 2k + 1 репліка (англійською мовою) і необхідно перекласти її і-те речення, використовуючи історію розмов на стороні джерела, представлену контекстним вектором o_{src} (розмірності H).

В умовах, коли представлення попередніх вхідних речень у розмові є відомими, представлення поточних вхідних речень передаються через Turn-RNN, які являють собою двонаправлені RNN, що є залежними від мови, але не від спікера, як показано на рисунку 3.1. Останні приховані стани прямої та зворотної RNN об'єднуються, щоб отримати остаточне представлення репліки r_j, де j позначає індекс репліки. Представлення окремих реплік об'єднуються, засновуючись на поточній мові, для отримання векторів контексту o_{en} та o_{fr}, обчислюваних кількома можливими способами, які надалі об'єднуються за допомогою механізму обмеження, щоб надати різну важливість кожному елементу контекстного вектору:

$$o_{en,fr} = \alpha \odot o_{en} + (1 - \alpha) \odot o_{fr}, \quad (3.13)$$

де o_{en}, o_{fr} – вектори контексту для англійської та французької мов;

α – коефіцієнт співвідношення, який вираховується за наступною формулою:

$$\alpha = \sigma(U_{en} \times o_{en} + U_{fr} \times o_{fr} + b_g), \quad (3.14)$$

де $\sigma(\cdot)$ – логістична сигмоіда;

U_{en}, U_{fr} – матриці параметрів;

b_g – вектор зсуву.

Для остаточного отримання контексту джерела виконується процедура зменшення розмірності:

$$o_{src} = \tanh(W_T \times o_{en,fr} + b_T), \quad (3.15)$$

де b_T – вектор зсуву;

W_T – вагова матриця;

$o_{en,fr}$ – контексти мов, отримані з (3.13).

Параметри W, U, b – це специфічні для мови змінні параметри. В роботі [24] автори пропонують п’ять способів обчислення конкретних мовних представлень контексту, o_{en} та o_{fr}. В рамках даної кваліфікаційної роботи буде розглянуто метод на основі прямого перетворення.

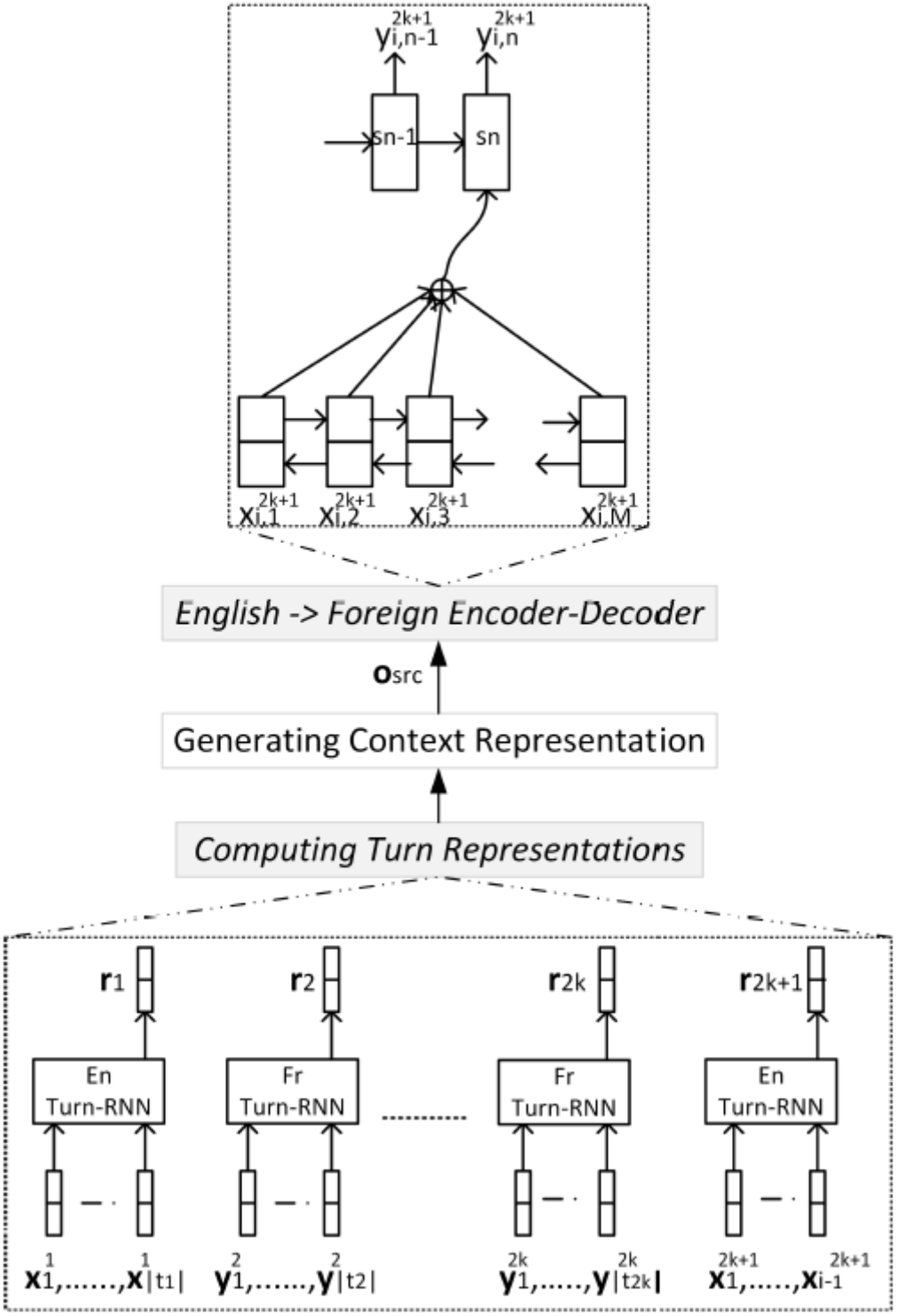

Рисунок 3.1 – Архітектурний огляд перекладу і-го речення у 2k + 1-й репліці з використанням історії джерела

Пряме перетворення, найпростіший підхід, полягає в поєднанні представлень реплік за допомогою специфічного для мови перетворення зі зменшенням розмірності:

$$o_{en} = \tanh([W_{en}; \dots; W_{en}] \times [r_1; \dots; r_{2k+1}] + b_{en}), \quad (3.16)$$

$$o_{fr} = \tanh\big([W_{fr}; \dots; W_{fr}] \times [r_2; \dots; r_{2k}] + b_{fr}\big), \quad (3.17)$$

де W_l – матриці для зміни розмірності для мов;

r_i – вагові матриці, об'єднуються по рядам;

b_l – вектори зсуву для мов.

3.3.2 Історія на стороні цілі

Використання історії розмов на стороні цілі настільки ж важливе, як і історії розмов на стороні джерела, оскільки це допомагає зробити переклад більш відповідним на цільовій мові. Це стає важливим для перекладу бесід, де попередні репліки були однією мовою. Історія цільової сторони реалізується за допомогою уваги на рівні речення, подібної до тієї, що описана для історії на стороні джерела. Для всіх попередніх англійських вихідних речень формується матриця R_{en}, що складається з відповідних репрезентацій цільового речення в іноземній мові та інша матриця R_{fr} представлення цільових речень (англійською мовою) для попередніх іноземних реплік. Кожне представлення цільового речення має розмірність H. Тоді, обчислення контексту репліки відбувається наступним чином:

$$p_{en} = softmax(R_{en}^T \times \tanh(W_{t,en} \times h_i + b_{t,en})), \quad (3.18)$$

$$p_{fr} = softmax(R_{fr}^{T} \times \tanh(W_{td,en} \times h_i + b_{td,en})), \quad (3.19)$$

$$o_{en} = R_{en} \times p_{en}, \quad (3.20)$$

$$o_{fr} = \tanh\left(W_{t,en} \times \left(R_{fr} \times p_{fr}\right) + b_{t,en}\right), \quad (3.21)$$

де W – матриця ваг;

R – матриця репрезентацій цільового речення;

h – вектор запиту;

b – вектор зсуву.

Параметри $W_{t,en}$, $b_{t,en}$ призначені для зменшення розмірності та зміни мовного простору вектору запиту h_i та вектору контексту, тоді як $\{W_{td,en}, b_{td,en}\}$ призначені лише для зменшення розмірності. o_{en} та o_{fr} додатково поєднуються за допомогою механізму обмеження, як у рівнянні. (3.13), щоб отримати кінцевий цільовий вектор контексту o_{tgt} (розмірності H).

3.3.3 Подвійна історія розмови

Найпростіший спосіб використовувати одночасно історію розмови на стороні джерела та цілі – включити обидва контекстні вектори o_{src} та o_{tgt} в базову модель, яка називається подвійним контекстом Src-Tgt.

Інший інтуїтивний підхід, як видно з рисунка 3.1, полягає в окремому моделюванні англійських та іноземних речень за допомогою двох окремих контекстних векторів $o_{en,m}$ та $o_{fr,m}$, кожен з них побудований із суміші оригінальних джерел або цільових перекладів, залежить від мови і, можливо, містять менше шуму. Автори [24] називають це подвійним контекстом Src-Tgt-Mix. Вони роблять припущення, згідно з яким, $R_{en,m}$ містить змішані представлення джерела та цілі для англійської

мови (розміри для представлень джерела зменшені до H), а $R_{fr,m}$ містить ті ж дані для іноземної мови. Тоді розрахунок комбінованого контексту відбувається за наступними формулами:

$$p_{en,m} = softmax(R_{en,m}^{T} \times \tanh(W_{td,en} \times h_i + b_{td,en})), \quad (3.18)$$

$$p_{fr,m} = softmax(R_{fr,m}^{T} \times \tanh(W_{tt,en} \times h_i + b_{tt,en})), \quad (3.19)$$

$$o_{en,m} = \tanh\left(W_{tr,en} \times \left(R_{en,m} \times p_{en,m}\right) + b_{tr,en}\right), \quad (3.20)$$

$$o_{fr,m} = R_{fr,m} \times p_{fr,m}, \quad (3.21)$$

де $W_{td,en}$, $W_{tr,en}$ та $W_{tt,en}$ – матриці, призначені для зменшення розмірності

3.3.4 Інтеграція історії у базову модель

Остаточні представлення o_{src} та o_{tgt} або $o_{en,m}$ та $o_{fr,m}$, можуть бути включені разом або окремо в базову модель за допомогою наступних методів:

- InitDec;
- AddDec;
- InitDec+AddDec – комбінації двох попередніх методів.

Метод InitDec заснований на використанні нелінійного перетворення для ініціалізації декодера, подібно до [26]:

$$s_{i,0} = \tanh(V \times o_i + b_s) \quad (3.22)$$

де i – індекс речення в поточній репліці 2k + 1;

V, b_s – специфічні параметри для пари кодера-декодера;

o_i – окремий вектор контексту історії, або об'єднання (перетворене) двох контекстів.

AddDec – це метод, в якому контексти використовуються як допоміжний вхід в декодер (подібний до [24], [26]):

$$s_{i,n} = \tanh(W_s \cdot s_{i,n-1} + W_{sn} \cdot E_t[y_{i,n}] + \\ + W_{sc} \cdot c_{i,n} + W_{ss} \cdot o_{i,src} + W_{st} \cdot o_{i,tgt}), \quad (3.23)$$

де W_i – матриці ваг;

s_a – стан декодеру;

c_b – вектор контексту уваги;

o_n – вектори контексту історії.

Параметри моделі навчаються наскрізь, шляхом максимізації суми логарифму ймовірності двомовних розмов у навчальному наборі D. Наприклад, для розмови з чергуванням англійської та іноземної мов функція втрат, реалізована за допомогою ймовірностей перекладу, складатиме:

$$\sum_{k=0}^{\frac{|T|}{2}-1} \left(\sum_{i=1}^{|t_{2k+1}|} logP_\theta(y_i|x_i, o_i) + \sum_{j=1}^{|t_{2k+2}|} logP_\theta(y_j|x_j, o_j) \right), \quad (3.24)$$

де i, j – індекси речень, що належать до 2k + 1-ї або 2k + 2-ї репліки;

o_i – представлення контексту історії;

|T| – загальна кількість реплік (вважається парною).

Найкраща вихідна послідовність для даної вхідної послідовності для і-го речення під час тестування, тобто декодування, отримується за допомогою наступної формули:

$$arg \max_{y_i} P_\theta(y_i|x_i, o_i), \quad (3.25)$$

3.3.5 Використання комбінованої уваги для декількох кодерів

В рамках кваліфікаційної роботи реалізовано модель, здатну виконувати переклад тексту у чаті з декількома учасниками. Для цього буде доцільним, якщо декодер в моделі буде здатним обробляти запити з кодера будь-якою іншою мовою. Саме така можливість розглядається в рамках роботи [44].

Механізм уваги при навчанні послідовність-до-послідовності (S2S) дозволяє декодеру на основі RNN отримувати безпосередній доступ до інформації про вхідні дані кожного разу, коли він генерує символ. Механізм уваги, що базується на адресації на основі вмісту, що був розроблений для нейронних машин Тьюринга, оцінює розподіл ймовірності між прихованими станами кодера на кожному кроці декодування. Цей розподіл використовується для обчислення контекстного вектору – зваженого середнього значення прихованих станів кодера – як додатковий вхід до декодера.

Стандартна модель уваги, описана у роботі [9] визначає енергії уваги e_{ij}, розподіл уваги α_{ij} та вектор контексту c_i на і-му кроці декодера як:

$$c_i = \sum_{j=1}^{T_x} \alpha_{ij} h_j \tag{3.26}$$

де c_i – контекстний вектор для і-го кроку;

h_j – стан кодера на j-му кроці;

α_{ij} – розподіл уваги, вираховується наступною формулою:

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^{T_x} \exp(e_{ik})} \tag{3.27}$$

де T_x – довжина вхідної послідовності

e_{ij} – енергія уваги, що вираховується наступною формулою:

$$e_{ij} = v_a^{\mathrm{T}} \tanh\left(W_a s_i + U_a h_j\right) \tag{3.28}$$

де s_i – стан декодера на і-му кроці;

h_j – стан кодера на j-му кроці;

v_a – зважений вектор розмірностей для простору обчислень;

W_a, U_a – параметри, що піддаються навчанню, проекційні матриці для перетворення станів кодера та декодера у загальний векторний простір

Нещодавно Лю в роботі [43] представив сторожовий шлюз, розширення RNN декодера з увагою та з блоками LSTM [5]. Розглянутий метод адаптує це розширення для GRU [6]:

$$\psi_i = \sigma(W_y y_i + W_s s_{i-1}) \tag{3.29}$$

де W_y, W_s – параметри, що піддаються навчанню;

y_i – вправлений вхідний сигнал до декодера;

s_{i-1} – попередній стан декодера.

Аналогічно рівнянню (3.28), скалярний енергетичний доданок для сторожової одиниці обчислюється за наступною формулою:

$$e_{\psi_i} = v_a^{\mathrm{T}} \tanh\left(W_a s_i + U_a^{(\psi)}(\psi_i \odot s_i)\right) \tag{3.30}$$

де W_a, $U_a^{(\psi)}$ – проекційні матриці;

v_a^{T} – зважений вектор розмірностей для простору обчислень;

$\psi_i \odot s_i$ – вектор сторожової одиниці.

Варто зауважити, що вектор сторожової одиниці (ВСО) не залежить від будь-якого прихованого стану будь-якого кодера. ВСО проектується на той самий векторний простір, що і стан кодера $\mathrm{h_j}$ у рівнянні (3.28). Параметр e_{ψ_i} додається як додатковий елемент енергії уваги до рівняння (3.27), а

вектор сторожової одиниці використовується як відповідний вектор у підсумовуванні в рівнянні (3.26).

Цей прийом повинен дозволити декодеру обирати, чи слідкувати за станом кодера, або зосередитись на власному стані та діяти більше як мовна модель. Це може бути корисно, якщо кодер не містить багато необхідної інформації для поточного кроку декодування.

Додатково, в моделях S2S з кількома кодерами декодер повинен мати можливість поєднувати інформацію про увагу, зібрану з кодерів.

Широко прийнятою методикою поєднання моделей множинної уваги в декодері є об'єднання контекстних векторів $c_i^1, \dots, c_i^N$. Цей підхід змушує модель приділяти увагу кожному кодеру незалежно і дозволяє неявно проводити комбінацію уваги на наступних рівнях мережі.

В роботі [44] автори пропонують дві альтернативні стратегії поєднання уваги від декількох кодерів. Вони або дозволяють декодеру вивчити розподіл c_i спільно по всіх прихованих станах кодера (плоске комбінування уваги), або розкласти на фактори розподіл по окремих кодерах (ієрархічне комбінування).

Обидва варіанти дозволяють явно обчислити розподіл по кодерам і, таким чином, інтерпретувати, скільки уваги приділяти кожному кодеру на кожному кроці декодування. В рамках даної роботи був обраний метод плоского комбінування уваги.

Плоске комбінування уваги проектує приховані стани всіх кодерів у спільний простір, а потім обчислює довільний розподіл за проекціями. Різниця між конкатенацією контекстних векторів та плоским комбінуванням уваги полягає в тому, що коефіцієнти α_i обчислюються спільно для всіх кодерів:

$$\alpha_{ij}^{(k)} = \frac{\exp(e_{ij}^{(k)})}{\sum_{n=1}^{N} \sum_{m=1}^{T_x^{(n)}} \exp(e_{im}^{(n)})} \tag{3.31}$$

де $T_x^{(n)}$ – довжина вхідної послідовності n-го кодера;

$e_{ij}^{(k)}$ – енергія уваги j-го стану k-го кодера на і-му етапі декодування.

Ці енергії уваги обчислюються, як у рівнянні (3.28). Параметри v_a та W_a розподіляються між кодерами, а U_a є різним для кожного кодера і служить специфічною для кодера проекцією прихованих станів у загальний векторний простір.

Стани окремих кодерів займають різні векторні простори і можуть мати різну розмірність, тому вектор контексту не можна обчислити як їх зважену суму. В рамках розглянутого методу вони проектуються в єдиний простір за допомогою лінійних проекцій:

$$c_i = \sum_{k=1}^{N} \sum_{j=1}^{T_x^{(k)}} \alpha_{ij}^{(k)} U_c^{(k)} h_j^{(k)} \tag{3.32}$$

де $U_c^{(k)}$ – додаткові навчальні параметри.

Матриці $U_c^{(k)}$ проектують приховані стани у спільний векторний простір. Це порушує питання, чи може цей простір бути таким самим, як той, на який проектується енергетичний розрахунок за допомогою матриць $U_a^{(k)}$ у рівнянні (3.28), тобто чи $U_c^{(k)} = U_a^{(k)}$. У своїх експериментах автори роботи досліджують обидва варіанти. Вони також досліджують аспект додавання та не додавання до контекстного вектору ВСО $\alpha_i^{(\psi)} U_c^{(\psi)} \psi_i \odot s_i$. В поточній роботі використаний підхід, згідно з яким матриці проекції різняться.

3.4 Підготовка даних

Зазначені в розділі 3.1 методи покращення перекладу можна поділити на дві групи: лінійні та розмовні. До першої групи відносяться методи оптимізації перекладу на основі кешу та комбінування уваги з різних

кодерів. Ці методи спрямовані на поліпшення простого перекладу, коли контекст є нерозривною складовою оброблюваного тексту і виражений на одній мові. До другої групи відноситься метод перекладу полілогів, в основі якого лежить принцип, що контекст витягується з послідовних реплік на різних мовах. Тому, для підготовки різних типів моделей потрібні різні набори даних, які будуть розглянуть в подальших підрозділах. Приклади даних, що містяться в наборах можна знайти у додатку А.

Оскільки всі моделі є частиною однієї системи, навчання проводилось на наборах зі спільними мовами. В рамках поточної роботи розглядаються англійська, німецька та французька мови, оскільки вони є одними з найбільш поширених у задачах МП.

3.4.1 Набори для навчання моделей лінійного перекладу

Автори роботи [25] проводили китайсько-англійські перекладацькі експерименти в кількох доменах, кожен з яких відрізняється від інших темою, жанром, стилем, рівнем офіційності тощо. Для цього вони використовували корпуси з трьох основних джерел:

- новини;
- субтитри;
- TED.

Домен новин витягується з корпусу LDC. Більшість речень у цьому корпусі – це офіційні статті із синтаксичними структурами, такими як складні сполучникові фрази, що дуже ускладнює переклад тексту. Нажаль, корпус знаходиться в закритому доступі і отримати з нього дані нема можливості.

Субтитри витягуються із телевізійних серіалів, та, як правило, є простими та короткими. Більшість перекладів субтитрів взагалі не зберігають синтаксичних структур своїх початкових речень. Автори роботи,

нажаль, використовують виключно англійську та китайську мову, що виключає це джерело зі списку можливих.

TED корпуси взяті з завдання для МП на основі TED доповідей з IWSLT 2015 [45]. Дослідники зазначають, що системи НМП мають більш круту криву навчання порівняно з обсягом навчальних даних, що призводить до погіршення якості в умовах малої кількості ресурсів. Доповіді на TED важко перекласти через різноманітність тем та невеликий об’єм навчальних даних. Варто зазначити, що кожне змагання має обмежений набір мов. Тому, для поточної роботи обрано більш близький за часом набір даних з IWSLT 2017, групи C [45], оскільки він має мовні сети англійською, німецькою та французькою мовами. В якості набору налаштувань обрано набір даних «dev2010», а в якості тестового – комбінацію наборів даних «tst2010– 2013».

Оригінальні корпуси IWSLT 2017, розміщені на ресурсі [45] у відкритому доступі, були оброблені для спрощення процесу навчання моделі. З цією метою був написаний скрипт на мові Scala, представлений у лістингу Б.1.

Оригінальне форматування наборів даних IWSLT 2017 та їх трансформована скриптом з лістингу Б.1 версія наведені в лістингах А.1-2 та таблиці А.1 відповідно.

Статистика набору наведена в таблиці 3.1.

Таблиця 3.1 – Статистика з навчальних наборів даних для кешу

Область	Кількість речень	Кількість слів		
		Німецька	Англійська	Французька
IWSLT 2017				
Навчання	210 тис.	4.1 млн.	4.4 млн.	4.1 млн.
Налаштування	887	21.3 тис.	17.5 тис.	21.3 тис.
Тестування	5.5 тис.	104.1 тис.	92.2 тис.	104.1 тис.

3.4.2 Набори даних для навчання розмовної моделі

В стандартному вигляді, завдання перекладу розмов значно відрізняється від класичного завдання МП. Надано набір даних, який містить паралельні розмови, і кожна розмова складається з реплік. Кожна репліка складається з речень, виголошених одним мовцем, позначених x або y, якщо речення відповідно англійською або іноземною мовою. Мета полягає в тому, щоб вивчити модель, яка здатна використовувати змішану історію розмов для створення високоякісних перекладів.

Стандартні набори даних МП неприйнятні для завдання Bi-MSMT, оскільки вони не складаються з бесід або відсутні анотації динаміків. У цьому розділі буде описано, яким чином отримуються дані із вихідних корпусів Europarl v7 та OpenSubtitles 2016 для цього завдання.

3.4.2.1 Набір даних Europarl

Неопрацьований корпус Europarl v7 містить теги SPEAKER та LANGUAGE, де останній вказує мову, якою насправді користувався мовець. Окремі файли спочатку поділяються на розмови. Дані токенізуються (за допомогою сценаріїв) та очищаються (видаляються заголовки та одинарні речення). Бесіди діляться на менші, якщо кількість доповідачів перевищує 5. Потім корпус випадковим чином поділяється на набори для тренування, розробки та тестування у співвідношенні 100 : 2 : 3. Англійська сторона корпусу встановлена як основна, і якщо мовний тег відсутній, мовою джерела є англійська, інакше іноземна. Речення на вихідній стороні корпусу зберігаються або замінюються речами на цільовій стороні на основі цього тегу.

Вищезазначені кроки виконуються для англійсько-французької, англійсько-естонської та англійсько-німецької мов, о дозволяє отримати двомовні корпуси з багатьма мовами для трьох мовних пар. Перш ніж

розподілятись на набори для тренування, розробки і тестування, речення видаляються з набору, якщо мають більше 100 лексем для англійсько-французької, англійсько-німецької та більше 80 лексем для англійсько-естонської відповідно, щоб обмежити довжину речення для використання підслів. Статистика даних наведена в таблиці 3.2. Приклади даних з наборів можна знайти у додатках.

Оригінальні дані Europarl v7, розміщені авторами [25] у відкритий доступ, були оброблені для спрощення процесу навчання моделі. З цією метою був написаний скрипт на мові Scala, представлений у лістингу Б.2.

Оригінальне форматування наборів даних Europarl v7 та їх трансформована скриптом з лістингу Б.2 версія наведені в лістингах А.3-4 та таблиці А.2 відповідно.

Таблиця 3.2 – Статистика з навчального набору даних

	Europarl		
	En-Fr	En-Et	En-De
Кільк. розмов	6997	4394	3582
Кільк. речень	246540	174218	109241
Середня статистика по розмовах			
Кільк. речень	36,24	40,65	31,50
Кільк. реплік	4,77	4,85	4,79
Розмір репліки	7,12	7,92	6,16

3.4.2.2 Набор даних з субтитрів

Для створення цього набору даних була проведена робота з отримання відміток ораторів шляхом автоматичної сегментації реплік для корпусу OpenSubtitles 2016. Англійська сторона корпусу наявна у відкритому доступі. Щоб отримати паралельний корпус, використовується посилання

для зіставлення OpenSubtitles для зіставлення іноземних субтитрів до розмічених англійських. Для кожного субтитру витягуються індивідуальні бесіди з більш ніж 5 реченнями та принаймні двома репліками. Розмови з більш ніж 30 репліками відкидаються. Нарешті, оскільки субтитри написані однією мовою, мовний тег призначається таким чином, щоб одна і та ж мова йшла по черзі через одну репліку.

Таким чином в рамках оригінальної роботи [24] було сформовано корпус Bi-MSMT для англійської та російської мов, який потім був поділений на набори для тренування, розробки та тестування. Нажаль, підготовлений набір не входить до переліку мов та мовних пар, які нам необхідні, а формування необхідних наборів стає складніше через необхідність відповідного зіставлення для трьох різних мов, тому в рамках поточної роботи це джерело не буде використано.

Таким чином, в кваліфікаційні роботі були розглянуті різні набори даних, придатних для навчання моделі. Обрано …..

Загалом, в кваліфікаційній роботі були розглянуті різні набори даних, придатні для навчання моделі. Для поточної моделі були обрані наступні набори:

– набір для лінійного навчання IWSLT 2017 з наступними парами мов: англійська-французька, англійська-німецька,німецька-французька;

– набір для навчання розмовам Europarl v7 з наступними парами мов: англійська-французька, англійська-німецька.

4 АПРОБАЦІЯ МОДЕЛІ В РЕАЛЬНИХ УМОВАХ

4.1 Базова структура моделі

Реалізація структури фінальної моделі відбувалась згідно з функціями та алгоритмами, описаними у розділі 3. Для реалізації моделей був використаний фреймворк Tensorflow на мові Python, оскільки він має в своєму складі повний набір інструментів для реалізації усіх необхідних складових, а також має вбудований механізм оптимізації роботи моделей, що дозволяє прискорити процес навчання.

Кодер має стандартну модель, описану у розділі 3.1 та у роботі, присвяченій GRU [6]. Код, що реалізує кодер наведено у лістингу 4.1

Лістинг 4.1 – Код, що реалізує логіку кодера

```
 class Encoder(tf.keras.Model):
    def __init__(self, vocab_size, embedding_dim, enc_units,
batch_sz):
        super(Encoder, self).__init__()
        self.batch_sz = batch_sz
        self.enc_units = enc_units
        self.embedding = tf.keras.layers.Embedding(vocab_size,
embedding_dim)
        self.gru = tf.keras.layers.GRU(self.enc_units,
                                       return_sequences=True,
                                       return_state=True,

recurrent_initializer='glorot_uniform')

    def call(self, x, hidden):
        x = self.embedding(x)
        output, state = self.gru(x, initial_state=hidden)
        return output, state

    def initialize_hidden_state(self):
        return tf.zeros((self.batch_sz, self.enc_units))
```

Аналогічно кодеру, декодер також реалізований згідно з описом у підрозділі 3.1. Реалізація алгоритму кодера наводиться у лістингу 4.2.

Наведений у лістингу механізм вже має розширення через інтеграцію методів оптимізації.

Лістинг 4.2 – Реалізація логіки декодера

```
    class Decoder(tf.keras.Model):
        def __init__(self, vocab_size, embedding_dim,
dec_units, batch_sz):
            super(Decoder, self).__init__()
            self.batch_sz = batch_sz
            self.dec_units = dec_units
            self.embedding =
tf.keras.layers.Embedding(vocab_size, embedding_dim)
            self.gru = tf.keras.layers.GRU(self.dec_units,
                                return_sequences=True,
                                return_state=True,
                           recurrent_initializer='glorot_uniform')
            self.fc = tf.keras.layers.Dense(vocab_size)
            self.attention = CachedAttention(self.dec_units)
            self.cache = Cache(batch_sz, dec_units, dec_units)
            self.cache_enabled = False

        def call(self, x, hidden, enc_output):
            context_vector, attention_weights =
self.attention(hidden, enc_output)
            x = self.embedding(x)
            x = tf.concat([tf.expand_dims(context_vector, 1),
x], axis=-1)
            output, state = self.gru(x)
            if self.cache_enabled:
                state = self.cache.modify(state,
context_vector)
            output = tf.reshape(output, (-1, output.shape[2]))
            x = self.fc(output)
            if self.cache_enabled:
                self.cache.put(context_vector, state, x)
            return x, state, attention_weights
```

Переклад вхідних речень у вихідні виконується з допомогою алгоритму, зазначеного в лістингу 4.3.

Лістинг 4.3 – Функція виконання перекладу

```
    def evaluate(sentence):
        attention_plot = np.zeros((max_length_targ,
max_length_inp))
        sentence = preprocess_sentence(sentence)
        inputs = [inp_lang.word_index[i] for i in
sentence.split(' ')]
        inputs =
tf.keras.preprocessing.sequence.pad_sequences([inputs],
    maxlen=max_length_inp,
    padding='post')
        inputs = tf.convert_to_tensor(inputs)
        result = ''
        hidden = [tf.zeros((1, units))]
        enc_out, enc_hidden = encoder(inputs, hidden)

        dec_hidden = enc_hidden
        dec_input =
tf.expand_dims([targ_lang.word_index['<start>']], 0)
        for t in range(max_length_targ):
            predictions, dec_hidden, attention_weights =
decoder(dec_input, dec_hidden, enc_out)
            attention_weights = tf.reshape(attention_weights,
(-1,))
            attention_plot[t] = attention_weights.numpy()
            predicted_id = tf.argmax(predictions[0]).numpy()
            result += targ_lang.index_word[predicted_id] + ' '
            if targ_lang.index_word[predicted_id] == '<end>':
                return result, sentence, attention_plot
            dec_input = tf.expand_dims([predicted_id], 0)
        return result, sentence, attention_plot
```

4.2 Навчання моделі з кешем

Дві стратегії проходження були визначені корисними для полегшення труднощів у навчанні, коли модель порівняно складна. Кеш додається до попередньо навченої моделі НМП з підготовкою лише нових параметрів, безпосередньо пов'язаних з ним.

В першу чергу, попередньо навчається стандартна модель НМП, яка здатна генерувати достатньо якісні переклади (c_t та s_t) для взаємодії з кешем. На наступному етапі, треновані параметри базової моделі фіксуються і тренуються нові параметри, пов’язані з кешем. Детальний опис методів навчання наведені у пункті 3.2.3. Для реалізації цього механізму навчання був написаний скрипт, наведений у лістингу Б.3.

4.3 Навчання моделі обробки декількох кодерів

В процесі інтеграції методу оптимізації в структуру моделі було виявлено, що метод конфліктує з методом кешування контекстів та методом перекладу розмов. Через це реалізація алгоритму була зупинена, а замість нього був використаний більш розповсюджений алгоритм реалізації уваги Багданау [9]. Кінцева структура елементу, що реалізує увагу, наведена у лістингу 4.5.

Лістинг 4.5 – Структура механізму уваги Багданау

```
class CachedAttention(tf.keras.layers.Layer):
    def __init__(self, units):
        super(CachedAttention, self).__init__()
        self.W1 = tf.keras.layers.Dense(units)
        self.W2 = tf.keras.layers.Dense(units)
        self.V = tf.keras.layers.Dense(1)

    def call(self, query, values):
```

Продовження лістингу 4.5

```
            query_with_time_axis = tf.expand_dims(query, 1)
            score = self.V(tf.nn.tanh(
                self.W1(query_with_time_axis)                +
self.W2(values)))
            attention_weights = tf.nn.softmax(score, axis=1)
            context_vector = attention_weights * values
            context_vector   =   tf.reduce_sum(context_vector,
axis=1)
            return context_vector, attention_weights
```

4.4 Навчання моделі перекладу розмов

Навчання базової моделі проводиться поетапно, тобто спочатку тренується частина, яка перекладатиме з англійської на іноземну та з іноземної на англійську, використовуючи паралельний корпус на рівні речень. Обидві архітектури мають однаковий словниковий запас, але окремі параметри, щоб уникнути зміщення вкладених елементів до моделі, навченої останньою. Контекстуальна модель попередньо підготовлена подібно до того, як готується базова модель. Найкраща модель обирається, виходячи з мінімальної загальної суперечності на двомовному датасеті для розробки.

Для відображення контексту джерела використовуються представлення речень, сформовані двома двонаправленими RNN на рівні речення (по одному для англійської та іноземних мов). Для представлення цільового речення використовуються останні приховані стани декодера, створені з попередньо навченої базової моделі. Однак під час декодування використовується останній прихований стан декодера, обчислений кінцевою моделлю (а не базовою), як цільові речення.

Реалізація даного алгоритму навчання наведена в лістингу Б.4.

4.5 Результати навчання моделі

Модель навчалась згідно з описаними алгоритмами на тренувальних наборах даних. Навчання проводилось в три етапи. На першому етапі була підготовлена лінійна частина моделі, на другому – модель кешу, а на третьому – модель перекладу розмов. На рисунках 4.1-2 наводяться порівняльні графіки зниження помилки у часі для першого та другого етапу відповідно. На ньому можна побачити порівняння швидкості навчання гібридної моделі зі звичайною лінійною моделлю.

Навчена система була перевірена на тестових даних та вручну за допомогою декількох запитів на переклад. Результати ручної перевірки наводяться у таблиці 4.1.

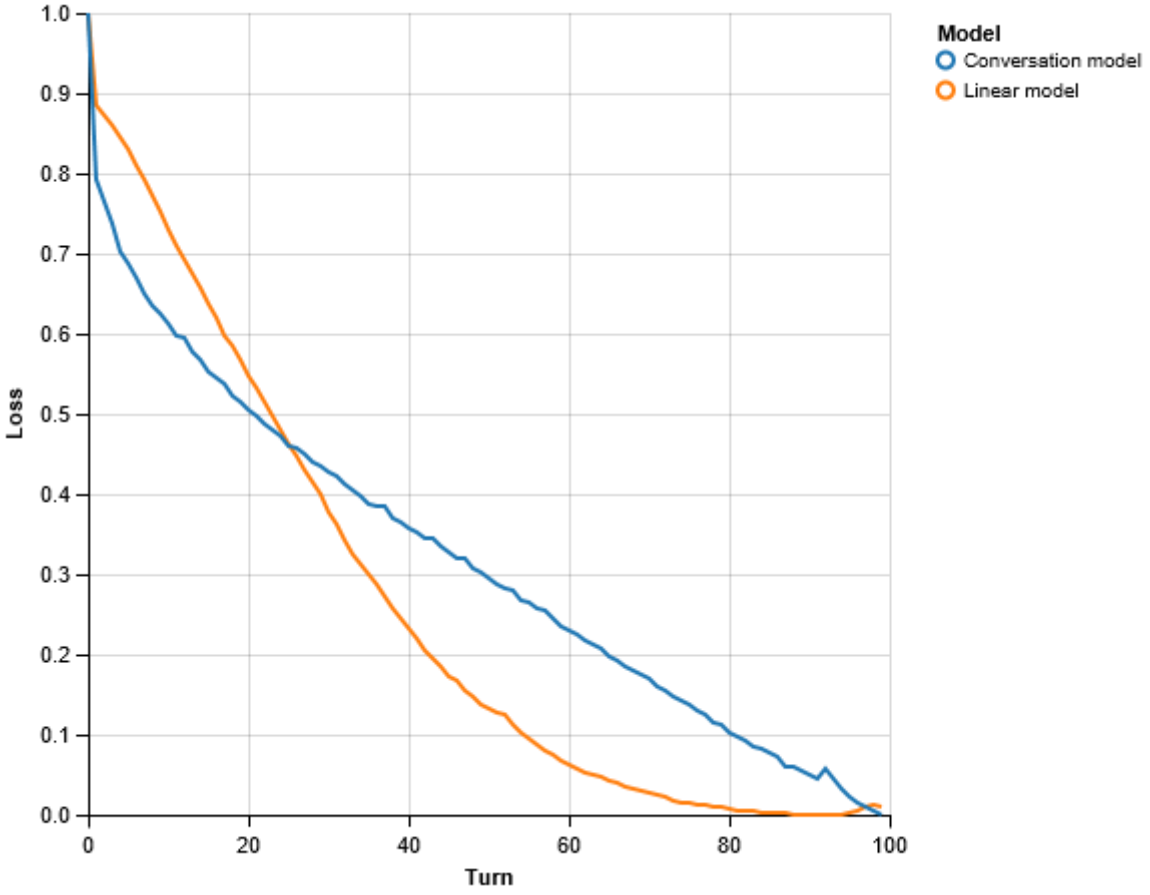

Рисунок 4.1 – Графік зниження помилки у часі для лінійного етапу. Значення функції втрат було нормовано до проміжку [0;1]

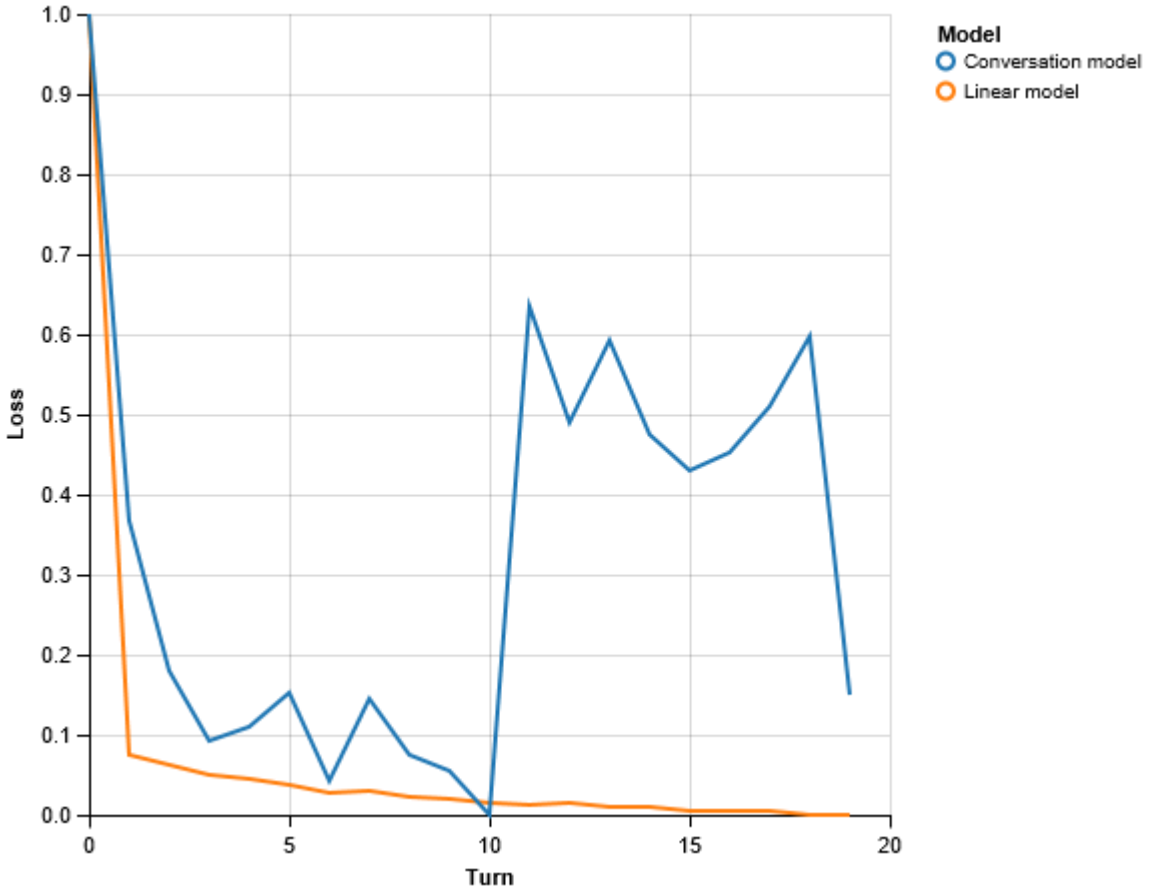

Рисунок 4.2 – Графік зниження помилки у часі для етапу навчання кешу. Значення функції втрат було нормовано до проміжку [0;1]

Як можна побачити з графіків, процес навчання гібридної моделі протікає повільніше, порівняно зі стандартною лінійною. Це пов'язано з відмінностями у структурі мереж. На відміну від лінійної структури, структура мережі для перекладу розмов заснована на двонаправлених RNN. Необхіність використання саме цього типу мереж зумовлена специфікою предметної області. Як наслідок, етапи навчання проходять повільніше та з меншою стабільністю, як можна відмітити з поведінки функції помилки під час навчання кешу.

Таблиця 4.1 – Результати ручної перевірки моделі. Дужками з трьома крапками відмічені повторювані фрази

Фраза	Лінійний переклад	Переклад як репліки	Еталон, отримано з допомогою Google Translate
This is an important example of european solidarity	die kommission ist ein notwendiger (schritt …)	die kommission ist ein notwendiger schritt	Dies ist ein wichtiges Beispiel für europäische Solidarität
This time we are going south	die von der eu afrika ist ein problem .	die von der eu afrika ist ein problem .	Diesmal fahren wir nach Süden
I have therefore voted in favour of this report .	ich habe fur die europaische parlament nimmt (den bericht gestimmt …).	ich habe fur die europaische parlament nimmt den bericht gestimmt .	Ich habe daher für diesen Bericht gestimmt.
i shall inform my honourable friend , who is unable to be here today to discuss this with you .	ich habe mich ich denke , und die kommission (, dass wir glauben …)	ich habe mich ich denke , und die kommission (, dass wir glauben …)	Ich werde meinen ehrenwerten Freund informieren, der heute nicht hier sein kann, um dies mit Ihnen zu besprechen.

Як можна побачити з таблиці 4.1, натренована модель з недостатньою точністю виконує переклад. Значна частина перекладених реплік, особливо в лінійному режимі, страждає від проблеми повторів, оскільки модель не здатна надійно визначити, коли необхідно закінчити речення. У порівнянні, модель у лінійному режимі демонструє гірші результати, ніж модель для перекладу розмов.

Отримані результати потенційно можна пояснити обмеженнями, які були в процесі навчання моделі, а саме малою кількістю навчальних прикладів та малою кількістю ітерацій навчання. Такі умови виникли через обмеження ресурсів платформи, на якій проводилось навчання.

Таким чином, на основі описаних у розділі 3 методів була створена модель машинного перекладу для перекладу розмов у груповому листуванні. Модель є життєздатною та може виконувати переклад, але зробити висновки щодо її можливої ефективності в рамках даної роботи неможливо.

ВИСНОВКИ

В кваліфікаційній роботі вирішується задача машинного перекладу в реальному часі листування багатонаціональної групи. Для вирішення цієї задачі були використані методи оптимізації нейронного машинного перекладу, а також метод, спеціально розроблений для обробки розмов.

Під час виконання роботи були докладно проаналізовані наукові джерела з обраної тематики, а саме – методи нейронного машинного перекладу текстів: методи засновані на RNN, CNN та статистичних методах, основна модель кодер-декодер, методи оптимізації роботи з контекстом в процесі перекладу. Під час розробки застосовані різні алгоритми та бібліотеки мови Python, яка найкращим чином пристосована для створення методів машинного навчання.

Отримана в ході виконання роботи модель була навчена на наборах даних, специфічних для її поетапного навчання. Якість перекладу є досить низькою та має певні недоліки, такі як зациклення та втрата слів. Потенційно, кращі результати можна було б отримати при більших обсягах корпусу даних, а також при довшій процедурі навчання, яку не було змоги провести, оскільки робота виконувалась в рамках обмежених ресурсів.

В роботі використовувались тексти англійською, німецькою та французькою мовою, отримані з відкритих офіційних джерел та попередньо оброблені задля подальшого використання.

Дана кваліфікаційна робота і подальші дослідження в цьому напрямку є актуальними, оскільки здатні полегшити та спростити міжнародне спілкування за рахунок зменшення необхідності активно використовувати лінійні перекладачі або покладатися на профільних фахівців в умовах, коли оплата їх послуг для ведення переговорів занадто велика.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Bengio Y. et. al. A neural probabilistic language model. Journal of Machine Learning Research. 2003. 2 March. P. 1137–1155.

2. Schwenk H., Dechelotte D., Gauvain J. Continuous space language models for statistical machine translation. In Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, Sydney, Australia, July, 2006. Stroudsburg, PA: Association for Computational Linguistics, 2006. P. 723–730.

3. Devlin J. et. al. Fast and robust neural network joint models for statistical machine translation. In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, Baltimore, MD, USA, June, 2014. Stroudsburg, PA: Association for Computational Linguistics, 2014. Vol. 1. P. 1370 – 1380.

4. Kalchbrenner N., Blunsom P. Recurrent continuous translation models. In Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing. Stroudsburg, PA: Association for Computational Linguistics, 2013. P. 1700–1709.

5. Hochreiter S., Schmidhuber J. Long short-term memory. Neural Computation. 1997. Vol. 9. No. 8. P. 1735– 1780.

6. Cho K. et. al. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Doha, Qatar, 25–29 October, 2014. Stroudsburg, PA: Association for Computational Linguistics, 2014. P. 1724–1734.

7. Chung J. et. al. Empirical evaluation of gated recurrent neural networks on sequence modeling. In Proceedings of NIPS 2014 Deep Learning and Representation Learning Workshop, Montreal, QC, Canada, 8-13 December 2014.

8. Pouget-Abadie J. et. al. Overcoming the curse of sentence length for neural machine translation using automatic segmentation. In Proceedings of

SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation, Doha, Quatar, 25 October, 2014. Stroudsburg, PA: Association for Computational Linguistics, 2014. P. 78–85.

9. Bahdanau D., Cho K., Bengio Y. Neural machine translation by jointly learning to align and translate. In Proceedings of the International Conference on Learning Representations, San Diego, CA, USA, 7-9 May, 2015. Stroudsburg, PA: Association for Computational Linguistics, 2015. P. 141-155.

10. Luong T., Pham H., Manning D. C. Effective approaches to attention-based neural machine translation. In Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP), Lisbon, Portugal, 17-21 September, 2015. Stroudsburg, PA: Association for Computational Linguistics, 2015. P. 1412–1421.

11. Wiseman S., Rush A. M. Sequence-to-sequence learning as beam-search optimization. In Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, Austin, TX, USA, 1-5 November, 2016. Stroudsburg, PA: Association for Computational Linguistics, 2016. P. 1296–1306.

12. Bojar O. et. al. Findings of the 2016 conference on machine translation: in Proceedings of the First Conference on Machine Translation, Berlin, Germany, 11-12 August, 2016. Stroudsburg, PA: Association for Computational Linguistics, 2016. Vol. 2. P. 131–198.

13. Sennrich R., Haddow B., Birch A. Edinburgh neural machine translation systems for WMT16. In Proceedings of the First Conference on Machine Translation, Berlin, Germany, 11-12 August, 2016. Stroudsburg, PA: Association for Computational Linguistics, 2016. Vol. 2. P. 371–376.

14. Sennrich R., Haddow B., Birch A. Improving neural machine translation models with monolingual data. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, Berlin, Germany, 7-12 August, 2016. Stroudsburg, PA: Association for Computational Linguistics, 2016. Vol. 1. P. 86–96.

15. Sennrich R., Haddow B., Birch A. Neural machine translation of rare words with subword units. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, Berlin, Germany, 7-12 August, 2016. Stroudsburg, PA: Association for Computational Linguistics, 2016. Vol. 1. P. 1715–1725.

16. Sennrich R., Haddow B. Linguistic input features improve neural machine translation. In Proceedings of the First Conference on Machine Translation, Berlin, Germany, 11-12 August, 2016. Stroudsburg, PA: Association for Computational Linguistics, 2016. Vol 2. P. 83–91.

17. Junczys-Dowmunt M., Dwojak T., Hoang H. Is neural machine translation ready for deployment? A case study on 30 translation directions. In Proceedings of IWSLT (13th International Workshop on Spoken Language Technology), Seattle, WA, USA, 8-9 December, 2016. Stroudsburg, PA: Association for Computational Linguistics, 2016. P. 1-8.

18. Wu Y. et. al. Google's neural machine translation system: Bridging the gap between human and machine translation. CoRR, 2016. abs/1609.08144. 23 p.

19. Gehring J. et. al. Convolutional sequence to sequence learning / ed I. D. Precup, Y. W. Teh. Proceedings of Machine Learning Research: Proceedings of the 34th International Conference on Machine Learning, Sydney, Australia, 06–11 Aug 2017. Vol. 70. P. 1243–1252.

20. Vaswani A. et. al. Attention is all you need / I. I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, R. Garnett. Advances in Neural Information Processing Systems. Curran Associates, Inc., 2017. Vol. 30. P. 5998–6008.

21. Bojar O. et. al. Findings of the 2018 conference on machine translation (WMT18). In Proceedings of the Third Conference on Machine Translation, Brussels, Belgium, 31 October – 1 November, 2018. Stroudsburg, PA: Association for Computational Linguistics, 2018. P. 272–303.

22. Chen M. X. et. al. The best of both worlds: Combining recent advances in neural machine translation. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, Melbourne, Australia, 15-20 July, 2018. Stroudsburg, PA: Association for Computational Linguistics, 2018. Vol. 1. P. 76–86.

23. Zhang B., Xiong D., Su J. Neural machine translation with deep attention. IEEE Transactions on Pattern Analysis and Machine Intelligence. 2018. Vol 40, No. 10. P. 154-163.

24. Maruf S., Martins A. F. T., Haffari G. Contextual Neural Model for Translating Bilingual Multi-Speaker Conversations. Proceedings of the Third Conference on Machine Translation, Brussels, Belgium, 31 October – 1 November, 2018. Stroudsburg, PA: Association for Computational Linguistics, 2018. P. 101-112.

25. Tu Z. et. al. Learning to Remember Translation History with a Continuous Cache. Transactions of the Association for Computational Linguistics. Stroudsburg, PA: Association for Computational Linguistics, 2018. Vol 7. P. 407-420.

26. Wang L. et. al. Exploiting Cross-Sentence Context for Neural Machine Translation. Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, Copenhagen, Denmark, 7-11 September, 2017. P. 2826-2831

27. Liu F., Lu H., Neubig G. Handling Homographs in Neural Machine Translation. Human Language Technologies: Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics, 1-6 June, 2018. Vol 1. P. 1336-1345.

28. Bahdanau D., Cho K., Bengio Y. Neural Machine Translation by Jointly Learning to Align and Translate. In Proceedings of 3rd International Conference on Learning Representations, San Diego, CA, USA, 7 – 9 May, 2015. Trier, Germany: Schloss Dagstuhl – Leibniz Center for Informatics, 2015. P. 114-128.

29. Gonzales A. R., Mascarell L., Sennrich R. Improving Word Sense Disambiguation in Neural Machine Translation with Sense Embeddings. Proceedings of the Second Conference on Machine Translation, Copenhagen, Denmark, 7-8 September, 2017. Stroudsburg, PA: Association for Computational Linguistics, 2017. P. 11-19.

30. Wong K., Maruf S., Haffari G. Contextual Neural Machine Translation Improves Translation of Cataphoric Pronouns. Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, 5-10 July, 2020. Stroudsburg, PA: Association for Computational Linguistics, 2020. P. 5971-5978.

31. Voita E. et. al. Context-Aware Neural Machine Translation Learns Anaphora Resolution. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, Melbourne, Australia, 15-20 July, 2018. Stroudsburg, PA: Association for Computational Linguistics, 2018. Vol. 1. P. 1264-1274.

32. Dauphin Y. N. et. al. Language Modeling with Gated Convolutional Networks. Proceedings of the 34th International Conference on Machine Learning, Sydney, Australia, 6-11 August, 2017. PMLR, 2017. Vol 70. P. 933-941.

33. Choi H., Cho K., Bengio Y. Context-Dependent Word Representation for Neural Machine Translation. Computer Speech & Language. 2017. September. Vol. 45. P 149-160.

34. Xiao T. et. al. Document-level consistency verification in machine translation. In Proceedings of 13th Machine Translation Summit, Xiamen, China, 1-23 September, 2011. P. 131–138.

35. Carpuat M., Simard M. The trouble with SMT consistency. In Proceedings of the 7th Workshop on Statistical Machine Translation. Montreal, Quebec, Canada, 7-8 June, 2012. Stroudsburg, PA: Association for Computational Linguistics, 2012. P. 442–449.

36. Sordoni A. et. al. A hierarchical recurrent encoder-decoder for generative context-aware query suggestion. In Proceedings of the 24th ACM International Conference on Information and Knowledge Management. Melbourne, Australia, 19-23 October, 2015. New York, NY: Association for Computing Machinery, 2015. P. 553–562.

37. Vinyals O., Le Q. A neural conversational model. In Proceedings of the International Conference on Machine Learning, Deep Learning Workshop, Lille, France, 6-11 July, 2015. PMLR, 2015. P. 1–8.

38. Serban I. V. et. al. Building end-to-end dialogue systems using generative hierarchical neural network models. In Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, Phoenix, Arizona, USA, 12-17 February, 2016. AAAI Press, 2016. P. 3776–3783.

39. Pelevina M. et. al. Making Sense of Word Embeddings. In Proceedings of the 1st Workshop on Representation Learning for NLP, Berlin, Germany, 11 August, 2016. Stroudsburg, PA: Association for Computational Linguistics, 2016. P. 174–183.

40. Grave E., Joulin A., Usunier N. Improving neural language models with a continuous cache. In Proceedings of International Conference on Learning Representations, Toulon, France, 24 – 26 April, 2017. Trier, Germany: Schloss Dagstuhl – Leibniz Center for Informatics, 2017. P. 1-9.

41. Daniluk M. et. al. Frustratingly short attention spans in neural language modeling. In Proceedings of International Conference on Learning Representations, Toulon, France, 24 – 26 April, 2017. Trier, Germany: Schloss Dagstuhl – Leibniz Center for Informatics, 2017. P. 100-110.

42. Miller A. et. al. Key-value memory networks for directly reading documents. In Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP), 1-5 November, 2016. Stroudsburg, PA: Association for Computational Linguistics, 2016. P. 1400–1409.

43. Tu Z. et. al. Context gates for neural machine translation. In Proceedings of the Eighth International Conference on Topology, Algebra and

Categories in Logic, 26–30 June, 2017. Stroudsburg, PA: Association for Computational Linguistics, 2017. Vol 5. P. 87– 99.

44. Libovický J., Helcl J. Attention Strategies for Multi-Source Sequence-to-Sequence Learning. Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, Vancouver, Canada, 30 July – 4 August, 2017. Stroudsburg, PA: Association for Computational Linguistics, 2017. Vol. 2. P. 196-202.

45. Cettolo M., Girardi C., Federico M. *WIT3*: Web Inventory of Transcribed and Translated Talks. In Proceedings of 16th annual Conference of the European Association for Machine Translation, Trento, Italy, 28-30 May, 2012. Trier, Germany: Schloss Dagstuhl – Leibniz Center for Informatics, 2012. P. 261-268.

46. Чала Л. Е., Удовенко С. Г., Гринєв С. А. Метод нейромережевої корекції помилок в електронних текстах, що редагуються. *Біоніка інтелекту*. 2017. № 1 (88). С. 15 – 21.