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)

Розроблення системи виправлення
граматичних та синтаксичних помилок в україномовних текстах
(тема)

Виконав:
здобувач другого року навчання,
групи ДСм-23-1
Пилипенко Р.О.
(прізвище, ініціали)

Спеціальність 122 Комп'ютерні науки
(код і повна назва спеціальності)
Тип програми освітньо-професійна
(освітньо-професійна або освітньо-наукова)
Освітня програма Науки про дані (Data Science)
(повна назва спеціалізації)
Керівник доц. Дейнеко А.О.
(посада, прізвище, ініціали)

Допускається до захисту

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

2025 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-професійна
(освітньо-професійна або освітньо-наукова)

Освітня програма Науки про дані (Data Science)
(повна назва)

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

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

здобувачеві Пилипенку Роману Олеговичу
(прізвище, ім'я, по батькові)

1. Тема роботи Розроблення системи виправлення граматичних та синтаксичних помилок в україномовних текстах

затверджена наказом університету від 22 листопада 20 24 р. № 1238Ст
2. Термін подання студентом роботи до екзаменаційної комісії 23 січня 20 25 р.
3. Вихідні дані до роботи Науково-технічні публікації, дані Інтернет-джерел та відомих наукових проєктів, документація до бібліотек

4. Перелік питань, що потрібно опрацювати в роботі
1) Аналіз предметної галузі
2) Дослідження методів виправлення граматичних помилок
3) Дослідження методів генерування штучних помилок
4) Побудова та тестування системи виправлення граматичних помилок

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Отримання завдання на кваліфікаційну роботу	25.11.2024	виконано
2	Аналіз предметної галузі	27.11.2024	виконано
3	Огляд методів виправлення граматичних	02.12.2024	виконано
	помилок		
4	Огляд методів генерування штучних наборів	15.12.2024	виконано
	даних		
5	Навчання моделей виправлення граматичних	27.12.2024	виконано
	помилок		
6	Написання пояснювальної записки	02.01.2025	виконано
7	Перевірка на академічний плагіат	13.01.2025	виконано
8	Нормоконтроль	14.01.2025	виконано
9	Підготовка презентації та доповіді	14.01.2025	виконано
10	Представлення на рецензування	18.01.2025	виконано
11	Захист перед ЕК	23.01.2025	

Дата видачі завдання 25 листопада 2024 р.

Здобувач _____
(підпис)

Керівник роботи _____ доц. Дейнеко А.О.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка: 83 с., 14 рис., 4 табл., 2 дод., 31 джерело.

ВИПРАВЛЕННЯ ГРАМАТИЧНИХ ПОМИЛОК, ІН'ЄКЦІЯ ШУМУ, НЕЙРОННИЙ МАШИННИЙ ПЕРЕКЛАД, СТАТИСТИЧНИЙ МАШИННИЙ ПЕРЕКЛАД, СТВОРЕННЯ СИНТЕТИЧНИХ ПОМИЛОК, ТРАНСФОРМЕР, UA-GEC.

Об'єкт дослідження – процес виправлення граматичних помилок в україномовних текстах.

Предмет дослідження – трансформерні моделі, що навчаються на анотованих та штучно створених даних для вирішення задачі виправлення граматичних помилок.

Мета роботи – розроблення системи виправлення граматичних та синтаксичних помилок для української мови.

Методи дослідження – аналіз літературних джерел, використання алгоритмів нейронного машинного перекладу, методів обробки природної мови, генерація синтетичних помилок, експериментальне навчання моделей на основі архітектури трансформерів.

У ході роботи були досліджені теоретичні підходи до виправлення граматичних помилок та створення синтетичних помилок для малоресурсних завдань. Розглянуто різні методи, включаючи ін'єкцію шуму та зворотний переклад. Експериментально реалізовано новий підхід із генерацією англіцизмів для синтетичних прикладів. Також проведено дослідження різних нейромережевих архітектур для розв'язання задачі корекції граматичних помилок.

ABSTRACT

Master’s thesis contains: 83 pp., 14 fig., 4 tabl., 2 ann., 31 references.

GRAMMATICAL ERROR CORRECTION, NEURAL MACHINE TRANSLATION, NOISE INJECTION, STATISTICAL MACHINE TRANSLATION, SYNTHETIC DATA GENERATION, TRANSFORMER, UA-GEC.

The object of the research is the process of grammatical error correction in Ukrainian-language texts.

The subject of the research is transformer models trained on annotated and artificially generated data for solving the task of grammatical error correction.

The aim of the work is to develop a system for correcting grammatical and syntactical errors for the Ukrainian language.

The research methods include analysis of literary sources, the use of neural machine translation algorithms, natural language processing techniques, synthetic error generation, and experimental training of models based on transformer architecture.

During the work, theoretical approaches to grammatical error correction and synthetic error generation for low-resource tasks were explored. Various methods were considered, including noise injection and back-translation. A new approach was experimentally implemented involving the generation of Anglicisms for synthetic examples. Additionally, different neural network architectures were investigated for solving the task of grammatical error correction.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

AI – Artificial Intelligence – штучний інтелект;

GEC – Grammatical Error Correction – виправлення граматичних помилок;

LM – Language Model – мовна модель;

NLP – Natural Language Processing – обробка природної мови;

NMT – Neural Machine Translation – нейронний машинний переклад;

RNN – Recurrent Neural Network – рекурентна нейронна мережа;

Seq2Seq – Sequence-to-Sequence – архітектура мереж, де на вхід подається послідовність, а на вихід генерується інша послідовність;

SMT – Statistical Machine Translation – статистичний машинний переклад.

ВСТУП

У сучасному інформаційному суспільстві текстові дані набули важливого значення в багатьох сферах життя, включаючи освіту, науку, бізнес та міжособистісну комунікацію. Якість текстів, зокрема правильність їхньої граматичної та синтаксичної структури, є важливим аспектом ефективної передачі інформації. Однак в умовах цифрової епохи, коли обсяг текстових даних зростає, а час на їх створення та редагування обмежений, постає проблема автоматизації процесу виявлення та виправлення помилок у текстах.

Автоматичне виправлення граматичних і синтаксичних помилок є складним завданням, яке вимагає глибоких знань у галузі комп'ютерної лінгвістики, методів обробки природної мови та машинного навчання. Особливо актуальною ця проблема є для української мови, оскільки кількість доступних інструментів та ресурсів для її автоматичної обробки значно менша порівняно з іншими мовами.

Дослідження, спрямовані на створення систем автоматичного виправлення помилок, сприяють не лише підвищенню якості текстів, але й стимулюють розвиток нових технологій у галузі обробки природної мови. У цій роботі зосереджено увагу на розробленні системи для виправлення граматичних та синтаксичних помилок в україномовних текстах. Вибір цієї теми обумовлений її актуальністю, зумовленою потребами суспільства та перспективністю досліджень у цій галузі.

Результати дослідження можуть бути корисними для різних сфер, зокрема для освітніх платформ, редакційних систем та інших застосунків, що використовуються для роботи з текстами. У межах цієї роботи буде здійснено аналіз існуючих підходів до вирішення проблеми, проведено експериментальні дослідження та запропоновано ефективні методи для реалізації системи виправлення помилок.

1 АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ

1.1 Огляд обробки природної мови

Обробка природної мови (англ. Natural Language Processing, NLP) – міждисциплінарна галузь, яка поєднує знання з інформатики, лінгвістики та штучного інтелекту з метою автоматизації аналізу, обробки та розуміння текстів або усного мовлення природними мовами. Завдяки швидкому розвитку обчислювальних технологій, NLP стає все більш актуальною у вирішенні завдань автоматизації, пов'язаних із великими обсягами текстової інформації, таких як машинний переклад, виявлення фейкових новин, розпізнавання голосу, автоматичне резюмування текстів і, зокрема, виправлення граматичних та синтаксичних помилок.

Головною метою обробки природної мови є створення систем, які можуть правильно інтерпретувати мовленнєві та текстові дані, незважаючи на багатозначність і складність природних мов. Людська мова є висококонтекстуальною, включає багатозначні слова, неоднозначності у структурі речень, а також значну кількість ідіом, що створює значні виклики для обчислювальних моделей.

Сьогодні розвиток обробки природної мови значною мірою визначається поширенням великих мовних моделей (Large Language Models, LLMs), які базуються на сучасних алгоритмах машинного навчання. Досягнення в цій галузі дозволили значно підвищити точність та ефективність систем NLP завдяки використанню великих обсягів текстових даних для навчання та покращенню моделей за допомогою трансформерів.

Важливим фактором, який стимулює розвиток NLP, є необхідність роботи з текстами різними мовами, включаючи українську, де моделі стикаються з особливими викликами, зумовленими морфологічною складністю мови, варіативністю синтаксису та недостачею навчальних даних. Таким чином, обробка природної мови є ключовою галуззю, яка

сприяє автоматизації та підвищенню ефективності роботи з текстами, а її розвиток дозволяє створювати все більш досконалі системи для вирішення завдань у різних сферах.

1.2 Огляд задачі виправлення граматичних помилок

Темою цієї роботи є вирішення задачі виправлення граматичних та синтаксичних помилок, проте загалом ця задачі називається grammatical error correction (далі GEC), що включає в себе вирішення не тільки граматичних помилок, а й усіх інших пов’язаних з текстом – пунктуаційних, орфографічних, лексичних, граматичних, морфологічних, синтаксичних тощо. Більшість систем для виправлення граматичних помилок отримують на вхід необроблене, неграматичне речення, а на вихід видають відредаговане, граматично правильне речення. Типовий приклад проілюстрований на рисунку 1.1.

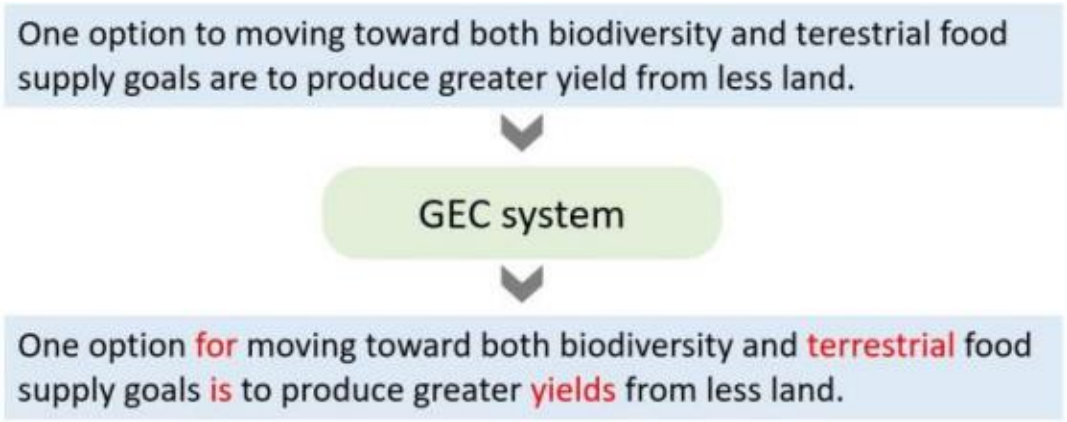

Рисунок 1.1 – Приклад виправлення граматичних помилок

2000-ті роки стали попереднім етапом розвитку систем виправлення граматичних помилок. На цьому етапі більшість систем GEC базувалися на ручних правилах, що використовували парсери та лінгвістичні характеристики, наприклад, Language Tool [1] та ESL Assistant [2]. Однак

складність у розробці правил та вирішенні конфліктів між ними потребувала значних зусиль. Хоча деякі сучасні роботи з GEC все ще використовують правила як додаткове джерело виправлення, продуктивність систем GEC на основі правил була перевершена підходами, що надалі розглядатимуться в цій роботі.

Помітний прогрес у задачі виправлення граматичних та синтаксичних помилок став можливим завдяки підходам, заснованим на аналізі даних. Наприкінці 2000-х років були розроблені методи, що використовували класифікацію для виправлення певних типів помилок, таких як неправильне вживання прийменників чи артиклів. У цих методах застосовували класифікатори, які тренувалися на великих масивах текстів без помилок. Такі моделі аналізували контекст речення та на основі його лінгвістичних особливостей передбачали правильний варіант слова. Незважаючи на свою ефективність для вузькоспеціалізованих задач, ці підходи були обмежені у виправленні більш складних чи комбінованих помилок.

На початку 2010-х років значної уваги набули системи виправлення на основі статистичного машинного перекладу (SMT). Цей підхід дозволяв виправляти всі типи помилок шляхом «перекладу» помилкових речень у виправлені. Для цього використовували моделі, треновані на паралельних наборах речень, у яких одне речення містило помилки, а інше було їхнім виправленням. Завдяки цьому SMT-системи стали першими універсальними підходами до задачі GEC, проте їхня ефективність залежала від наявності великих паралельних корпусів даних.

З часом, із поширенням глибокого навчання, системи на основі нейронного машинного перекладу (NMT) стали домінуючими в цій галузі. Вони використовували нейронні моделі послідовностей (seq2seq), які забезпечували ще вищу точність та гнучкість у виправленні помилок. Цей підхід дозволив досягти передового рівня продуктивності в задачі GEC. Однак NMT системи також залежать від великих паралельних корпусів, що стимулювало дослідження альтернативних підходів, таких як моделі,

засновані на мовних моделях, які не потребують настільки об'ємного навчального набору. Теорія підходів на основі статистичного машинного перекладу та нейронного машинного перекладу буде детально розглянута у другому розділі цієї роботи.

Згадані раніше підходи належать до категорії seq2seq, тобто таких, що на вхід приймають послідовність, тобто текст, і на виході віддають також послідовність. Існує також інша категорія підходів – seq2edit. Підхід на основі генерації правок (edit generation approach) пропонує створювати послідовність правок, які потрібно застосувати до вхідного речення, замість того, щоб генерувати весь текст цілком. Оскільки у задачі виправлення граматичних і синтаксичних помилок значна частина токенів у вихідному тексті є копією вхідного, дослідники Stahlberg та Kumar [3] зазначили, що повна генерація тексту є неефективною. Замість цього генерація лише правок дозволяє суттєво підвищити швидкість обробки: такі системи працюють у 5–10 разів швидше, ніж GEC-моделі, які створюють весь вихідний текст.

Однак цей підхід має і свої обмеження. Генерація правок зазвичай базується на токенах, що може ускладнювати внесення більш складних змін, які потребують корекції кількох токенів одночасно для покращення загальної зв’язності тексту. Підхід з використанням генерації правок реалізується або у вигляді задачі розмітки послідовностей (sequence tagging), як це описано у роботі Malmi та інших [4], або як задача послідовність-послідовність (sequence-to-sequence), як зазначено у раніше згаданій роботі [3].

У підході розмітки послідовностей (sequence tagging) для кожного токена у вхідному реченні система прогнозує операцію редагування, яку слід застосувати до цього токена. Для цього необхідно визначити набір тегів, які представлятимуть ці операції. Деякі правки можуть бути уніфікованими, наприклад, зміна форми дієслова або перетворення іменника з однини у множину. Інші ж, такі як вставка чи заміна слова,

залежать від конкретного токена. Для кожного можливого слова у словнику потрібен окремий тег, через що кількість таких токен-залежних тегів зростає лінійно зі збільшенням числа унікальних слів у навчальних даних. Таким чином, вибір кількості токен-залежних тегів стає компромісом між повнотою покриття та розміром моделі.

Натомість підхід послідовність-послідовність (sequence-to-sequence) є більш гнучким, оскільки він не обмежує вихідний результат попередньо визначеними тегами операцій редагування. Замість цього модель генерує послідовність правок, кожна з яких включає позицію фрагмента, рядок заміни та, за потреби, тег, який визначає тип правки. Використання тегів додає інтерпретованість процесу та, як показали дослідження, покращує продуктивність моделі.

Оскільки у підході sequence-to-sequence генерація відбувається зліва направо, процедура прогнозування є повільнішою, ніж у підході sequence tagging. Однак вона все одно приблизно у п'ять разів швидша за підхід, який генерує все речення повністю, оскільки створювана послідовність правок значно коротша за послідовність усіх токенів у реченні.

Незважаючи на перевагу швидкодії підходу seq2edit, теоретичне та практичне дослідження буде зосереджено на підході seq2seq, зважаючи на вже існуючі обґрунтовані методи для вирішення задачі виправлення граматичних помилок в україномовних текстах.

З точки зору задачі виправлення граматичних помилок українська мова – малоресурсна та недостатньо досліджена. Нещодавно був випущений корпус текстів українською мовою з анотаціями помилок, написаних носіями та неносіями мови [5]. У випадку невеликого обсягу даних застосовується техніка аугментації даних, що передбачає створення штучних екземплярів, яких не було в оригінальному датасеті. Різні техніки створення синтетичних даних будуть детально розглянуті у третьому розділі цієї роботи.

1.3 Актуальність задачі виправлення граматичних помилок

В сфері обробки природної мови задача виправлення граматичних помилок – одна з ключових через її практичну важливість у сучасному цифровому світі, нарівні із задачами POS, розпізнавання тональностей, класифікації текстів, резюмування, задачі на основі запитів і відповідей (Question Answering). З розвитком технологій текстова комунікація стає дедалі важливішою – електронні листи, месенджери, соціальні мережі та онлайн-освіта широко використовуються для спілкування. Автоматичне виправлення помилок не лише підвищує рівень грамотності, а й забезпечує ефективність спілкування, роблячи його зрозумілішим і доступнішим.

Як один із фундаментальних напрямків NLP, виправлення граматичних помилок сприяє природнішій взаємодії між людьми та машинами. Подібно до машинного перекладу чи аналізу тексту, ця задача фокусується на покращенні якості й зрозумілості текстів. Застосування таких технологій дозволяє навіть людям із базовими знаннями мови створювати тексти високого рівня, зменшуючи мовні бар’єри. Це також має значний вплив на автоматизацію бізнес-процесів, обробку великих обсягів текстової інформації та підвищення ефективності освітніх технологій, де граматичні коректори допомагають студентам розвивати навички письма.

У цьому контексті виправлення граматичних помилок для української мови є не менш актуальним, адже українська, будучи мовою зі складною морфологією та синтаксисом, залишається недостатньо дослідженою в порівнянні з іншими поширеними мовами, такими як англійська. Популяризація української мови та її зростаюче використання в офіційних, освітніх і ділових сферах створюють значний запит на інструменти, які допоможуть забезпечити грамотність текстів.

Помилки у текстах українською мовою можуть не лише ускладнювати сприйняття інформації, а й впливати на професійний чи академічний імідж

автора. Особливо це актуально у сфері цифрової комунікації, де тексти створюються швидко, часто без належної перевірки. Інструменти автоматичного виправлення помилок можуть не лише підтримувати якість текстів, а й сприяти підвищенню рівня грамотності серед користувачів, включаючи тих, хто вивчає українську як іноземну.

Розробка високоякісних систем GEC для української мови є також важливою в освітньому контексті, де такі інструменти допомагають студентам отримувати швидкий і точний зворотний зв’язок. Сучасні підходи, такі як seq2seq моделі, дозволяють створювати рішення, що враховують особливості української мови, забезпечуючи високу точність і адаптивність систем.

Таким чином, виправлення граматичних помилок залишається актуальним як у глобальному контексті, так і в аспекті розвитку технологій для української мови. Це не лише засіб забезпечення грамотності, а й важливий етап у створенні інтелектуальних систем, здатних глибше розуміти й обробляти природну мову.

1.4 Постановка задачі

У межах цієї роботи необхідно дослідити сучасні методи виправлення граматичних і синтаксичних помилок у текстах, зокрема ті, що базуються на підходах класифікації, статистичного машинного перекладу та нейронного машинного перекладу. Також слід опрацювати теоретичні аспекти механізму уваги та архітектури Трансформер, оскільки вони є ключовими у сучасних моделях для задач обробки природної мови.

Для виконання поставлених завдань передбачено здійснення пошуку корпусів даних, придатних для задачі автоматичного виправлення граматичних помилок в українськомовних текстах. З огляду на те, що задача виправлення помилок українською мовою належить до категорії малоресурсних задач, особливу увагу слід приділити дослідженню методів

генерації синтетичних даних. Це включає аналіз сучасної літератури та підходів до створення даних із використанням різноманітних технік.

Також необхідно вивчити та проаналізувати сучасні State-of-the-Art методи, застосовані для задач виправлення граматичних помилок, із фокусом на їх адаптацію для української мови. На основі проведеного аналізу буде визначено оптимальні методи, які доцільно використовувати в рамках цієї роботи.

Практична частина передбачає створення синтетичних даних відповідно до обраних підходів, що сприятиме підвищенню обсягу й різноманітності навчального корпусу. Заплановано навчити одну або кілька моделей для задачі автоматичного виправлення граматичних і синтаксичних помилок. Навчання моделей здійснюватиметься із використанням сучасних бібліотек глибокого навчання.

Результати досліджень дозволять розробити ефективну систему для виправлення помилок в україномовних текстах, яка буде адаптована до специфіки мови та враховуватиме її особливості.

2 МЕТОДИ ВИПРАВЛЕННЯ ГРАМАТИЧНИХ ПОМИЛОК

2.1 Методи на основі класифікації

У контексті корекції граматичних помилок, раніше широко використовувалися підходи, засновані на класифікації. У таких підходах тренується багатокласовий класифікатор, який визначає правильне слово з набору можливих синонімічних замін, різних форм оригінального слова тощо. Класифікатор використовує ознаки контексту слова, які можуть бути отримані або за допомогою штучного інженерства ознак, або через моделі глибинного навчання, для здійснення передбачень. Коли подається текст для корекції, класифікатор прогнозує найбільш імовірного кандидата для кожного токена в тексті, що належить до набору сплутаних варіантів. Класифікатор тренується на даних від носіїв мови, що означає, що йому не потрібно анотоване навчальне середовище, що робить цей метод переважним у порівнянні з методами з наглядом. До найбільш популярних алгоритмів класифікації належать Максимальна Ентропія (ME), Наївний Баєс (NB), Дерево рішень та Усереднений Перцептрон (AP). Наприклад, у роботі [6] було розроблено систему з 8 комбінованих алгоритмів класифікації (де кожен алгоритм є або NB, або AP) для кожного типу помилки. Система досягла значення $F_{0.5}$ на рівні 37.13% на набір даних CoNLL-2014 [7]. Хоча ці підходи були популярними в минулому, зараз вони не відповідають сучасним рішенням, що будуть описані далі в цій роботі.

2.2 Методи на основі статистичного машинного перекладу

У контексті корекції граматичних помилок, до використання SMT, основними були методи на основі правил та класифікації, кожні з яких мали свої обмеження.

З одного боку, розробка правил часто вимагає великої кількості попередньо зібраної лінгвістичної інформації та знань експертів. Процес створення правил є тривалим, а лінгвістичні ресурси не завжди доступні, особливо для мов, для котрих не існує великої кількості корпусів навчальної інформації. Крім того, універсальних правил не існує, оскільки природна мова настільки гнучка, що складно створити правила, які б ефективно враховували всі можливі винятки. З іншого боку, гнучкість природної мови також обмежує ефективність методів, заснованих на класифікації, для досягнення значних результатів у більш широких контекстах. Наперед задані мітки обмежують класифікаційні методи певними типами помилок і не дозволяють їх розширювати для вирішення більш складних завдань.

Що стосується виправлення граматичних помилок (GEC), то помилки зазвичай пов'язані не тільки з неправильними словами, але й з контекстом, який може включати навколишні токени в реченні або навіть інформацію між реченнями [8].

З математичної точки зору, метою статистичного машинного перекладу (SMT) є знаходження перекладу y в цільовій мові для заданого джерела x у вихідній мові, що ймовірнісно знаходить таке речення y, яке максимізує $p(y|x)$. Розподіл $p(y|x)$ є моделлю перекладу, яку ми хочемо оцінити. Використовуючи формулу Байєса, ми можемо сформулювати задачу перекладу через модель зашумленого каналу:

$$\mathrm{Arg}\max_{y} p\,(y|x) = \arg\max_{y} p\,(x|y)p(y). \qquad (2.1)$$

У цій моделі зазначено, що окрім інвертованої моделі перекладу, вона також включає мовну модель $p(y)$ на цільовій стороні. Додавши мовну модель, можна оцінити і покращити плавність вихідного речення. Ці дві моделі є незалежними, і для навчання мовної моделі не потрібні паралельні корпуси. Одномовний корпус з великою кількістю даних на цільовій мові може забезпечити добру мовну модель [9].

2.3 Підходи на основі нейронного машинного перекладу

Системи на основі нейронного машинного перекладу застосовують так званий механізм енкодера-декодера запропонованого у роботах Cho [10] та Sutskever [11]. Це нейромережева архітектура, що вчиться кодувати послідовність змінної довжини у векторне представлення фіксованої довжини та декодувати задане векторне представлення фіксованої довжини назад у послідовність змінної довжини. З точки зору ймовірностей, ця нова модель є загальним методом для вивчення умовного розподілу над послідовністю змінної довжини за умови іншої послідовності змінної довжини. Енкодер зчитує та закодовує повне вхідне речення $x = (x_1, x_2, \ldots, x_T)$ у вектор c:

$$c = q(h_1, h_2, \ldots, h_T), \tag{2.2}$$

де прихований стан h_T у момент часу t визначається як:

$$h_t = f(x_t, h_{t-1}). \tag{2.3}$$

Декодер запропонованої моделі – це інша RNN, що тренується генерувати вихідну послідовність $y = (y_1, y_2, \ldots, y_{T'})$, передбачаючи наступне слово y_t на основі закодованого вектора с та всіх попередньо передбачених слів $\{y_1, y_2, \ldots, y_{t-1}\}$:

$$p(y) = \prod_{t=1}^{T'} p(y_t | \{y_1, y_2, \ldots, y_{t-1}\}, c) = \prod_{t=1}^{T'} g(y_{t-1}, s_t, c), \tag{2.4}$$

де s_t – прихований стан декодера.

На рисунку 2.1 зображене графічне представлення архітектури моделі енкодер-декодер.

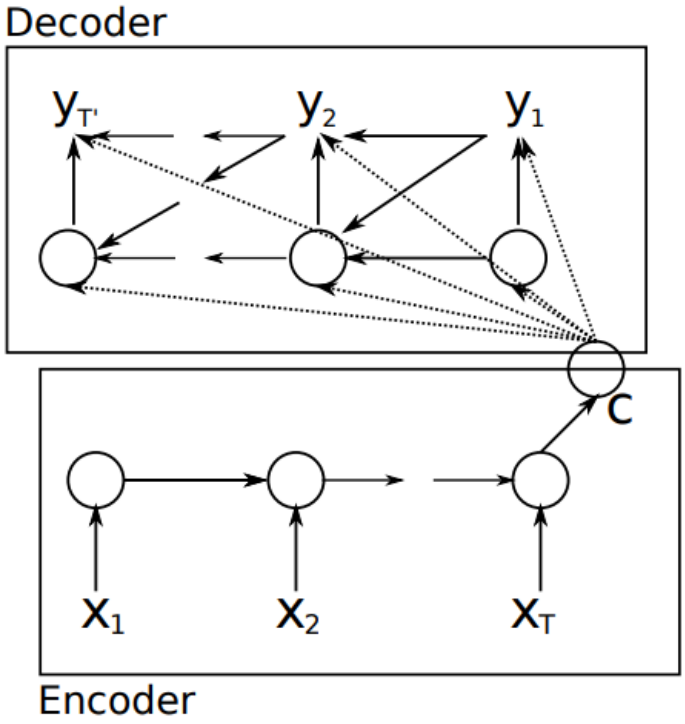

Рисунок 2.1 – Графічне представлення архітектури моделі енкодер-декодер

Два компонента запропонованої моделі RNN енкодера-декодера тренуються спільно для максимізації умовної логарифмічної ймовірності. Після тренування модель може бути використана двома способами. Один із способів – це використання моделі для генерації цільової послідовності на основі вхідної послідовності. З іншого боку, модель може бути використана для оцінки пари вхідної та вихідної послідовностей, де оцінка є просто ймовірністю $p_\theta(y \mid x)$.

У 2013 в своїй роботі Kalchbrenner та Blunsom [12] пропонують новий клас ймовірнісних безперервних моделей перекладу, званих Recurrent Continuous Translation Models, які базуються виключно на безперервних представленнях слів, фраз та речень, без використання вирівнювань чи фразових одиниць перекладу. Моделі мають два аспекти: генерацію перекладу за допомогою рекурентної мовної моделі та підлаштування на вихідному реченні через згорткову модель. У дослідженнях показано, що ці моделі значно знижують перплексію порівняно з тодішніми моделями

перекладу на основі вирівнювань і показують високу чутливість до порядку слів, синтаксису та значення вихідного тексту.

Sutskever та інші в статті пропонують використання Long Short-Term Memory (LSTM), яка перетворює вхідну послідовність у вектор фіксованої розмірності, а потім інший LSTM декодує цільову послідовність. На завданні перекладу з англійської на французьку модель досягла BLEU-оцінки 34,8. Крім того, модель продемонструвала високу чутливість до порядку слів і добре обробляла довгі речення [11]. Проте найвагомішою інновацією того часу було винайдення та впровадження механізму уваги.

2.4 Механізм уваги

В 2014 році Bahdanau, Cho та Bengio запропонували новий підхід для автоматичного пошуку частини вхідного речення, що стосуються передбачення цільового слова, без необхідності явно формувати ці частини як жорстко обмежені сегменти [13]. Нова архітектура складалася з двонаправленої RNN, як енкодера, та декодера, що імітує пошук через вхідне речення під час декодування перекладу. В новій архітектурі BiRNN складається з прямих та зворотних RNN. Прямий RNN $\vec{f}$ читає вхідну послідовність у заданому порядку (з x_1 до x_{T_x}) та обчислює послідовність прямих прихованих станів ($\vec{h}_1$, ..., $\vec{h}_{T_x}$). Зворотний RNN f зчитує послідовність у зворотному порядку (з x_{T_x} до x_1), що призводить до послідовності зворотних прихованих станів ($h_1, \dots, h_{T_x}$).

В новій архітектурі шару декодера кожна умовна ймовірність в (2.4) визначається як:

$$p(y_i \mid y_1, \dots, y_{i-1}, x) = g(y_{i-1}, s_i, c_i), \tag{2.5}$$

де s_i – прихований стан RNN для моменту часу i, який обчислюється як:

$$s_i = f(s_{i-1}, y_{i-1}, c_i). \quad (2.6)$$

Як видно, на відміну від попередньо описаного підходу з використанням енкодера та декодера, ймовірність обумовлена окремим контекстним вектором c_i для кожного цільового слова y_i. Контекстний вектор c_i залежить від послідовності анотацій $(h_1, \dots, h_{T_x})$ до яких енкодер відображає вхідне речення. Кожна анотація h_i містить інформацію про всю вхідну послідовність із сильним акцентом на частинах, що оточують i-те слово вхідної послідовності. Процес обчислення анотацій наведений вище.

Контекстний вектор c_i обчислюється як зважена сума цих анотацій h_i:

$$c_i = \sum_{j=1}^{T_x} \alpha_{ij} h_j. \quad (2.7)$$

Вага a_{ij} кожної анотації h обчислюється як:

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^{T_x} \exp(e_{ik})}, \quad (2.8)$$

де

$$e_{ij} = a(s_{i-1}, h_j). \quad (2.9)$$

Це модель вирівнювання, яка оцінює, наскільки добре вхідні дані навколо позиції j та вихід на позиції i відповідають одне одному. Оцінка ґрунтується на прихованому стані RNN s_{i-1} (безпосередньо перед виведенням y_i (2.5)) та j-й анотації h_j вхідного речення [13].

Модель вирівнювання параметризується як нейронна мережа прямого поширення, яка спільно навчається з усіма іншими компонентами запропонованої системи. Важливо зазначити, що на відміну від традиційного машинного перекладу, у цьому підході вирівнювання не розглядається як латентна змінна. Замість цього модель вирівнювання

безпосередньо обчислює м’яке вирівнювання, що дає змогу зворотньому поширенню градієнта проходити через модель. Цей градієнт використовується не лише для навчання моделі вирівнювання, але й для спільного навчання всієї системи перекладу.

Підхід, що полягає в обчисленні зваженої суми анотацій, розглядається як обчислення очікуваної анотації, де очікування здійснюється за можливими варіантами вирівнювань. Ймовірність того, що певне цільове слово відповідає або перекладається з конкретного вихідного слова, використовується для визначення контекстного вектора як очікуваної анотації для всіх анотацій з відповідними ймовірностями.

Ймовірність, або її асоційована енергія, відображає важливість кожної анотації в контексті попереднього прихованого стану при обчисленні наступного стану та генерації цільового слова. На рисунку 2.2 зображена графічна ілюстрація раніше описаної архітектури моделі.

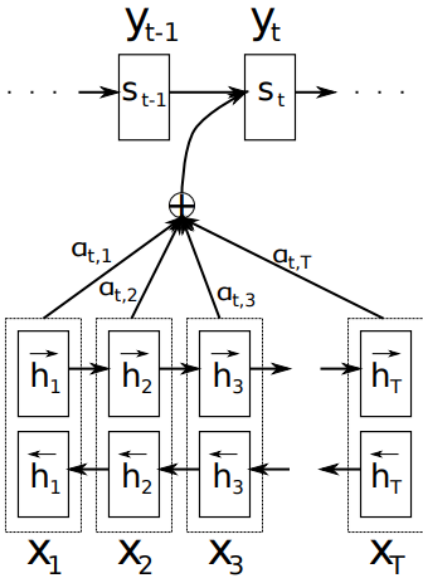

Рисунок 2.2 – Ілюстрація моделі, що намагається згенерувати t-те слово y_t, маючи вхідне речення $(x_1, x_2, \dots, x_T)$

Це реалізує механізм уваги в декодері, що дозволяє вирішувати, на які частини вхідного тексту слід звертати увагу. Такий підхід знімає з енкодера

необхідність передавати всю інформацію з вхідного речення у фіксований вектор, оскільки інформація розподіляється по всій послідовності анотацій, які декодер може вибірково використовувати відповідно до поточних потреб.

Наведена вище модель була випробувана на задачі перекладу з англійської на французьку. Порівнювалась вона з моделлю на основі RNN з енкодером та декодером, запропонованої Cho та іншими [10] і показала вищі показники BLEU в задачі перекладу.

Пізніше Luong та інші запропонують в своїй роботі новий механізм уваги. Ці моделі можна розділити дві основні категорії: глобальні та локальні. Ці категорії відрізняються тим, на які елементи вхідної послідовності фокусується «увага» – на всіх елементах чи тільки на деяких з них.

Спільним для цих двох типів є те, що на кожному етапі декодування, обидва підходи спочатку отримують прихований стан на верхньому шарі LSTM. Метою є отримати контекстний вектор, який містить важливу інформацію з вхідної послідовності, що допомагає передбачити поточне вихідне слово. Хоча ці моделі відрізняються за способом обчислення контекстного вектора, вони мають однакові наступні кроки.

Зокрема, маючи прихований стан на виході та контекстний вектор з вхідної сторони, використовується простий шар об'єднання для комбінування інформації з обох векторів, що дозволяє створити вектор з увагою. Цей вектор потім проходить через шар softmax, щоб отримати прогнозовану розподілену величину для поточного слова.

Ідея глобальної моделі уваги полягає в тому, щоб враховувати всі приховані стани енкодера при обчисленні контекстного вектора. У такій моделі створюється вектор вирівнювання змінної довжини, довжина якого дорівнює кількості кроків часу на стороні джерела, шляхом порівняння поточного прихованого стану на виході з кожним прихованим станом з

вхідної послідовності. Вага кожного елемента вхідної послідовності визначається на основі поточного стану і всіх станів джерела.

Цей процес можна описати через функцію оцінки, яку можна побудувати за кількома варіантами, включаючи використання скалярного добутку між поточним станом і вхідними станами, або ж застосування більш складних методів, таких як використання функції активації.

В рамках ранніх спроб побудови моделей на основі уваги використовували функцію оцінки, засновану лише на поточному прихованому стані, що дозволяло визначати важливість кожного елемента лише на основі цього стану.

Глобальна модель уваги є подібною до моделі, запропонованої Bahdanau, однак є кілька суттєвих відмінностей, що свідчать про те, як ця модель була спрощена та узагальнена. На рисунку 2.3 зображена графічна ілюстрація моделі з глобальною увагою.

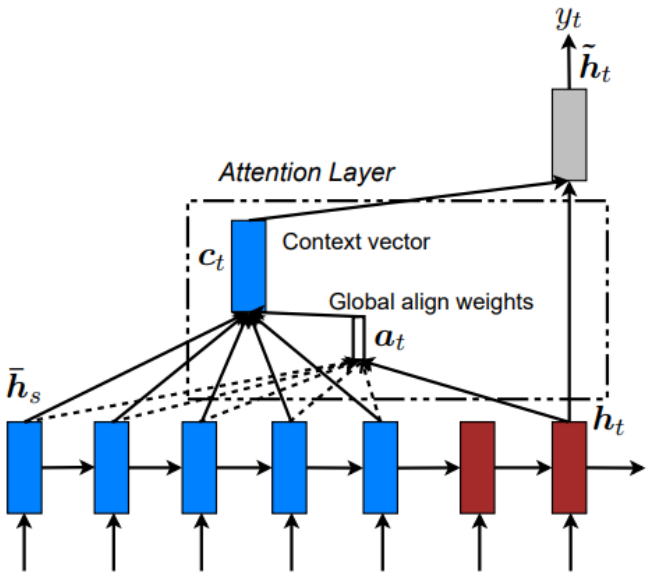

Рисунок 2.3 – Модель з глобальною увагою

По-перше, тут використовуються лише приховані стани на верхніх шарах LSTM як в енкодері, так і в декодері. У свою чергу, Bahdanau

використовує поєднання прямого і зворотного прихованих станів у бінаправленому енкодері та приховані стани в односторонньому декодері.

По-друге, шлях обчислення в моделі Luong’а є простішим: використовується ланцюг від поточного прихованого стану до вектора вирівнювання, потім до контекстного вектора і знову до передбачення. У Bahdanau обчислення складніше, оскільки використовуються попередні приховані стани та додаткові шари перед отриманням результату.

Також Bahdanau експериментував лише з однією функцією вирівнювання, тоді як Luong з іншими дослідниками продемонструють, що інші варіанти функцій можуть бути більш ефективними. Глобальний механізм уваги має недолік, оскільки для кожного цільового слова потрібно звертатися до всіх слів на вхідній стороні, що є витратним і може зробити його непридатним для перекладу довших послідовностей, наприклад, абзаців чи документів. Для вирішення цієї проблеми пропонується механізм локальної уваги, який фокусується лише на невеликій підмножині позицій на вхідній стороні для кожного цільового слова.

Ця модель натхненна компромісом між м'яким і жорстким механізмами уваги, запропонованими Xu та іншими [14] для завдання генерації підписів до зображень. М'яка увага відноситься до глобального механізму, в якому ваги розподіляються «м’яко» по всіх ділянках вхідного зображення. Жорстка увага, з іншого боку, вибирає одну ділянку зображення для обробки за раз. Хоча жорстка увага менш витратна під час передбачення, цей підхід не є диференційованим і вимагає складніших технік, таких як зменшення дисперсії або навчання з підкріпленням.

Механізм локальної уваги Loung’а вибірково фокусується на невеликому вікні контексту і є диференційованим. Такий підхід має перевагу в тому, що він уникає дорогих обчислень, що виникають при використанні м'якої уваги, а також є простішим у навчанні порівняно з жорсткою увагою. У конкретних деталях модель спочатку генерує позицію вирівнювання для кожного цільового слова на етапі декодування. Потім

контекстний вектор обчислюється як зважена середня величина по набору вхідних прихованих станів у межах вікна. На відміну від глобального підходу, вектор вирівнювання в локальній увазі тепер має фіксовану розмірність. На рисунку 2.4 зображена ілюстрація моделі з локальною увагою.

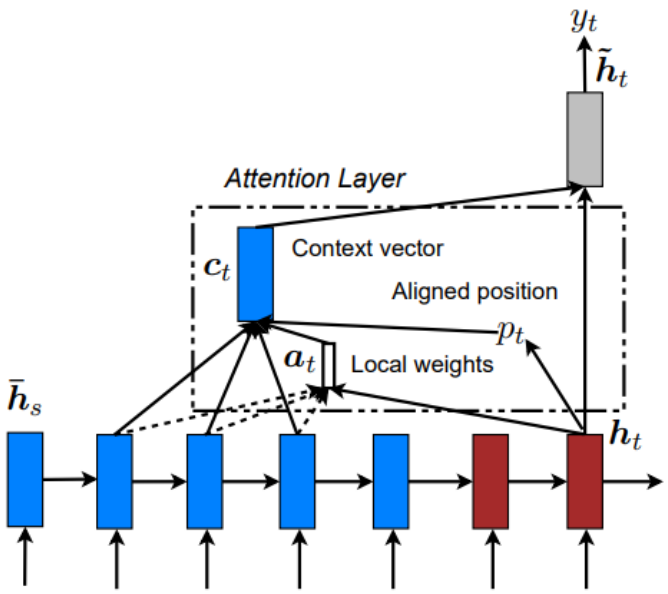

Рисунок 2.4 – Модель з локальною увагою

Розглядається дві варіації цієї моделі. Перша з них – монотонне вирівнювання (local-m), де позиція вирівнювання для кожного цільового слова приймається як просто поточний час t, що припускає, що вхідна і вихідна послідовності є приблизно монотонними. Вектор вирівнювання обчислюється за певною формулою.

Друга варіація – прогностичне вирівнювання (local-p), де замість припущення монотонного вирівнювання модель прогнозує позицію вирівнювання на основі прихованого стану на поточному етапі декодування. Для цього використовуються певні параметри моделі, які навчаються для прогнозування позицій.

Щоб сприяти вирівнюванню точок, розташованих близько до передбаченої позиції, використовується гаусове розподілення з центром на цій позиції. В результаті наші ваги вирівнювання визначаються за допомогою цієї функції.

Розроблення механізму уваги в 2014 році призвело до створення простішого, але набагато потужнішого інструменту для вирішення задач машинного перекладу, генерації тексту, розпізнавання та виправлення граматичних помилок та багато іншого.

2.5 Трансформери

Станом на 2017 рік провідними підходами в сфері моделювання послідовностей та трансдукційних задач, таких як моделювання мов та машинний переклад, були рекурентні нейронні мережі, LSTM та GRU [15]. Численні зусилля багатьох дослідників розширювали межі можливостей рекурентних мовних моделей і архітектур енкодер-декодер. Проте група дослідників з Google запропонували [16] нову просту архітектуру мережі, Трансформер, що базується виключно на механізмах уваги, повністю відмовляючись від рекурентності та згорток.

Енкодер складається з шести однакових шарів. Кожен шар включає дві підшари: багатоголовий механізм самоуваги (Multi-head self-attention mechanism) та повнозв’язну нейронну мережу прямого поширення, що застосовується до кожної позиції. Для збереження інформації на кожному рівні використовуються залишкові з’єднання та нормалізація шару. Всі підшари та вбудовування мають розмірність $d_{model} = 512$.

Декодер також складається з шести шарів. Крім двох підшарів, як в енкодері, додається третій, який виконує багатоголову увагу до виходу енкодера. Самоувага модифікована для запобігання доступу до наступних позицій, забезпечуючи залежність лише від попередніх виходів.

Функція уваги описується як відображення запиту та набору пар «ключ-значення» у вихідний вектор. Запит, ключі, значення та вихід представлені у вигляді векторів. Вихід обчислюється як зважена сума значень, де кожній вазі призначається коефіцієнт, який відображає ступінь важливості кожного значення для запиту.

Механізм часткової уваги в Трансформері називається Масштабований скалярний добуток уваги (Scaled Dot-Product Attention). Вхідними даними є запити (queries) та ключі (keys) розмірності d_k, а також значення (values) розмірності d_v. Ми обчислюємо скалярний добуток запиту з усіма ключами, ділимо кожен результат на $\sqrt{d_k}$ і застосовуємо функцію softmax для отримання ваг значень.

На практиці функцію уваги обчислюють одночасно для набору запитів, зібраних у матрицю Q. Ключі та значення також зібрані в матриці K і V. Матриця вихідних даних обчислюється як:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V. \qquad (2.10)$$

Два найбільш розповсюджений механізми уваги – це адитивна увага [13] та увага за допомогою скалярного добутку (множення). Увага за допомогою скалярного добутку ідентична запропонованому алгоритму, за винятком множення на параметр $\frac{1}{\sqrt{d_k}}$. Адитивна увага обчислює функцію сумісності за допомогою нейронної мережі з одним прихованим шаром. Хоча обидва механізми мають подібну теоретичну складність, увага з добутком значно швидша та ефективніша в практиці через оптимізоване множення матриць. Для малих значень d_k обидва механізми працюють подібно, але адитивна увага перевершує увагу за допомогою скалярного добутку при більших значеннях d_k. Дослідники припускають, що для великих значень d_k добутки стають занадто великими, що призводить до того, що функція softmax потрапляє в область з надзвичайно малими

градієнтами. Щоб компенсувати цей ефект, добутки множаться на $\frac{1}{\sqrt{d_k}}$. На рисунку 2.5 (ліворуч) графічно зображене представлення масштабованого скалярного добутку.

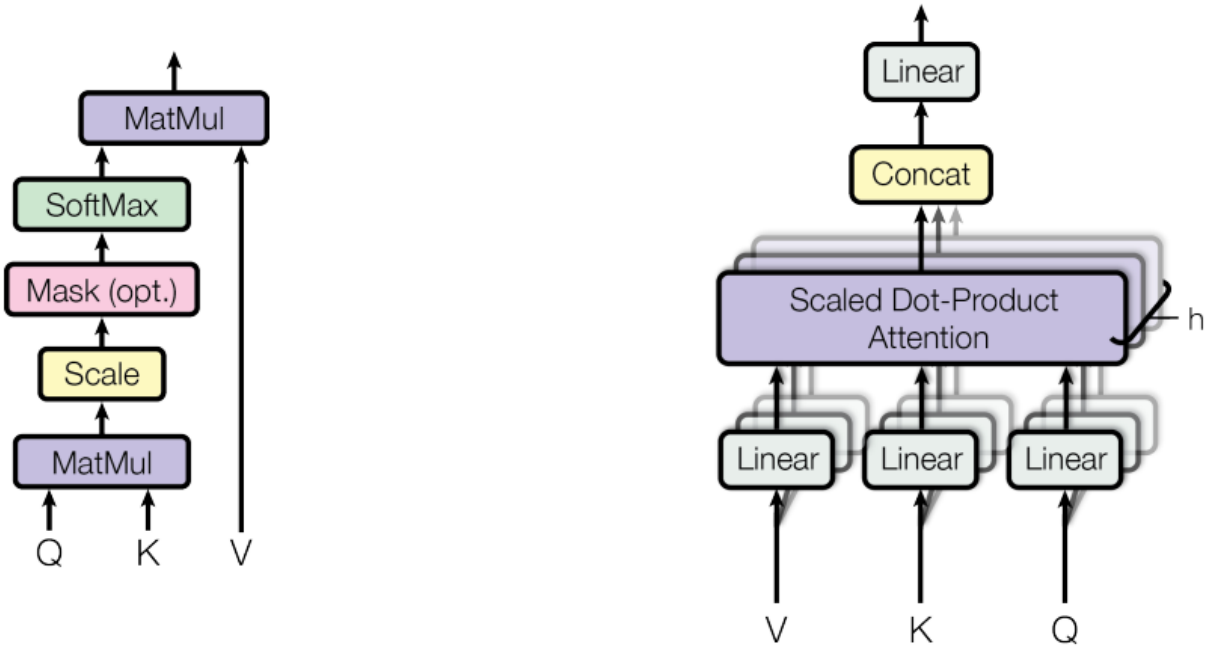

Рисунок 2.5 – Масштабований скалярний добуток уваги (ліворуч), багатоголовий механізм уваги (праворуч)

Багатоголовий механізм уваги дозволяє моделі одночасно звертати увагу на інформацію з різних підпросторів представлень на різних позиціях. У випадку з єдиною головою уваги, середнє значення гальмує цей процес. У формулах 2.11 та 2.12 представлений принцип роботи цього механізму, а на рисунку 2.X (праворуч) зображене графічне представлення.

$$\text{MultiHead}(Q,K,V) = \text{Concat}(\text{head}_1,\dots,\text{head}_h)W^O, \qquad (2.11)$$

де $\text{head}_i = \text{Attention}\left(QW_i^Q, KW_i^K, VW_i^V\right)$.

Модель Трансформер використовує механізм уваги трьома різними способами. Перший спосіб полягає в тому, що шарах уваги енкодера-

декодера запити надходять з попереднього шару декодера, а ключі та значення – з виходу енкодера, дозволяючи кожній позиції в декодері звертатися до всіх позицій у вхідній послідовності. Другий же спосіб полягає в тому, що енкодер містить шари самоуваги, де всі ключі, значення та запити йдуть з виходу попереднього шару енкодера. По-третє, шари самоуваги в декодері дозволяють кожній позиції в декодері звертатися до всіх позицій у декодері до і включно з поточною. Для збереження автогенеративної властивості необхідно запобігти потоку інформації вліво. Це реалізується в масштабованому механізмі уваги за допомогою маскування всіх значень у вході softmax, що відповідають незаконним зв'язкам.

Окрім підшарів уваги, кожен шар енкодера та декодера містить повнозв'язну нейронну мережу прямого поширення, яка застосовується до кожної позиції окремо та однаково. Вона складається з двох лінійних перетворень з активацією ReLU між ними. Лінійні перетворення однакові для різних позицій, але використовують різні параметри на кожному шарі. Це можна також описати як дві згортки з розміром ядра 1. Вхідна та вихідна розмірність становить 512, а внутрішній шар має розмірність 2048.

Отже, згідно з раніше викладеним описом механізм самоуваги має кілька переваг порівняно з рекурентними та згортковими шарами. По-перше, він забезпечує меншу обчислювальну складність, особливо коли довжина послідовності менша за розмірність представлення. Самоувага дозволяє кращу паралелізацію, вимагаючи меншої кількості послідовних операцій. Також вона добре справляється з довгостроковими залежностями, оскільки з’єднує всі позиції через сталу кількість операцій. А також самоувага може забезпечити більш інтерпретовані моделі, оскільки увага часто відображає синтаксичні та семантичні структури. На рисунку 2.6 зображена загальна архітектура моделі Трансформер.

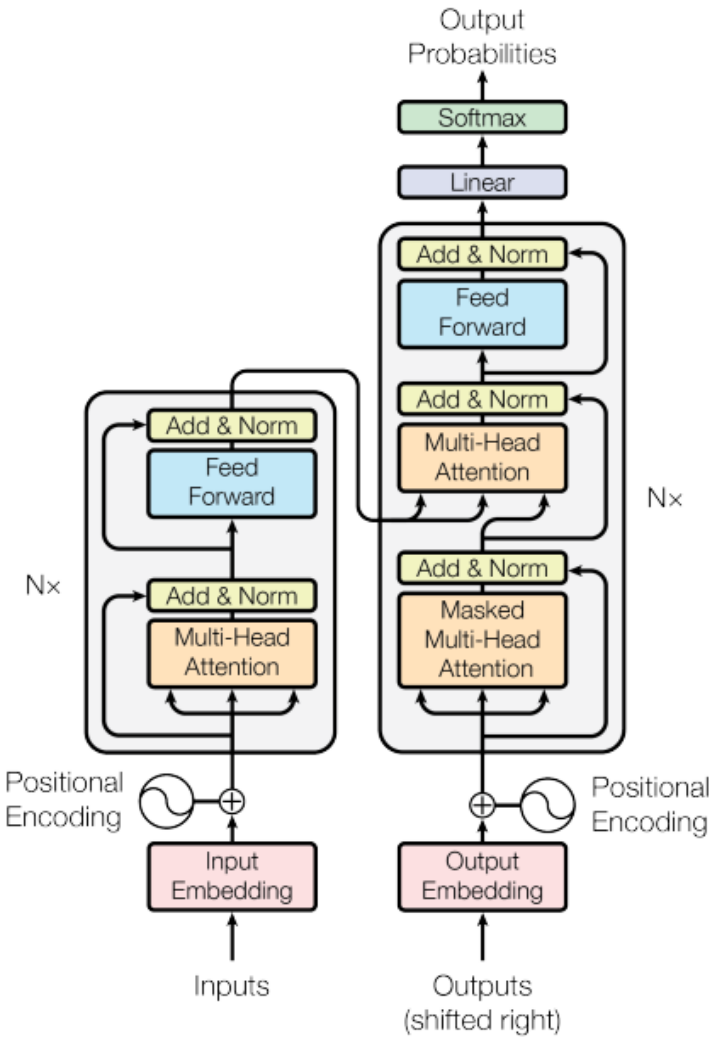

Рисунок 2.6 – Трансформер, архітектура моделі

Винайдення архітектури Трансформер стало основою для створення серйозніших проєктів та архітектурних рішень, які докорінно змінили сферу вирішення задач в галузі обробки природної мови. В цьому проєкті для вирішення задачі граматичної корекції текстів використовуватимуться покращені моделі на основі архітектури Трансформер. Конкретні моделі, їхні особливості, переваги та недоліки будуть наведені в наступних розділах цієї роботи.

3 СТВОРЕННЯ СИНТЕТИЧНИХ ДАНИХ

3.1 Корпуси даних

За останнє десятиліття дослідники в галузі обробки природної мови зосереджували свою уваги переважно на задачах обробки англійської мови, і досягли суттєвого прогресу. На CoNLL-2014 shared task показник $F_{0.5}$ на найкращих моделях спрогресував з 37.33 у 2014 році [7] до 72.8 у 2024 році, використовуючи ансамбль з 7 моделей [17]. Достатня кількість датасетів та навчальних даних стала вирішальним фактором, що призвів до такого успіху. Проте моделі, розроблені з урахуванням англійської мови, є менш ефективними для морфологічно багатих мов. Загальною проблемою є нестача даних – особливо високоякісних анотованих даних, які могли б бути використані для оцінювання та тонкого налаштування моделей.

У 2023 році був представлений набір даних для задачі корекції граматичних помилок UA-GEC [5]. Корпус текстів було створено на основі матеріалів, зібраних від різноманітної групи авторів, серед яких були як носії мови, так і ті, для кого вона не є рідною. Він охоплює широкий спектр доменів, таких як есе, публікації в соціальних мережах, чати, офіційні тексти та інші. Для виправлення помилок, пов’язаних із граматикою, орфографією, пунктуацією та плавністю тексту, залучалися професійні коректори. Хоч темою роботи є розроблення системи з виправлення граматичних та синтаксичних помилок, технічно тут досліджується питання вирішення задачі grammatical error correction, що включає в себе й інші типи помилок, такі як морфологічні, синтаксичні, лексичні, орфографічні, граматичні та пунктуаційні. Важливо зазначити, що і в датасеті UA-GEC, і в датасеті CoNLL-2014 найбільш частка помилок припадає на пунктуаційні помилки. Датасет UA-GEC містить 31046 речень, з котрих лише 14344, містять більше однієї помилки, тобто більше половини речень в цьому

датасеті займають граматично правильні речення. На рисунку 3.1 зображений розподіл помилок в датасеті.

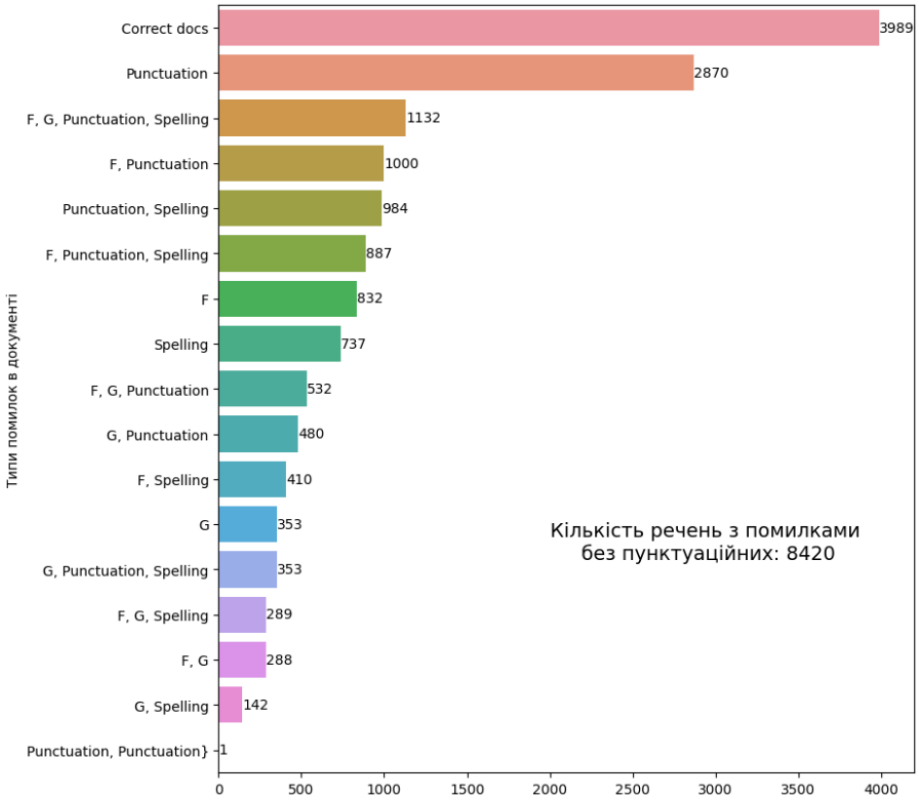

Рисунок 3.1 – Розподіл помилок в датасеті

Як згадувалось раніше, найкращий результат станом на зараз для задачі корекції граматичних помилок в англійській мові є показник $F_{0.5}$ 72.8 представлений групою розробників з Grammarly. У висновках до своєї роботи [17] автори припускають, що головним обмеженням у досягненні вищих показників оцінки є нестача даних високої якості, а не архітектура чи розмір використаних моделей. Зважаючи на те, що їхні моделі навчалися на значно більших корпусах даних для англійської мови, можна зробити висновок, що даних з корпусу української мови катастрофічно не вистачатиме для досягнення прийнятних результатів. У наступних підрозділах третього розділу роботи будуть описані методи для створення якісних синтетичних даних для розширення наявного датасету.

3.2 Створення синтетичних даних

Проблема нестачі даних стимулювала активні дослідження у сфері генерації синтетичних даних, особливо в контексті ресурсомістких підходів на основі нейронного машинного перекладу (NMT). Синтетичні дані вимагають лише корпусу текстів рідною мовою, без необхідності трудомісткого процесу ручної анотації. Відкритим питанням залишається те, як оцінювати моделі та методи для створення синтетичних даних. Більшість досліджень оцінюють їх побічно – через вплив на результати попередніх досліджень. Незважаючи на це, розширення даних значно сприяло покращенню систем GEC і стало невід'ємною частиною сучасних моделей.

3.2.1 Ін'єкція шуму

Один із способів штучного створення граматичних помилок у чистих монолінгвальних корпусах полягає у зміні правильного тексту, щоб зробити його граматично неправильним. Такі зміни можуть виконуватися у вигляді шумових операцій на основі правил або шаблонів помилок, які зазвичай зустрічаються в паралельних корпусах для задач GEC [18].

Найбільш очевидний спосіб додавання шуму до чистого корпусу полягає у виконанні серії змін, заснованих на заздалегідь визначених правилах. Ці правила застосовуються з певною ймовірністю, яка може бути визначена довільно, емпірично або на основі аналізу доступних даних.

Можна додавати одну помилку до кожного речення, використовуючи попередньо створені шаблони, які включають пропуски прийменників, повторення слів тощо. Lichtarge та інші [19] вносять орфографічні помилки до історії редагувань Wikipedia шляхом видалення, вставки, заміни та перестановки символів. Пізніше були застосування схожої стратегії, але на

рівні слів, видаляючи, додаючи, переставляючи або замінюючи слова в реченнях.

Grundkiewicz та інші [20] об’єднують підходи на рівні символів та слів, обмежуючи заміну слів парами з набору помилково схожих слів, створеного на основі перевірки орфографії. Метод додавання шуму на основі правил може використовуватися й динамічно під час навчання, щоб збільшити частоту помилок у паралельному корпусі без необхідності створювати додаткові навчальні дані.

Ще одним способом генерації синтетичних даних є внесення помилок, які часто трапляються в паралельних корпусах для задачі GEC. Такий підхід дозволяє отримати помилки, що є більш схожими на ті, які зазвичай роблять люди.

Rozovskaya та Roth [21] запропонували три різні методи внесення помилок у використанні артиклів, засновані на розподілі помилок у даних для вивчення англійської як другої мови. Вони запропонували додавати помилки, базуючись на:

– розподілі артиклів у тексті до виправлення;
– розподілі артиклів у виправленому тексті;
– розподілі самих виправлень, пов’язаних з артиклями.

Пізніше метод був вдосконалений, враховуючи морфологію, частини мови (POS), семантичні концепти та значення слів під час генерації штучних помилок.

Іншим напрямом імітації людських помилок є вилучення шаблонів виправлень із паралельних корпусів для GEC та застосування інверсії цих виправлень до граматично правильних речень. Шаблони виправлень вилучаються як у лексичній формі (наприклад, заміна an на the), так і за частинами мови (наприклад, зміна іменників в однині на іменники в множині) [18].

3.2.2 Зворотний переклад

Імітація людських помилок може бути виконана більш автоматизованим і динамічним способом за допомогою моделі «шумового каналу». Ця модель тренується на інверсії паралельного корпусу для виправлення граматичних помилок (GEC), де речення з помилками розглядається як ціль, а правильне речення – як джерело. Ця техніка зазвичай називається зворотним перекладом (back-translation).

Спочатку метод був запропонований для генерації додаткових даних у задачах машинного перекладу, але його також можна безпосередньо застосовувати для GEC. Пізніше був доданий механізм контролю якості до існуючого підходу, використовуючи ймовірності мовної моделі, щоб гарантувати, що синтетично згенеровані речення є менш ймовірними (а отже, менш граматично правильними), ніж вихідні речення.

Порівнюючи стратегії на основі правил і зворотного перекладу, дослідники повідомили [22], що зворотний переклад забезпечує кращі емпіричні результати. Вони також порівняли зворотний переклад із «шумовим» методом пошуку променем (noisy beam-search) та зворотний переклад із використанням стратегії вибірки (sampling), і повідомили, що обидва підходи досягають конкурентоспроможних результатів.

У 2021 група дослідників додатково з’ясовували вплив використання різних архітектур (CNN, LSTM, Transformer) для зворотного перекладу та виявили, що поєднання результатів кількох генераційних систем зазвичай дозволяє створювати кращі синтетичні дані для навчання систем GEC.

Ще один варіант зворотного перекладу запропонували Stahlberg та Kumar у своїй роботі [23] для генерації більш складних змін. Вони виявили, що створення послідовності виправлень за допомогою моделі Seq2Edit працює краще, ніж безпосереднє генерування речень із помилками. Також вони повідомили, що зворотний переклад із використанням стратегії

вибірки показав кращі результати, ніж пошук променем, у їхніх експериментах.

3.2.3 Round-trip translation

Менш поширеною альтернативою методу зворотного перекладу є метод перекладу через зворотний шлях (round-trip translation), який генерує синтетичні пари речень через проміжну мову (наприклад, англійську-китайську-англійську). Припущення полягає в тому, що система машинного перекладу буде робити помилки при перекладі, і тому результат через проміжну мову буде містити шуми порівняно з оригінальним текстом. Також досліджувався ефект використання різних проміжних мов. Zhou та інші [24] застосовують схожу техніку, але використовують проміжну мову як вхід для обох систем перекладу – низькоякісної та високоякісної (тобто, SMT та NMT), розглядаючи результат від першої як граматично некоректне зашумлене речення, а результат від другої як еталонне правильне речення.

3.3 Метрики оцінки

Одним із ключових компонентів будь-якої системи обробки природної мови є здатність оцінювати продуктивність моделі. У цьому розділі спочатку розглянуті найпоширеніші метрики оцінювання в завданнях корекції граматичних помилок, такі як MaxMatch (M^2) scorer, ERRANT та GLEU.

3.3.1 MaxMatch

Один із найбільш поширених методів оцінки, який використовується в сучасних дослідженнях з корекції граматичних помилок, це метрика MaxMatch (M^2), яка обчислює F_β-оцінку. Зокрема, оцінювач M^2 є метрикою,

заснованою на порівнянні з еталонами, яка порівнює виправлення гіпотез системи з людськими еталонними виправленнями і підраховує кількість істинно позитивних (TP), якщо виправлення гіпотези співпадає з еталонним виправленням, хибно позитивних (FP), якщо виправлення гіпотези не співпадає з жодним еталонним виправленням, і хибно негативних (FN), якщо еталонне виправлення не співпадає з жодним виправленням гіпотези.

Загальна кількість TP, FP і FN для набору даних може бути використана для обчислення точності (Precision) і повноти (Recall), які відповідно показують частку правильних виправлень гіпотези та частку еталонних виправлень, які були знайдені в виправленнях гіпотези. Це, в свою чергу, може бути використано для обчислення F_β-оцінки. У сучасних дослідженнях GEC зазвичай використовують $\beta = 0.5$, що дає точності вдвічі більше значення, ніж повноті, оскільки зазвичай вважається, що для системи GEC важливіше бути точною, ніж виправляти всі помилки [18].

3.3.2 ERRANT

Метрика ERRANT [25] подібна до метрики M^2, оскільки вона також заснована на порівнянні з еталонами, і вимірює ефективність за допомогою F-оцінки, орієнтуючись на виправлення. Однак основна відмінність полягає в тому, що ERRANT також може обчислювати оцінки для типів помилок. На відміну від M^2, цей оцінювач використовує лінгвістично вдосконалений алгоритм вирівнювання Дамерау-Левенштейна для витягнення виправлень з тексту гіпотези і класифікує їх відповідно до правила для визначення типів помилок. Це дозволяє обчислювати F-оцінки для кожного типу помилки окремо, а не тільки для загальної ефективності, що є дуже корисним для детального аналізу системи [18]. Наприклад, Система A може перевершувати Систему B за загальними результатами, але Система B може бути кращою за Систему A в корекції певних типів помилок, і ця інформація може бути використана для покращення Системи A.

3.3.3 GLEU

Як M^2 та ERRANT, GLEU є метрикою, заснованою на порівнянні з еталонами, однак вона не потребує явних анотацій виправлень, а лише виправлених еталонних речень. Вона була натхнена оцінкою BLEU, яка широко використовується в машинному перекладі, і була мотивована тим, що анотації виправлень, створені людьми, є дещо умовними та забирають багато часу на збори. Основна ідея GLEU полягає в тому, щоб надавати перевагу n-грамам гіпотези, які перекриваються з еталоном, але не з оригінальним текстом, і карати n-грами гіпотези, які перекриваються з оригінальним текстом, але не з еталоном. Вона враховує як правильні виправлення, так і уникнення непотрібних змін, що важливо для задач GEC.

3.3.4 Найкращі практики

Звичною помилкою для нових дослідників у галузі GEC є питання, яку метрику використовувати з яким набором даних; наприклад, оцінювач M2 для JFLEG або I-міра для BEA-2019. Хоча немає емпіричних підстав віддавати перевагу одній метриці перед іншою, на практиці найбільш популярні тестові набори GEC майже завжди оцінюються за допомогою однієї конкретної метрики:

– CoNLL-2014 оцінюється метрикою M^2;
– JFLEG оцінюється метрикою GLEU;
– BEA-2019 оцінюється метрикою ERRANT.

Цей вибір обумовлений переважно історичними причинами (наприклад, GLEU та ERRANT не існували під час CoNLL-2014), але він тим не менш зберігся для забезпечення справедливої порівнянності з подальшими роботами. Однією з особливо поширених помилок є оцінювання CoNLL-2014 за допомогою ERRANT або BEA-2019 за допомогою оцінювача M^2, оскільки обидві метрики повертають F-оцінку,

однак M^2 $F_{0.5}$ не є еквівалентом ERRANT $F_{0.5}$. Тому важливо, щоб набір даних оцінювався за допомогою відповідної метрики для забезпечення змістовного порівняння.

3.4 Існуючі практичні рішення

У 2023 році пройшло змагання UNLP-2023 [26], задача якого було розробити систему виправлення граматичних помилок для української мови. Вхідними даними був вищезгаданий корпус UA-GEC, проте автори змагання заохочували використовувати сторонні навчальні дані. Дані поділялись на навчальну та валідаційну вибірки. Валідаційна використовувалася для того, щоб обрати модель для подання на змагання. Проте результати самого змагання оцінювались на прихованому тестовому датасеті з використанням метрики $F_{0.5}$. Змагання складалось з двох завдань, перше GEC-Only, що обмежувалось внесенням мінімального набору виправлень граматики, орфографії, пунктуації. Друге завдання GEC+Fluency дозволяло проводити більші переписування речень, щоб зробити текст більш плавним – тобто таким, що звучить природно для носія мови. Корекція плавності є більш суб'єктивною і може бути складнішою для прогнозування. Задача GEC+Fluency охоплює виправлення граматики, орфографії, пунктуації та плавності. В таблиці 3.1 зображені результати цього змагання для GEC-Only та в таблиці 3.2 – результати GEC+Fluency.

Таблиця 3.1 – Результати завдання GEC-Only

№	Учасник	TP	FP	FN	Precision	Recall	$F_{0.5}$
1	QC-NLP (fpg)	636	192	400	76.81	61.39	73.14
2	UA-GEC	508	139	496	78.52	50.6	70.71
3	QC-NLP	661	253	386	72.32	63.13	70.27
4	WebSpellChecker	458	170	502	72.93	47.71	65.96

Таблиця 3.2 – Результати завдання GEC+Fluency

№	Учасник	TP	FP	FN	Precision	Recall	$F_{0.5}$
1	Pravopysnyk	580	153	742	79.13	43.87	68.17
2	QC-NLP (fpg)	735	269	646	73.21	53.22	68.09
3	WebSpellChecker	528	125	759	80.86	41.03	67.71
4	GrammarUA	526	138	776	79.22	40.40	66.45
5	QC-NLP	739	318	635	69.91	53.78	65.96
6	UA-GEC	594	219	745	73.06	44.36	64.69

Далі будуть розібрані підходи та методи, що використовувалися учасниками цього змагання.

3.4.1 RedPenNet

Команда WebSpellChecker використовувала власну архітектуру, подібну до трансформера, під назвою RedPenNet. Ця архітектура використовує попередньо навчений енкодер MLM разом з неглибоким декодером для генерації як токенів заміщення, так і ділянок для редагування в задачах GEC. Під час генерації токенів редагувань вага уваги між енкодером і декодером визначає ділянки редагування (початок і кінець), які вказують на місце редагування в початковому реченні. Токени редагування передбачаються автогресивно. Токени SEP відокремлюють редагування в вихідній послідовності. На кожному кроці зворотного зв'язку вбудовування токена редагування BPE поєднується з навчальним вбудовуванням позиційного кодування, специфічним для декодера. Отриману суму потім конкатенують з вбудовуванням ділянки. Крім того, компактні словники BPE для задачі GEC тренуються для зменшення вартості операції до softmax, що підвищує ефективність передбачення токенів заміщення.

Основною перевагою RedPenNet є здатність реалізувати будь-яку трансформацію від джерела до цілі за допомогою мінімальної кількості

автогресивних кроків, що дозволяє ефективно вирішувати задачі GEC, включаючи взаємопов'язані та багатотокенові редагування. Однак через спеціалізовану архітектуру RedPenNet немає готових рішень для попередньої обробки даних, тренування чи тонкої настройки. Тому зручні інструменти, такі як інфраструктура HuggingFace, не можуть бути використані для швидкої тонкої настройки і розгортання моделей.

3.4.2 QC-NLP

QC-NLP, переможці завдання для GEC-Only та друге місце в завданні GEC+Fluency, подали дві системи: (1) fpg та (2) rozovska. Обидві системи брали участь в обох треках спільного завдання. Система (1) показала кращі результати в обох треках порівняно з системою (2), але система (1) вимагає більших обчислювальних ресурсів.

У системі (1) автори провели донавчання попередньо навченого mT5-large, щоб виправляти граматично неправильні речення до їх правильних аналогів. Спочатку вони донавчали модель на 10 мільйонах синтетичних даних для виправлення граматичних помилок протягом трьох епох, а потім використовували набір даних із завдання для додаткових 10 епох. Синтетичні приклади були згенеровані за допомогою підходу, заснованого на наборах сплутувань Aspell, запропонованих у цій роботі [27] Цей метод був застосований до рідних українських даних з корпусу WNT News Crawl. До навчання на синтетичних та даних учнів було залучено 8 графічних процесорів Nvidia 80GB, що зайняло приблизно 16 годин загального часу.

Система (2) є трансформерною моделлю, що була попередньо навчена на 35 мільйонах синтетичних прикладів, що використовують сплутування Aspell та додатковий шум від зворотного перекладу, а також донавчену на найкращих даних для навчання. Три моделі були навчені з трьома різними початковими умовами, і кінцева модель є ансамблем трьох найкращих чекпоїнтів. Попереднє навчання на одному графічному процесорі

Nvidia 32GB займало 7 годин на епоху протягом приблизно 10 епох до досягнення збіжності. Донавчання моделі займала приблизно годину до збіжності.

3.4.3 Pravopysnyk

Команда Pravopysnyk [28], переможці завдання GEC+Fluency, об'єднали модель на основі трансформерів із системою виправлення орфографії на основі правил. Для трансформерної моделі вони донавчили mBART [29] на наборі даних UA-GEC, доповненому синтетично створеними помилками. Для генерації додаткових даних команда використала круговий переклад, спеціальний скрипт для створення пунктуаційних помилок, а також заміну українських слів їх русифікованими версіями.

Для виправлення орфографії команда застосувала алгоритм SymSpell, адаптований для української мови. Цей алгоритм використовує словник частот слів і словник частот біграм, побудований на основі 500 тисяч речень, зібраних з українських книг. Найбільш частотне слово, яке відповідає критеріям орфографії, обирається як результат.

Трансформерна модель відповідала за більшість виправлень. Переваги системи включають високу продуктивність, низькі витрати на навчання (навчання займає 10 хвилин на Google Colab з GPU A100) і можливість інтеграції різних джерел синтетичних даних у end-2-end процес навчання. Однак система працює повільніше в порівнянні з моделями seq2tag.

3.4.4 LLaMA

Це змагання проводилось у 2023 році, і в завданні GEC+Fluency виграла команда Pravopysnyk. Проте у 2024 році була опублікована

робота [30], в якій досліджувалось використання моделей LLaMA для вирішення задачі GEC та задачі AEG (Artificial Error Generation) для естонської, німецької та української мов. Основною метою роботи є застосування генеративних мовних моделей до задачі генерації помилок (AEG) через донавчання. Крім того, дослідники експериментують з підходом, що використовує запити до великих мовних моделей (LLM), щоб виконати ту ж саму задачу, а також включають дві моделі seq2seq, до яких застосовують аналогічне донавчання. Ефективність запропонованих рішень для генерації помилок оцінюється через їх використання для покращення моделі виправлення граматичних помилок (GEC). Таким чином, в роботі також донавчаються генеративні LLM для виконання завдання GEC і порівнюються результати з тими, що отримані за допомогою запитів, а також з результатами інших суміжних досліджень.

Загальний процес їхнього підходу є простим і складається з кількох етапів. На першому етапі LLM донавчається для генерації помилок, використовуючи дані людських помилок, де правильні речення є вхідними, а речення з помилками – вихідними. Далі, на другому етапі, застосовується ця модель генерації помилок для виправлення речень, додаючи синтетичні помилкові аналоги. На третьому етапі інша LLM донавчається на цьому створеному синтетичному наборі даних для виправлення граматичних помилок, тепер на вхід подаються неправильні речення, а як вихідні позначаються правильні. Четвертий етап передбачає продовження донавчання моделі GEC на меншому наборі даних з людськими помилками. Останнім етапом є застосування моделей до помилкових речень з тестових наборів даних для оцінки результатів. Всього використовувалось два способи генерації синтетичних даних: підхід заснований зворотному ймовірнісному виправнику помилок (probabilistic reverse-speller) та зворотний переклад, описаний раніше. В таблиці 3.3 наведені результати попереднього навчання та доналаштування моделей на основі LLaMA.

Таблиця 3.3 – Порівняння моделей LLaMA

Метод	TP	FP	FN	Precision	Recall	$F_{0.5}$
LLaMA + gold	–	–	–	79.98	51.76	72.12
LLaMA + prob + gold	–	–	–	80.37	53.19	72.92
LLaMA + BT + gold	712	156	621	82.03	53.41	74.09

Найкращий результат показала модель, що була на першому етапі налаштована на синтетичних даних, згенерованих за допомогою зворотного перекладу, а потім доналаштована на тренувальних даних з датасету UA-GEC (позначені gold). Модель була навчена на одному мільйоні штучно згенерованих речень, що набагато більше, ніж використала команда Pravopysnyk. Проте і результати набагато кращі показник $F_{0.5}$ 74.09 в порівнянні з 68.17. Крім того, підхід з використанням LLaMA перевершив навіть найкращий результат в завданні GEC-Only, незважаючи на те, що модель подавалася лише в категорію GEC+Fluency.

4 ЕКСПЕРИМЕНТАЛЬНІ ДОСЛІДЖЕННЯ

4.1 Вибір програмних засобів

Для розробки системи для вирішення задачі виправлення граматичних помилок із застосуванням генерації штучних помилок були обрані сучасні програмні засоби, такі як PyTorch, бібліотека Transformers від Hugging Face, pymorphy2 та обчислювальна платформа Google Colab. Ці інструменти забезпечують надійну і зручну основу для навчання та тестування моделей, що відповідає завданням, поставленим у цій роботі.

PyTorch є потужною платформою для розробки нейронних мереж, яка відома своєю гнучкістю, широкими можливостями налаштування та підтримкою апаратного прискорення, такого як GPU і TPU. Завдяки цьому стає можливим ефективне навчання великих мовних моделей навіть на значних обсягах даних. Бібліотека Transformers від Hugging Face доповнює можливості PyTorch, надаючи зручні інструменти для інтеграції передових мовних моделей, таких як mBART чи LLaMA. Вона значно спрощує процес тонкого налаштування та дозволяє реалізувати складні алгоритми з мінімальними зусиллями.

Бібліотека pymorphy2 була обрана для аналізу та морфологічної обробки текстів, що є важливим етапом у виправленні граматичних помилок. Цей інструмент надає розширені можливості для розпізнавання та перетворення форм слів у різних мовах, що особливо корисно при роботі з українськими текстами.

Google Colab обрано як основне середовище для проведення експериментів завдяки доступу до потужних обчислювальних ресурсів, зручності роботи з кодом і інтеграції з іншими інструментами. Це дозволяє ефективно навчати моделі, проводити оцінювання їхньої роботи та вносити необхідні корективи. У сукупності ці програмні засоби забезпечують

оптимальні умови для досягнення цілей дослідження та отримання якісних результатів.

4.2 Генерація пунктуаційних помилок

Як було вказано раніше, найбільш розповсюджений тип помилок в англійському та українському корпусах даних – пунктуаційні помилки. Найпоширенішою пунктуаційною помилкою залишається відсутність розділового знаку в місці, де він потрібен (розділення частин складносурядного чи складнопідрядного речень). Тому для створення такого роду помилки варто лише прибрати кому. Існують і складніші випадки, зокрема наявність знаку пунктуації в місці, де його не передбачено. Першим, що спадає на думку, є додавання розділового знаку у випадкове місце в реченні. Проте таке штучно-згенероване речення не можна назвати якісним, бо це взагалі не схоже на помилку, що характерна для людини. Люди роблять пунктуаційні помилки в суперечливих неочевидних місцях, наприклад при відокремленні дієприкметникового чи дієприслівникового зворотів, відокремлення порівнянь в деяких випадках та інше. Якщо взяти за основу дослідження питання з боку виявлення таких структур у реченні, виявиться, що це виходить за межі цього дослідження. Варто не забувати, що кінцева мета – вирішення задачі корекції пунктуаційних помилок, а не їх якісна генерація. Очікується, що пізніше обрані моделі будуть добре справлятися з виправленням пунктуаційних похибок, навіть попри ці «неякісні» дані.

Отже, як було наведено вище, найпростіший спосіб для штучної генерації пунктуаційних помилок в корпусі коректних текстових даних– це заміна певного розділового знаку на інший з деякою ймовірністю. Для цього створена матриця підстановки розділових знаків (зображена на рисунку 4.1), де рядки позначають вхідний розділовий знак, стовпці позначають новий розділовий знак, а поля позначають ймовірність заміни.

		,	;	:	—	-	.	?	!	…
	0.950000	0.033333	0.003333	0.003333	0.006667	0.003333	0.000000	0.000000	0.000000	0.0
,	0.800000	0.198214	0.001786	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.0
;	0.800000	0.085106	0.110638	0.000000	0.004255	0.000000	0.000000	0.000000	0.000000	0.0
:	0.858846	0.050000	0.000000	0.048426	0.042729	0.000000	0.000000	0.000000	0.000000	0.0
—	0.871132	0.050000	0.000597	0.002987	0.075283	0.000000	0.000000	0.000000	0.000000	0.0
-	0.800000	0.000000	0.000000	0.000000	0.025000	0.175000	0.000000	0.000000	0.000000	0.0
.	0.159982	0.000000	0.000000	0.000000	0.000000	0.000000	0.839903	0.000065	0.000050	0.0
?	0.800000	0.000000	0.000000	0.000000	0.000000	0.000000	0.041667	0.116667	0.041667	0.0
!	0.800000	0.000000	0.000000	0.000000	0.000000	0.000000	0.100000	0.100000	0.000000	0.0
…	0.800000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.2

Рисунок 4.1 – Матриця підстановки розділових знаків

Варто зазначити, що задля такого використання матриці підстановки пробіл вважається розділовим знаком. Цей підхід забезпечує додавання нових розділових знаків на місця, де їх не повинно бути, а також прибирання ком, тире, двокрапок, замінюючи їх на пробіли. В роботі використовуються не тільки внутрішньо реченнєві знаки , а й кінцеві. Основна мета полягає в тому, щоб навчити модель розуміти, якого типу речення вона обробляє – питальне, окличне, розповідне. Перед обробкою кожне речення очищується від зайвих пробілів, токенізується та на кожний токен, що є розділовим знаком (включно з пробілом) застосовується матриця підстановки.

4.3 Генерація граматичних помилок

Ще один тип помилок, що буде виправлятись в цій роботі, граматичні помилки. Граматичні помилки – це порушення правил граматики мови, які стосуються неправильного вживання слів, їх форм, а також узгодження між ними. Вони можуть виникати на рівні слів, словосполучень або речень і впливають на правильність висловлювання. Граматичні помилки застосовуватимуться до іменників, прикметників та дієслів. Суть

полягатиме в тому, щоб змінювати граматичні характеристики окремих слів в реченні. Для визначення цих характеристик була обрана бібліотека pymorphy2. Цей інструмент надає змогу визначити частину мови, рід, число, відмінок, час, вид тощо. Такий підхід забезпечує створення високоякісного корпусу даних із граматичними помилками, що є важливим для тренування та тестування моделей виправлення граматичних помилок. На рисунку 4.2 зображений приклад створення граматично некоректного речення.

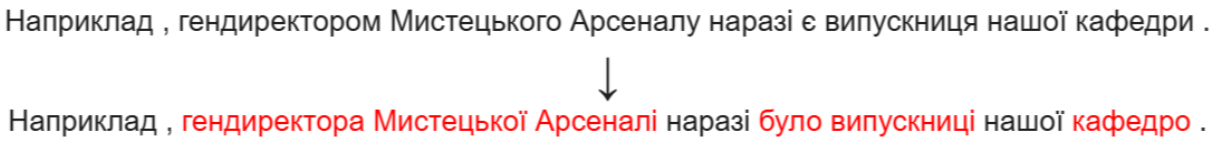

Рисунок 4.2 – Створення граматичних помилок

4.4 Генерація лексичних помилок

Лексична помилка – це неправильне вживання слів у мовленні або тексті, яке порушує правила літературної мови через недоречність або невідповідність контексту, значення чи стилістичних норм. Такі помилки можуть впливати на зрозумілість висловлювання, змінювати його зміст або створювати неправильне враження про мовця чи автора тексту. Зокрема, до лексичних помилок можна віднести вживання слів з неправильним значенням, недоречне використання іншомовних запозичень або впровадження елементів, характерних для інших мовних систем.

4.4.1 Генерація русизмів

У своїй роботі команда Pravopysnyk використовує генерацію русизмів та суржика [28]. Русизм – це слово, якого не існує в українській мові, було взяте з російської та транслітероване на український лад. Суржик, у контексті дослідження, визначено як речення, що формувались через

круговий переклад за допомогою некоректного токенізатора. У результаті такого перекладу утворювались тексти, в яких слова українською мовою змішувались із словами російською. Зокрема, російські слова залишались без транслітерації, тобто писались у їхньому оригінальному вигляді.

У рамках експериментальної роботи було вирішено використати підхід, аналогічний до генерації русизмів, що передбачає модифікацію вихідного тексту для створення синтетичних даних. Іменники та дієслова у вихідних реченнях випадковим чином перекладалися російською мовою із застосуванням трансформерної моделі, яка забезпечує високу точність і гнучкість у перекладі. Такий підхід дозволяє моделювати реальні ситуації використання русизмів у текстах, що є важливим для тренування моделей, здатних автоматично розпізнавати та виправляти ці помилки.

Для визначення частин мови, які необхідно перекладати, використовувалася бібліотека pymorphy2. Це було особливо корисним для вибору лише тих слів, які могли б створити впізнавані русизми, не порушуючи загальної граматичної структури речення.

Для забезпечення коректної транслітерації між українською та російською мовами було створено спеціальний словник відповідників звуків і літер. Цей словник враховує фонетичні особливості обох мов, дозволяючи уникати поширених помилок у процесі транслітерації. Він був побудований із використанням частотного аналізу відповідностей у паралельних текстах, що дозволило забезпечити максимальну точність перекладу.

Для генерації датасету використовувались частини коректних речень з існуючих датасетів UA-GEC та brown, які служили базою для створення синтетичних прикладів із русизмами та суржиком. Це дозволило зберегти структуру речень, водночас додаючи до них лексичні помилки, необхідні для тренування моделей. Приклад перетвореного речення з русизмами зображений на рисунку 4.3.

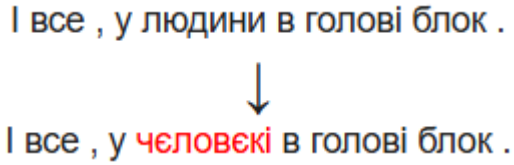

Рисунок 4.3 – Приклад речення з русизмом

4.4.2 Генерація англіцизмів

Попередні ідеї були запозичені в інших авторів, як-от додавання синтетично створених русизмів чи суржика до навчального датасету, що дозволяє ефективніше тренувати моделі для виправлення лексичних помилок. Нововведенням цього дослідження є ін’єкція англіцизмів до коректних речень, яка створює новий рівень складності для моделей граматичного аналізу та корекції.

Працює цей підхід схоже на попередній спосіб: іменники та дієслова з деякою ймовірністю перекладаються англійською мовою, після чого транслітеруються українськими літерами. Наприклад, слово «м’яч» може бути замінене на «бол», а «бігати» – на «ранати». Це дозволяє моделювати ситуації, коли в текстах трапляються англіцизми, що виникають через мовний вплив або неправильне використання запозичених слів.

Транслітерація здійснюється за допомогою бібліотеки transliterate, яка забезпечує підтримку кількох мов, зокрема української. Ця бібліотека дозволяє автоматично генерувати відповідники для англійських слів у вигляді українських транслітерацій, що спрощує процес створення синтетичних текстів із помилками.

Запровадження англіцизмів дозволяє дослідити ефективність моделі виправлення граматичних помилок, а також оцінити їхню здатність розрізняти запозичення, які є допустимими в літературній українській мові, і ті, що не відповідають її нормам. Таким чином, цей підхід сприяє вдосконаленню алгоритмів для роботи з реальними текстами, у яких

лексичні помилки часто зумовлені глобалізацією чи технологічним прогресом. Приклад некоректного речення з англіцизмами можна побачити на рисунку 4.4.

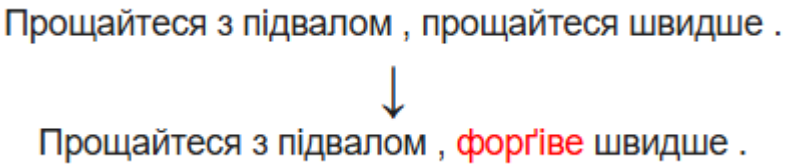

Рисунок 4.4 – Приклад речення з англіцизмом

4.4.3 Генерація кальок

Українським текстам з помилками також властиве калькування російської мови, що є типовим проявом мовної інтерференції. Це явище часто призводить до утворення структур, невластивих українській мові, і спотворення її милозвучності. Такі помилки можна певною мірою назвати русизмами, проте в цій роботі під русизмами розуміються лише окремі слова або вирази російської мови, які транслітеруються чи адаптуються до української форми без урахування граматичних і стилістичних норм.

Водночас, у задачі виправлення граматичних помилок у текстах буде враховано не тільки класичні русизми, а й проблеми стилістичної плавності й природності українського тексту. Ця задача вирішуватиметься в межах завдання «GEC+Fluency», як було зазначено раніше. Помилки плавності часто стосуються використання калькованих словосполучень, які не відповідають нормам української мови. Наприклад, вирази типу «прийняти участь» замість нормативного «взяти участь» або «на протязі» замість «протягом». Важливо зауважити, що контекст має ключове значення: у деяких випадках форма «на протязі» є коректною, якщо йдеться про фізичний протяг у приміщенні.

Ці помилки здебільшого є наслідком послівного перекладу з російської мови, коли структура та лексика вихідного тексту механічно переносяться в український переклад. Для усунення таких недоліків у текстах у цій роботі пропонується два підходи до генерування помилок плавності та милозвучності. Суть першого полягатиме у створенні вручну правил для заміни окремих мовленнєвих форм, у яких немає змінюваних частин мови (іменників, дієслів, прикметників тощо). Наприклад, фіксуються помилкові словосполучення, які часто зустрічаються у текстах, і на їх основі генеруються помилки для подальшого виправлення. Другий спосіб – послівний круговий переклад. Цей метод передбачає застосування перекладу тексту з української на російську, а потім знову на українську. На відміну від підходу, який використовувала команда Pravopysnyk [28], у цьому методі буде застосовано правильний токенізатор на кожному етапі перекладу. Це дозволить гарантувати, що всі слова у круговому перекладі залишатимуться написаними відповідно до української орфографії, без змішування з російським написанням.

4.5 Зворотний переклад

У третьому розділі цієї роботи докладно описано метод генерації синтетичних помилок, що базується на навчанні моделі: коректні речення подаються як вхідні дані, а помилкові – як вихідні. Таким чином, у рамках цього підходу вирішується задача AEG (artificial error generation), яка є протилежною до задачі GEC. На попередніх етапах створення синтетичних даних використовувалися методи, засновані на правилах. Результати цих підходів доповнять уже існуючі помилкові приклади з датасету UA-GEC, що дозволить суттєво розширити обсяг навчальних даних. Очікується, що збільшення кількості якісних помилкових екземплярів сприятиме більш ефективному навчанню моделі для задачі AEG.

На цьому етапі буде проведено серію експериментів, спрямованих на вивчення різних типів помилок. Зокрема, одна модель буде навчена виключно на помилкових прикладах із початкового датасету UA-GEC. Водночас інші моделі отримуватимуть додаткові навчальні дані, що включатимуть помилки, згенеровані за допомогою підходів на основі правил. Такий порівняльний підхід дозволить оцінити вплив різних джерел синтетичних даних на якість генерованих помилок.

Основна мета цього етапу – навчити модель максимально якісно генерувати помилки, пов’язані з плавністю мови (fluency). Це завдання є надзвичайно важливим, оскільки пунктуаційні та граматичні помилки, спричинені неузгодженістю частин речення, відносно легко виявляти та виправляти. Водночас помилки, що стосуються плавності мови, є значно складнішими для аналізу та генерації, адже вони залежать від контексту, лексичних зв’язків та стилістичних норм мови.

Виклик цього етапу полягає в тому, щоб створити модель, здатну не лише розпізнавати патерни помилок, пов’язаних із плавністю, але й генерувати на їх основі нові некоректні приклади. Такий підхід дозволить краще відтворити реальні помилки, які часто трапляються в текстах, і, в результаті, підвищить якість подальшого навчання моделей для задачі корекції текстів.

Для донавчання моделі AEG була обрана публічно доступна претренована mBART-large-50. BART (Bidirectional and Auto-Regressive Transformer) – це нейронна мовна модель, розроблена для задач обробки природної мови, яка поєднує в собі дві архітектурні особливості: двонаправлене кодування, що дозволяє моделі враховувати контекст з обох боків слова, та авторегресивне декодування, що дозволяє послідовно генерувати текст. Вона претренується на відновленні пошкодженого тексту, що робить її універсальним інструментом для вирішення задач, які потребують глибокого розуміння і генерації тексту. mBART-large-50 є багатомовним розширенням BART, яке підтримує 50 мов, зокрема

українську, і добре підходить для задач, де потрібна робота з кількома мовами або адаптація до специфічних мовних даних.

В своїй роботі [31] дослідники донавчають модель GEC в кілька етапів, спочатку на синтетично згенерованих даних, а потім на золотих даних, тобто початкових розмічених. Подібний принцип використовуватиметься і під час навчання моделей AEG в цій роботі. Як було зазначено раніше, не існує способу напряму протестувати ефективність моделі AEG, усі розробки в цьому напрямі тестуються вже на моделях для задачі виправлення граматичних помилок.

4.6 Навчання моделей GEC

На попередніх етапах були згенеровані окремі набори даних завдяки методам, описаним раніше. Вони включають синтетичні пунктуаційні та граматичні помилки, русизми та англіцизми, кальки та дані, що згенеровані навченою моделлю AEG завдяки зворотному перекладу. Також все ще залишається початковий анотований датасет UA-GEC та безпомилкові речення.

Було вирішено навчити кілька моделей на різних типах помилкових даних та порівняти їхню ефективність. Для навчання була обрана раніше згадана модель mBART-large-50. Остаточна оцінка $F_{0.5}$ розраховувалась за допомогою скрипта наданого авторами змагання UNLP-2023 в їхньому репозиторії. Скрипт повертає дві оцінки Span-Based Correction, що показує, чи правильно були виправлені речення, і Span-Based Detection, що показує, чи взагалі була ідентифікована помилка. Через велику теоретичну кількість моделей та обмежений обсяг доступних безкоштовних ресурсів на Google Colab, загальний обсяг навчальних даних для кожної моделі був обмежений, щоб підвищити швидкість навчання на GPU.

Спершу була навчена одна модель тільки на золотому датасеті UA-GEC, включно із помилковими та коректними реченнями. Результат

проілюстрований на рисунку 4.5. Це початковий результат і надалі задача полягає в том, щоб знайти найоптимальнішу комбінацію навчальних даних.

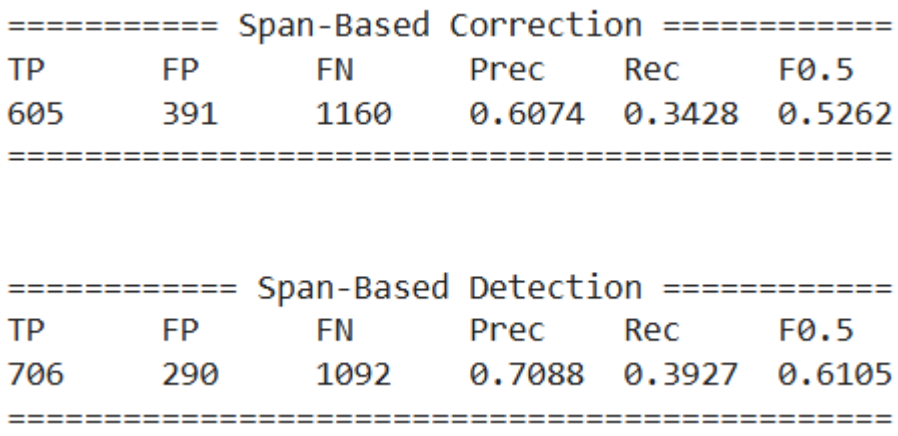

Рисунок 4.5 – Модель навчена на датасеті UA-GEC

В таблиці 4.1 наведені дані та метрика оцінки, на яких були навчені моделі у цьому експериментальному дослідженні. Моделі розміщені в хронологічному порядку їх навчання.

Таблиця 4.1 – Порівняння моделей GEC

correct	ua_gec	punct	grammar	rus	anglicism	calques	back	$F_{0.5}$
	full							0.5262
	full		1.5k	2k	2k	1k		0.5567
2k	full		2k	3k	4k		1.5k	0.5723
4k	full	3k	1.5k	2k	2k			0.5788
5k			5k	7k	5k	3k	3k	0.3782
5k	full	4k	3k	5k	2k	1k		0.5964
4.5k	full	3k	2k	5k	3k	2k	1.5k	0.5909
4.5k	full	5k		3k	2k	1.5k		0.5834

Як було згадано раніше перша модель – базова, лише на даних UA-GEC. У наступній моделі до початкового датасету було додано 1.5 тисячі речень із пунктуаційними помилками, 2 тисячі із граматичними, 2 тисячі із

русизмами та 1 тисячу із англіцизмами. Це дозволило моделі навчитися коригувати ці типи помилок, що призвело до покращення метрики $F_{0.5}$ на 3.05% порівняно з початковою моделлю.

Щоб підвищити якість роботи, у третій моделі було додано 2 тисячі речень без помилок, що знизило FN і збільшило обсяг граматичних і синтетичних помилок за допомогою моделі AEG. Це дало змогу краще балансувати між виявленням і виправленням помилок, покращивши $F_{0.5}$ ще на 1.57%. В усіх наступних моделях використовуватимуться додаткові коректні екземпляри для зниження FN.

Четверта модель була навчена на 4 тисячах коректних речень, 2 тисячах прикладів із англіцизмами й русизмами, а також на зменшеному обсязі граматичних помилок, що підвищило $F_{0.5}$ до 0.5788.

У п’ятій моделі модель навчалась виключно на синтетичних даних без золотого датасету UA-GEC. Збільшено обсяг синтетичних даних: 5 тисяч речень із пунктуаційними помилками, 7 тисяч із граматичними, 5 тисяч із русизмами, 3 тисячі із англіцизмами, 3 тисячі із кальками й 3 тисячі з AEG. Однак це різко знизило $F_{0.5}$ до 0.3782, вказуючи на необхідність використання початкових даних.

Шоста модель показала результат із $F_{0.5} = 0.5964$ завдяки збалансованому підходу до вибору даних. Було використано 5 тисяч речень без помилок, додано 4 тисячі пунктуаційних помилок, 2 тисячі граматичних, 5 тисяч русизмів, 2 тисячі англіцизмів і 2 тисячі кальок.

У сьомій моделі була підвищена кількість англіцизмів. Це незначно вплинуло на результат, хоча метрика $F_{0.5}$ знизилася до 0.5909. У восьмій моделі також зменшили кількість граматичних помилок і прикладів AEG, та повністю були прибрані дані від зворотного перекладу, що призвело до подальшого зниження $F_{0.5}$ до 0.5834.

Отже, найкращою моделлю виявилась шоста, в якій не були використані дані від моделі AEG, та було більше русизмів в порівнянні з англіцизмами. Повний звіт цієї моделі зображений на рисунку 4.6.

```
========== Span-Based Correction ============
TP      FP      FN      Prec    Rec     F0.5
611     247     1079    0.7121  0.3615  0.5964
=============================================


============ Span-Based Detection ============
TP      FP      FN      Prec    Rec     F0.5
687     171     1040    0.8007  0.3978  0.6658
=============================================
```

Рисунок 4.6 – Найкраща GEC модель

Вирівнювання балансу між кількістю помилкових і коректних речень у датасеті призвела до зменшення кількості хибних негативних результатів (FN). Додавання синтетично згенерованих пунктуаційних помилок позитивно вплинуло на показники TP, оскільки це дозволило досягти максимальної ефективності в цій категорії помилок. Зміни, пов'язані з додаванням граматичних помилок та русизмів, також покращили ефективність моделі, що видно з порівняння різних моделей. Однак додавання помилок, пов'язаних з англіцизмами, не призвело до значних покращень. Це може бути зумовлено низьким відсотком таких помилок у валідаційному наборі даних. Не можна стверджувати, що синтетично згенеровані моделю AEG екземпляри позитивно вплинули на здатність моделі GEC виправляти помилки плавності мови (fluency), оскільки цей тип помилок був частково вирішений за допомогою початкового золотого набору UA-GEC. Це, ймовірно, пов'язано з внутрішніми складнощами виправлення таких помилок.

ВИСНОВКИ

В ході виконання цієї кваліфікаційної роботи було досліджене питання розроблення системи виправлення граматичних та синтаксичних помилок в україномовних текстах. Як виявилось, ця задача називається grammatical error correction, і сюди входять не тільки граматичні та синтаксичні помилки, а й усі інші, що пов’язані з текстом, пунктуаційні, орфографічні, лексичні, морфологічні тощо. Був проведений ґрунтовний аналіз важливості цього роду задач у сучасному інформаційному світі. Теоретична частина дослідження цієї роботи була спрямована на вивчення основних існуючих підходів для вирішення задачі виправлення граматичних помилок, а саме на історичний розвиток state-of-the-art підходів в окремі періоди часу, починаючи з підходів на основі класифікації, SMT підходів, використання рекурентних нейронних мереж, впровадження механізму уваги та винайдення дослідниками з Google відносно простої архітектури Трансформер. Ця архітектура дала початок усім сучасним методам та моделям обробки природної мови. Одна з таких, mBART, була використана в роботі в якості GEC моделі.

В процесі дослідження предметної галузі стало зрозуміло, що задача виправлення граматичних помилок саме для української мови – малоресурсна та потребує штучно згенерованих даних. Надалі були теоретично досліджені підходи для генерації синтетичних помилок для англійської мови. Ідея полягала в тому, щоб пристосувати їх для нашої задачі, тобто для українських текстів. Лідери задачі виправлення граматичних помилок для англійської мови зазначають, що отримання найкращих результатів буде зумовлене не винайденням якоїсь нової архітектури моделі, а використання кращих навчальних даних. Тому за основу був взятий саме цей напрям дослідження.

В ході вивчення української сторони цієї задачі, були проаналізовані поточні state-of-the-art підходи запропоновані на конференції UNLP-2023.

Вони включали раніше досліджені методи генерації штучних даних, як от ін’єкція шуму та зворотний переклад. Експериментальна частина цієї кваліфікаційної роботи полягала у реалізації різних методів генерування синтетичних помилок та навчання моделі виправлення граматичних помилок. Були реалізовані генерація пунктуаційних, граматичних, кілька видів лексичних помилок, а також навчена модель за допомогою зворотного перекладу. Новаторська ідея цієї роботи – використання штучно згенерованих англіцизмів.

Для досягнення поставленої задачі, тобто розроблення системи виправлення граматичних помилок, були навчені кілька моделей на різних наборах даних та проведений порівняльний аналіз. Найкращий результат – модель, що має показник $F_{0.5}$ у 0.5964. Цей рівень був досягнутий внаслідок додавання штучно згенерованих пунктуаційних, граматичних помилок та русизмів. На жаль, використання англіцизмів не вплинуло позитивно на ефективність моделі.

Найскладнішим викликом все ще залишається виправлення помилок пов’язаних з плавністю мови. Ні це, ні жодне з раніше згаданих досліджень не досягло значного успіху у виправленні цього типу помилок. Це може бути вирішено, або більшою кількістю якісних анотованих даних, що міститимуть різноманітні патерни помилок плавності мови, або впровадженням ефективного способу для штучного генерування саме такого типу помилок. Поточний state-of-the-art застосовує потужнішу мовну модель, аніж ті, що використовувались на UNLP-2023, проте в тому підході для навчання не використовуються такі види помилок як русизми, наприклад. Комбінація методів, запропонованих в цій роботі, переможцями UNLP-2023 та мовної моделі LLaMa може призвести до значного покращення нинішньої ситуації із результатами задачі виправлення граматичних помилок.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Naber D. A rule-based style and grammar checker. Munich : GRIN Verlag, 2003. 76 p.

2. Using Statistical Techniques and Web Search to Correct ESL Errors / M. Gamon et al. *CALICO Journal*. 2013. Vol. 26, no. 3. P. 491–511. URL: https://doi.org/10.1558/cj.v26i3.491-511 (date of access: 16.01.2025).

3. Stahlberg F., Kumar S. Seq2Edits: Sequence Transduction Using Span-level Edit Operations. *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Online. Stroudsburg, PA, USA, 2020. URL: https://doi.org/10.18653/v1/2020.emnlp-main.418 (date of access: 16.01.2025).

4. Encode, Tag, Realize: High-Precision Text Editing / E. Malmi et al. *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, Hong Kong, China. Stroudsburg, PA, USA, 2019. URL: https://doi.org/10.18653/v1/d19-1510 (date of access: 16.01.2025).

5. UA-GEC: Grammatical Error Correction and Fluency Corpus for the Ukrainian Language / O. Syvokon et al. *Proceedings of the Second Ukrainian Natural Language Processing Workshop (UNLP)*, Dubrovnik, Croatia. Stroudsburg, PA, USA, 2023. URL: https://doi.org/10.18653/v1/2023.unlp-1.12 (date of access: 16.01.2025).

6. The Illinois-Columbia System in the CoNLL-2014 Shared Task / A. Rozovskaya et al. *Proceedings of the Eighteenth Conference on Computational Natural Language Learning: Shared Task*, Baltimore, Maryland. Stroudsburg, PA, USA, 2014. URL: https://doi.org/10.3115/v1/w14-1704 (date of access: 16.01.2025).

7. The CoNLL-2014 Shared Task on Grammatical Error Correction / H. T. Ng et al. *Proceedings of the Eighteenth Conference on Computational*

Natural Language Learning: Shared Task, Baltimore, Maryland. Stroudsburg, PA, USA, 2014. URL: https://doi.org/10.3115/v1/w14-1701 (date of access: 16.01.2025).

8. A Comprehensive Survey of Grammatical Error Correction / Y. Wang et al. *ACM Transactions on Intelligent Systems and Technology*. 2021. Vol. 12, no. 5. P. 1–51. URL: https://doi.org/10.1145/3474840 (date of access: 16.01.2025).

9. Brockett C., Dolan W. B., Gamon M. Correcting ESL errors using phrasal SMT techniques. *the 21st International Conference*, Sydney, Australia, 17–18 July 2006. Morristown, NJ, USA, 2006. URL: https://doi.org/10.3115/1220175.1220207 (date of access: 16.01.2025).

10. Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation / K. Cho et al. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Doha, Qatar. Stroudsburg, PA, USA, 2014. URL: https://doi.org/10.3115/v1/d14-1179 (date of access: 16.01.2025).

11. Sutskever I., Vinyals O., Le Q. V. Sequence to Sequence Learning with Neural Networks. *arXiv.org*. URL: https://arxiv.org/abs/1409.3215 (date of access: 16.01.2025).

12. Kalchbrenner N., Blunsom P. Recurrent Continuous Translation Models. *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, Seattle, Washington, USA. Stroudsburg, PA, USA, 2013. P. 1700–1709. URL: https://doi.org/10.18653/v1/d13-1176 (date of access: 16.01.2025).

13. Bahdanau D., Cho K., Bengio Y. Neural Machine Translation by Jointly Learning to Align and Translate. arXiv.org. URL: https://arxiv.org/abs/1409.0473 (date of access: 16.01.2025).

14. Show, Attend and Tell: Neural Image Caption Generation with Visual Attention / K. Xu et al. *arXiv.org*. URL: https://arxiv.org/abs/1502.03044 (date of access: 16.01.2025).

15. Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling / J. Chung et al. *arXiv.org*. URL: https://arxiv.org/abs/1412.3555 (date of access: 16.01.2025).

16. Attention Is All You Need / A. Vaswani et al. *arXiv.org*. URL: https://arxiv.org/abs/1706.03762 (date of access: 16.01.2025).

17. Pillars of Grammatical Error Correction: Comprehensive Inspection Of Contemporary Approaches In The Era of Large Language Models / K. Omelianchuk et al. *arXiv.org*. URL: https://arxiv.org/abs/2404.14914 (date of access: 16.01.2025).

18. Grammatical Error Correction: A Survey of the State of the Art / C. Bryant et al. *Computational Linguistics*. 2023. P. 1–59. URL: https://doi.org/10.1162/coli_a_00478 (date of access: 16.01.2025).

19. Corpora Generation for Grammatical Error Correction / J. Lichtarge et al. *Proceedings of the 2019 Conference of the North*, Minneapolis, Minnesota. Stroudsburg, PA, USA, 2019. URL: https://doi.org/10.18653/v1/n19-1333 (date of access: 16.01.2025).

20. Grundkiewicz R., Junczys-Dowmunt M., Heafield K. Neural Grammatical Error Correction Systems with Unsupervised Pre-training on Synthetic Data. *ACL Anthology*. URL: https://aclanthology.org/W19-4427/ (date of access: 16.01.2025).

21. Rozovskaya A., Roth D. Training Paradigms for Correcting Errors in Grammar and Usage. *ACL Anthology*. URL: https://aclanthology.org/N10-1018/ (date of access: 16.01.2025).

22. An Empirical Study of Incorporating Pseudo Data into Grammatical Error Correction / S. Kiyono et al. *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, Hong Kong, China. Stroudsburg, PA, USA, 2019. URL: https://doi.org/10.18653/v1/d19-1119 (date of access: 16.01.2025).

23. Stahlberg F., Kumar S. Synthetic Data Generation for Grammatical Error Correction with Tagged Corruption Models. *arXiv.org*. URL: https://arxiv.org/abs/2105.13318 (date of access: 16.01.2025).

24. Improving Grammatical Error Correction with Machine Translation Pairs / W. Zhou et al. *Findings of the Association for Computational Linguistics: EMNLP 2020*, Online. Stroudsburg, PA, USA, 2020. URL: https://doi.org/10.18653/v1/2020.findings-emnlp.30 (date of access: 16.01.2025).

25. Bryant C., Felice M., Briscoe T. Automatic Annotation and Evaluation of Error Types for Grammatical Error Correction. *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Vancouver, Canada. Stroudsburg, PA, USA, 2017. URL: https://doi.org/10.18653/v1/p17-1074 (date of access: 16.01.2025).

26. Syvokon O., Romanyshyn M. The UNLP 2023 Shared Task on Grammatical Error Correction for Ukrainian. *Proceedings of the Second Ukrainian Natural Language Processing Workshop (UNLP)*, Dubrovnik, Croatia. Stroudsburg, PA, USA, 2023. URL: https://doi.org/10.18653/v1/2023.unlp-1.16 (date of access: 16.01.2025).

27. Náplava J., Straka M. Grammatical Error Correction in Low-Resource Scenarios. *Proceedings of the 5th Workshop on Noisy User-generated Text (W-NUT 2019)*, Hong Kong, China. Stroudsburg, PA, USA, 2019. URL: https://doi.org/10.18653/v1/d19-5545 (date of access: 16.01.2025).

28. Comparative Study of Models Trained on Synthetic Data for Ukrainian Grammatical Error Correction / M. Bondarenko et al. *Proceedings of the Second Ukrainian Natural Language Processing Workshop (UNLP)*, Dubrovnik, Croatia. Stroudsburg, PA, USA, 2023. URL: https://doi.org/10.18653/v1/2023.unlp-1.13 (date of access: 16.01.2025).

29. Multilingual Translation from Denoising Pre-Training / Y. Tang et al. *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, Online. Stroudsburg, PA, USA, 2021.

URL: https://doi.org/10.18653/v1/2021.findings-acl.304 (date of access: 16.01.2025).

30. To Err Is Human, but Llamas Can Learn It Too / A. Luhtaru et al. *Findings of the Association for Computational Linguistics: EMNLP 2024*, Miami, Florida, USA. Stroudsburg, PA, USA, 2024. P. 12466–12481. URL: https://doi.org/10.18653/v1/2024.findings-emnlp.727 (date of access: 16.01.2025).

31. Palma Gomez F., Rozovskaya A., Roth D. A Low-Resource Approach to the Grammatical Error Correction of Ukrainian. *Proceedings of the Second Ukrainian Natural Language Processing Workshop (UNLP)*, Dubrovnik, Croatia. Stroudsburg, PA, USA, 2023. URL: https://doi.org/10.18653/v1/2023.unlp-1.14 (date of access: 16.01.2025).