НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

Навчально-науковий інститут прикладного системного аналізу

Кафедра штучного інтелекту

До захисту допущено:
В. о. завідувачки кафедри
_____ Ірина ДЖИГИРЕЙ
«___»_____20__ р.

Дипломна робота

на здобуття ступеня бакалавра
за освітньо-професійною програмою «Системи і методи штучного інтелекту»
спеціальності 122 «Комп'ютерні науки» на тему:
«Аналіз впливу машинного перекладу на метрики анотування текстів»

Виконав:
Студент IV курсу, групи КІ-13
Думайло Тарас Михайлович _____

Керівник:
доцент кафедри штучного інтелекту, к.ф-м.н., доцент,
Пишнограєв Іван Олександрович _____

Консультант з економічного розділу:
доцент кафедри економічної кібернетики, к.е.н., доцент,
Рощина Надія Василівна _____

Консультант з нормоконтролю:
фахівець першої категорії кафедри штучного інтелекту,
к.т.н., доцент, Комариста Богдана Миколаївна _____

Рецензент:
доцент кафедри економічної кібернетики, к.ф-м.н., доцент,
Лазаренко Ірина Сергіївна _____

Засвідчую, що у цій дипломній роботі
немає запозичень з праць інших
авторів без відповідних посилань.
Студент _____

Київ – 2025 року

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Навчально-науковий інститут прикладного системного аналізу
Кафедра штучного інтелекту

Рівень вищої освіти – перший (бакалаврський)

Спеціальність – 122 «Комп'ютерні науки»

Освітньо-професійна програма «Системи і методи штучного інтелекту»

ЗАТВЕРДЖУЮ
В. о. завідувачки кафедри
_____ Ірина ДЖИГИРЕЙ
«15» січня 2025 р.

ЗАВДАННЯ
на дипломну роботу студенту
Думайлу Тарасу Михайловичу

1. Тема роботи «Аналіз впливу машинного перекладу на метрики анотування текстів», керівник роботи Пишнограєв Іван Олександрович, доцент кафедри ШІ, затверджені наказом наказом по НН ІПСА від «26» травня 2025 р. № 1759-с.
2. Термін подання студентом роботи «09» червня 2025 року.
3. Вихідні дані до роботи набір тестових текстів
4. Зміст роботи Аналіз впливу машинного перекладу на автоматичну анотацію текстів, підготовка корпусу даних, вибір моделей для сумаризації та перекладу, реалізація метрик оцінки якості, експериментальна перевірка та аналіз результатів.
5. Перелік ілюстративного матеріалу електронна презентація.
6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Економічний	Рощина Надія Василівна, доцент, к. е. н.		

7. Дата видачі завдання «03» лютого 2025 року.

Календарний план

№ з/п	Назва етапів виконання дипломної роботи	Термін виконання етапів роботи	Примітка
1	Огляд літератури за темою	21.04.2025	Виконано
2	Підготовка першого розділу	28.04.2025	Виконано
3	Підготовка другого розділу	05.05.2025	Виконано
4	Розробка програмного продукту	12.05.2025	Виконано
5	Підготовка третього розділу	19.05.2025	Виконано
6	Оформлення розділів дипломної роботи	02.06.2025	Виконано
7	Підготовка презентації доповіді	09.06.2025	Виконано

Студент Тарас ДУМАЙЛО

Керівник Іван ПИШНОГРАЄВ

РЕФЕРАТ

Дипломна робота: 86 стор., 10 рис., 8 табл., 18 посилань, 2 додатки.

МАШИННИЙ ПЕРЕКЛАД, АНОТУВАННЯ ТЕКСТІВ, МЕТРИКИ ЯКОСТІ, СУМАРИЗАЦІЯ, НЕЙРОННІ МОДЕЛІ

У роботі досліджується вплив машинного перекладу на якість автоматичного анотування текстів за допомогою сучасних нейронних моделей. Основну увагу приділено оцінці змін у результатах сумаризації після перекладу текстів з англійської на українську мову та навпаки. Для аналізу використовуються різні метрики якості — ROUGE, BLEU, BERTScore та інші. Побудовано експериментальний пайплайн, що включає збір даних, автоматичне узагальнення текстів, машинний переклад, а також обчислення метрик та візуалізацію результатів.

Об’єктом дослідження є процес автоматичного анотування текстів у контексті багатомовного середовища. Предметом дослідження виступає зміна метрик якості сумаризації в результаті застосування машинного перекладу. Актуальність роботи зумовлена широким використанням автоматичних систем обробки текстів у багатомовних інформаційних середовищах, де якість узагальнень може значно залежати від проміжного перекладу.

Проаналізовано вплив типових систем машинного перекладу на результати сумаризації, зокрема на лексичні та семантичні відповідності між оригінальними й перекладеними текстами. Показано, як різні метрики реагують на перекладні зміни та які моделі виявляються найбільш стійкими до таких змін. Робота робить внесок у розуміння впливу мовної трансформації на автоматичну обробку текстів та відкриває перспективи для підвищення точності багатомовних NLP-систем.

ABSTRACT

Thesis: 86 pages, 10 figures, 8 tables, 18 references, 2 appendices.

MACHINE TRANSLATION, TEXT ANNOTATION, QUALITY METRICS, SUMMARIZATION, NEURAL MODELS

This thesis explores the impact of machine translation on the quality of automatic text annotation using modern neural models. The main focus is on evaluating changes in summarization results after translation texts from English to Ukrainian and vice versa. Various quality metrics — ROUGE, BLEU, BERTScore, and others — are used for analysis. An experimental pipeline is developed, including data collection, automatic summarization, machine translation, metric computation, and result visualization.

The object of the research is the process of automatic text annotation in a multilingual context. The subject of the research is variation in quality metrics summarization resulting from use of machine translation. The urgency of this research arises from the widespread application of automatic text processing systems in multilingual digital environments, where the quality of summaries can be significantly affected by intermediate translation.

The impact of typical machine translation systems on summarization results, in particular on lexical and semantic correspondence between original and translated texts, is analyzed. How different metrics respond to translation-induced changes and which models are most resilient to such transformations is demonstrated in this thesis. This research contributes to a better understanding of the effects of language transformation on automatic text processing and introduces prospects for improving the accuracy of multilingual NLP systems.

ЗМІСТ

ПЕРЕЛІК ПРИЙНЯТИХ СКОРОЧЕНЬ 8
РОЗДІЛ 1 АНАЛІЗ ПРОБЛЕМИ ВПЛИВУ МАШИННОГО ПЕРЕКЛАДУ НА ЯКІСТЬ АНОТУВАННЯ ТЕКСТІВ 11
1.1 Загальна характеристика задачі анотування текстів 11
1.2 Розвиток технологій машинного перекладу та їх застосування 12
1.3 Проблематика перекладу в контексті анотування текстів 14
1.4 Огляд метрик оцінювання якості анотації текстів 16
1.5 Постановка задачі дослідження 18
Висновки до розділу 1 19
РОЗДІЛ 2 ПОБУДОВА ЕКСПЕРИМЕНТАЛЬНОГО ПАЙПЛАЙНУ ДЛЯ ОЦІНКИ ВПЛИВУ ПЕРЕКЛАДУ 21
2.1 Основи автоматичного анотування текстів у задачах сумаризації .. 21
2.2 Огляд сучасних інструментів машинного перекладу 23
2.3 Обґрунтування вибору корпусу, моделей і метрик 27
Висновки до розділу 2 30
РОЗДІЛ 3 ЕКСПЕРИМЕНТАЛЬНИЙ АНАЛІЗ ТА ІНТЕРПРЕТАЦІЯ РЕЗУЛЬТАТІВ 31
3.1 Опис експериментального пайплайну та сценаріїв перекладу 31
3.2 Реалізація пайплайну автоматичної сумаризації та перекладу 33
3.3 Методика оцінювання якості узагальнень 34
3.4 Порівняльний аналіз результатів для різних сценаріїв перекладу .. 37
3.5 Візуалізація, інтерпретація результатів та лінгвістичні спостереження 40
Висновки до розділу 3 44

РОЗДІЛ 4 ФУНКЦІОНАЛЬНО-ВАРТІСНИЙ АНАЛІЗ СИСТЕМИ ОЦІНКИ ВПЛИВУ МАШИННОГО ПЕРЕКЛАДУ НА ЯКІСТЬ АНОТУВАННЯ 46
4.1 Постановка задачі проектування 47
4.2 Обґрунтування функцій програмного продукту 48
4.3 Обґрунтування системи параметрів програмного продукту 51
4.4 Аналіз експертного оцінювання 53
4.5 Аналіз рівня варіантів реалізації функцій 57
4.6 Економічний аналіз варіантів розробки ПП 58
4.7 Вибір кращого варіанту ПП техніко-економічного рівня 63
Висновки до розділу 4 64
ВИСНОВКИ 67
ПЕРЕЛІК ПОСИЛАНЬ 69
ДОДАТОК А ЛІСТИНГ ПРОГРАМИ 71
ДОДАТОК Б ЕЛЕКТРОННА ПРЕЗЕНТАЦІЯ 83

ПЕРЕЛІК ПРИЙНЯТИХ СКОРОЧЕНЬ

MT – Machine Translation (машинний переклад).

NLP – Natural Language Processing (обробка природної мови).

ML – Machine Learning (машинне навчання).

DL – Deep Learning (глибинне навчання).

ROUGE – Recall-Oriented Understudy for Gisting Evaluation (метрика оцінки якості резюме).

BLEU – Bilingual Evaluation Understudy (метрика якості машинного перекладу).

BERT – Bidirectional Encoder Representations from Transformers.

BERTScore – метрика якості тексту на основі BERT-ембеддингів.

HF – Hugging Face (платформа для NLP-моделей).

UI – User Interface (користувацький інтерфейс).

EN – English (англійська мова).

UK – Ukrainian (українська мова).

GPU – Graphics Processing Unit (графічний процесор).

TF-IDF – Term Frequency-Inverse Document Frequency (метод зважування слів у тексті).

PP – Програмний продукт.

FVA – Функціонально-вартісний аналіз.

CNN/DM – CNN/DailyMail (корпус новинних текстів).

CSV – Comma-Separated Values (формат табличних даних).

Colab – Google Colaboratory (хмарне середовище для обчислень).

HDD – Hard Disk Drive (жорсткий диск).

RAM – Random Access Memory (оперативна пам’ять).

NMT – Neural Machine Translation (нейронний машинний переклад).

SUM – Summary (резюме/анотація).

T-E – Translation–Evaluation (переклад–оцінка).

ВСТУП

У сучасному[1] світі автоматична обробка текстової інформації набуває все більшого значення. Однією з ключових задач у цій галузі є автоматичне анотування текстів, зокрема їх сумаризація. В умовах глобалізації та активного використання багатомовних інформаційних систем зростає потреба у високоякісному машинному перекладі, який є проміжним етапом у багатьох NLP-завданнях. Проте машинний переклад не є ідеальним — він може змінювати лексичну, синтаксичну та семантичну структуру тексту, що, своєю чергою, впливає на результати подальшої обробки, зокрема на точність автоматичного узагальнення.

У багатьох прикладних задачах, де тексти перекладаються перед обробкою або після неї, важливо розуміти, наскільки переклад змінює оцінки якості анотацій. Зокрема, це стосується таких метрик, як ROUGE, BLEU, BERTScore та інші, що широко застосовуються для оцінювання якості сумаризації. Без урахування впливу перекладу на ці метрики можна отримати хибні висновки про ефективність моделей, що використовуються для анотації.

Актуальність даного дослідження полягає в тому, що в багатьох системах обробки текстів переклад є невід'ємною частиною, і від його точності залежить якість кінцевого результату. При цьому, попри значний прогрес у розвитку систем машинного перекладу, існує недостатньо досліджень, які б системно аналізували вплив перекладу на метрики якості анотацій, особливо в контексті сумаризації.

---

[1] Тут і нижче використано такий інструмент штучного інтелекту як чат-бот з генеративним штучним інтелектом ChatGPT виключно для корегування та редагування тексту, створеного автором цієї дипломної роботи, на основі автоматизованої перевірки граматики, структури та стилю, що відповідає Політиці використання штучного інтелекту для академічної діяльності в КПІ ім. Ігоря Сікорського (протокол №11 Вченої ради КПІ ім. Ігоря Сікорського від 11 грудня 2023 р.).

Метою цієї дипломної роботи є аналіз впливу машинного перекладу на обчислення метрик якості анотування текстів. У межах дослідження буде розроблено експериментальний пайплайн, що включає автоматичну генерацію резюме текстів, їх переклад із використанням нейронних моделей, обчислення метрик та порівняння результатів до і після перекладу. Особливу увагу буде приділено аналізу лексичних і семантичних змін, які виникають унаслідок перекладу, а також тому, як ці зміни впливають на якісні показники сумаризації.

Результати роботи можуть бути корисними для подальшого вдосконалення багатомовних NLP-систем, розробки більш стійких метрик оцінювання та підвищення точності анотацій у середовищах, де використовується машинний переклад.

РОЗДІЛ 1 АНАЛІЗ ПРОБЛЕМИ ВПЛИВУ МАШИННОГО ПЕРЕКЛАДУ НА ЯКІСТЬ АНОТУВАННЯ ТЕКСТІВ

1.1 Загальна характеристика задачі анотування текстів

Анотування текстів є важливим завданням в області обробки природної мови (Natural Language Processing, NLP), що передбачає створення узагальненого або стислого представлення вихідного тексту. Ця задача має ключове значення для багатьох практичних застосувань, зокрема автоматичного реферування, побудови систем рекомендацій, покращення результатів пошуку, чат-ботів та аналізу великих обсягів інформації [1]. Анотації дозволяють скоротити час на ознайомлення з великими текстовими даними, зберігаючи при цьому основний зміст і ключові повідомлення.

У сучасній лінгвістиці та штучному інтелекті розрізняють кілька основних типів анотацій. Найпоширенішими є екстрактивні та абстрактивні анотації. Екстрактивна анотація полягає у виборі окремих речень або фраз з тексту, які найкраще передають його суть. Такий підхід є менш складним у реалізації, проте має обмеження в точності та гнучкості. Абстрактивна анотація, навпаки, генерує новий текст, що стисло переказує оригінальний зміст, використовуючи інші слова або структури. Цей підхід вимагає складніших моделей, зокрема трансформерних нейронних мереж, але дає змогу досягати більш природного та осмисленого результату [2].

Завдання автоматичного анотування набуло особливої актуальності з появою великих трансформерних моделей, таких як BART, T5, PEGASUS та інших, які демонструють високі результати в задачах сумаризації [3]. Такі моделі навчаються на величезних обсягах текстових даних і здатні генерувати текст, зберігаючи логіку, змістову цілісність і навіть стиль оригіналу.

У системах сумаризації анотації виступають як кінцевий продукт, який допомагає користувачеві швидко зрозуміти суть великого тексту без потреби

читати його повністю. Це особливо важливо для новинних порталів, наукових публікацій, юридичних документів, медичних звітів та багатьох інших сфер [4]. Анотації також мають ключову роль у підготовці датасетів, що використовуються для навчання та оцінювання моделей штучного інтелекту, які повинні не лише «читати», а й розуміти текст.

Анотування текстів — це не лише інструмент для економії часу, а й критично важлива складова ефективної взаємодії людини з інформаційними системами. Розвиток методів автоматичного анотування сприяє підвищенню якості комунікації, оптимізації обробки текстових даних і поглибленню розуміння природної мови в контексті сучасного штучного інтелекту [5].

1.2 Розвиток технологій машинного перекладу та їх застосування

Машинний переклад є одним із найстаріших напрямів дослідження в галузі обробки природної мови, історія якого починається ще з середини XX століття. Перші системи машинного перекладу базувалися на правилах (rule-based machine translation, RBMT), де для кожної пари мов вручну створювалися лексикони та граматичні правила. Такий підхід був надзвичайно трудомістким і не здатен масштабуватися на велику кількість мовних пар, особливо через синтаксичну та семантичну складність природних мов [6].

У 1990-х роках з розвитком обчислювальних потужностей популярності набув статистичний машинний переклад (statistical machine translation, SMT), який ґрунтувався на статистичному аналізі паралельних корпусів текстів. Найвідомішою реалізацією SMT є система Moses, яка використовувала фразовий підхід для визначення відповідності між мовами [7]. Однак SMT мав свої обмеження, зокрема погану здатність до

узагальнення, втрату контексту та невисоку якість перекладу для мов з вільним порядком слів.

Значний прорив у якості перекладу стався з появою нейронного машинного перекладу (neural machine translation, NMT). Перші реалізації NMT базувалися на рекурентних нейронних мережах (RNN), які дозволяли обробляти послідовності слів і зберігати контекст попередніх елементів. Архітектури на основі RNN, як-от seq2seq із механізмом attention, значно покращили якість перекладу, проте мали обмеження щодо довготривалого контексту та паралельного обчислення [8].

Найбільш суттєве покращення NMT-систем настало з появою архітектури Transformer, запропонованої дослідниками компанії Google у 2017 році [9]. Transformer повністю ґрунтується на механізмі attention і дає змогу обробляти текстові послідовності паралельно, що істотно підвищує ефективність обчислень і якість перекладу. Саме ця архітектура стала основою для сучасних систем машинного перекладу, зокрема Google Translate, DeepL, Meta AI та багатьох моделей з відкритим кодом.

Однією з відомих реалізацій NMT є платформа MarianMT — відкритий фреймворк нейронного машинного перекладу, розроблений компанією Microsoft [10]. MarianMT базується на оптимізованій реалізації Transformer і уможливлює швидко тренувати багатомовні моделі, що можуть працювати на обмежених обчислювальних ресурсах. Моделі MarianMT активно використовуються у дослідницьких задачах, а також інтегруються в системи автоматичного перекладу текстів, чат-ботів і голосових асистентів.

Із розвитком технологій зростає і точність машинного перекладу, особливо при роботі з популярними мовними парами. Проте навіть найсучасніші моделі мають свої обмеження — зокрема, при перекладі з/на мови з різною структурою, семантикою або багатозначністю. Ці труднощі можуть мати істотний вплив на якість сумаризації або анотування, коли переклад використовується як проміжний крок у багатомовній обробці

даних. Зокрема, навіть незначні відхилення в перекладі можуть вплинути на точність порівняльних метрик та зрозумілість анотацій.

Розвиток технологій машинного перекладу пройшов шлях від жорстких правил до потужних нейронних моделей, здатних до глибокого контекстного аналізу тексту. Незважаючи на значні досягнення, подальше вдосконалення таких систем залишається актуальним, особливо в контексті збереження семантики та стилістичної точності при міжмовній трансформації тексту в задачах NLP.

1.3 Проблематика перекладу в контексті анотування текстів

Використання машинного перекладу в завданнях анотування текстів, зокрема при створенні резюме (сумаризації), породжує низку складних викликів, пов’язаних зі збереженням точності, семантичної цілісності та стилістичної відповідності інформації. Коли тексти перекладаються з однієї мови на іншу перед процесом анотації або навпаки — після створення резюме, можуть виникати спотворення, які безпосередньо впливають на якість обробки даних та оцінку результатів.

Однією з головних проблем є втрата або трансформація змісту. Під час перекладу окремі слова або вирази можуть втрачати свої нюанси, багатозначність або культурний контекст, що є критичним у процесі узагальнення інформації. Наприклад, синонімічні або метафоричні конструкції, які не мають прямих відповідників у мові перекладу, можуть бути передані неправильно або надмірно спрощено. Це призводить до того, що згенерована анотація не повністю відповідає початковому тексту, або має зміщений акцент, що унеможливлює адекватну оцінку якості сумаризації [11].

Ще одним важливим аспектом є стилістичні зміни, які виникають внаслідок особливостей мовної моделі перекладача. Різні моделі NMT можуть мати власні стилістичні уподобання: наприклад, одна модель може віддавати перевагу офіційно-науковому стилю, тоді як інша — більш розмовному. Така стилістична невідповідність між оригіналом та перекладом порушує однорідність корпусу, що є важливим фактором у машинному навчанні. Внаслідок цього можуть знижуватися показники точності моделей, що навчаються або тестуються на змішаних корпусах.

Крім того, використання перекладених текстів впливає на метрики автоматичного оцінювання, зокрема ROUGE, BLEU, METEOR, BERTScore тощо. Ці метрики оцінюють схожість між автоматично згенерованим та еталонним текстом, і будь-яке відхилення, викликане помилками перекладу, може суттєво знизити їх результати. Наприклад, якщо переклад не зберігає точну лексичну форму або порядок слів, навіть при збереженні загального сенсу, метрика ROUGE може показати низьку відповідність через неспівпадіння підрядків. Особливо це помітно у BLEU, який чутливий до точного формулювання [12].

У дослідженнях відзначається, що наявність перекладеного контенту у корпусах може спричинити хибне уявлення про продуктивність моделей сумаризації або класифікації. Відтак, при побудові багатомовних систем або аналізі їх якості потрібно враховувати, чи залучена стадія перекладу, та проводити окрему перевірку її впливу на результати. Існує також необхідність у розробці спеціалізованих метрик, стійких до перекладних варіацій, або адаптації існуючих методів з урахуванням лексичних та стилістичних втрат [13].

Отже, хоча машинний переклад значно розширює можливості обробки текстів у багатомовному середовищі, його використання в задачах анотування повинне супроводжуватись глибоким аналізом впливу на якість і достовірність результатів. Залучення додаткових стратегій, таких як

постредагування або використання багатомовних моделей сумаризації, може стати ефективним рішенням для зниження негативного впливу перекладу.

1.4 Огляд метрик оцінювання якості анотації текстів

Оцінювання якості автоматичної анотації текстів є ключовим етапом у розробці та тестуванні систем природної мовної обробки. Існує кілька загальноприйнятих метрик, які дозволяють кількісно порівнювати згенеровані анотації з еталонними людськими референтами. Найбільш поширеними з них є ROUGE, BLEU, BERTScore та MoverScore. Кожна з цих метрик має свої особливості, переваги й обмеження, особливо в контексті використання перекладених текстів.

ROUGE (Recall-Oriented Understudy for Gisting Evaluation) — одна з найпоширеніших метрик для оцінювання якості автоматичної сумаризації. Вона базується на підрахунку збігів між словами, n-грамами, підрядками або лемами в автоматично створеному й референтному текстах. ROUGE-N (де N — довжина n-грам) дає змогу виявити ступінь подібності на лексичному рівні. Проте, ця метрика є чутливою до варіативності формулювань та синтаксису. У разі перекладених текстів, навіть незначні зміни в порядку слів або синонімія можуть призвести до зниження показників ROUGE, що ускладнює об’єктивну оцінку [14].

BLEU (Bilingual Evaluation Understudy) спочатку розроблялась для машинного перекладу, але широко використовується і для сумаризації. BLEU оцінює точність шляхом вимірювання кількості спільних n-грам між гіпотезою та еталоном. Її головна особливість — строгість: метрика карає за відсутність точних збігів. Це робить BLEU досить вразливою до стилістичних відмінностей, що часто виникають при перекладі. Наприклад,

якщо переклад містить синоніми або парафрази, BLEU може дати низький бал, навіть якщо зміст передано правильно [15].

BERTScore — більш сучасна метрика, яка використовує контекстуальні векторні представлення слів на основі моделі BERT. Вона обчислює подібність між текстами, порівнюючи вектори кожного токена в гіпотезі та референті, що дає змогу враховувати семантичну близькість, а не лише збіг лексичних форм. Завдяки цьому, BERTScore краще переноситься на перекладені тексти, оскільки здатна розпізнати схожість навіть за наявності синонімів або зміни структури речень. Проте її точність залежить від вибору моделі та мови, що може створювати додаткову складність у багатомовному контексті [16].

MoverScore — ще одна семантично орієнтована метрика, яка використовує векторні представлення слів і обчислює вартість переміщення (за аналогією з Earth Mover’s Distance) між гіпотезою та референтом. Вона оцінює, наскільки "далеко" одне представлення потрібно перемістити, щоб відповідати іншому. Завдяки цьому MoverScore є менш чутливою до змін структури речення або синонімії. Як і BERTScore, вона краще адаптована до перекладених текстів, але її обчислення є складнішим і потребує значних обчислювальних ресурсів.

Загалом, традиційні метрики, як ROUGE і BLEU, демонструють обмеження у випадку перекладених текстів, оскільки покладаються на поверхневі лексичні збіги. У той же час, метрики нового покоління — BERTScore та MoverScore — надають більш надійні результати при оцінюванні текстів з різними лексичними та стилістичними варіаціями, що виникають під час перекладу. Це особливо актуально в задачах міжмовної анотації, де збереження смислової цілісності важливіше за точну відповідність формулюванням.

1.5 Постановка задачі дослідження

У контексті стрімкого розвитку технологій природної мовної обробки та поширення багатомовних систем, що охоплюють машинний переклад, все більш актуальним стає питання збереження якості текстової анотації при зміні мовного середовища. Машинний переклад часто використовується як інструмент для адаптації даних до іншої мови в задачах автоматичного узагальнення, класифікації, реферування тощо. Проте цей процес може впливати на точність, повноту й адекватність автоматично створених анотацій, особливо в разі багатокрокової обробки текстів. У зв’язку з цим виникає необхідність комплексного аналізу того, як машинний переклад впливає на результати анотування текстів, зокрема на значення метрик, що використовуються для його оцінювання.

Метою дослідження є вивчення впливу машинного перекладу на якість текстової анотації в системах автоматичного узагальнення на прикладі англомовних і перекладених українською текстів, з подальшою оцінкою результатів за допомогою стандартних метрик якості.

Об’єктом дослідження є процес автоматичного анотування текстів у системах обробки природної мови.

Предметом дослідження є вплив машинного перекладу на результати анотування текстів та точність обчислення оцінювальних метрик якості таких анотацій.

Для досягнення мети дослідження необхідно вирішити такі основні задачі:

- провести огляд існуючих методів текстової анотації та метрик для їх оцінювання;
- проаналізувати сучасні підходи до машинного перекладу та вибрати відповідні інструменти для досліду;

- побудувати експериментальний пайплайн із використанням системи сумаризації, перекладу та автоматичного оцінювання якості анотацій;
- порівняти якість анотацій до та після машинного перекладу з використанням метрик ROUGE, BLEU, BERTScore та MoverScore;
- зробити висновки щодо зміни точності оцінювання внаслідок машинного перекладу та надати рекомендації щодо використання перекладених даних в NLP-системах.

Висновки до розділу 1

У першому розділі розглянуто ключові аспекти проблематики впливу машинного перекладу на якість анотування текстів у сфері обробки природної мови. Визначено, що автоматичне анотування є важливою складовою багатьох NLP-завдань, зокрема сумаризації, де від точності, змістовності й збереження сенсу залежить якість кінцевого результату.

Окреслено історію та сучасний стан розвитку машинного перекладу, зокрема перехід від статистичних моделей до нейронних архітектур, таких як RNN, Transformer, MarianMT, що забезпечують високу точність, проте не позбавлені помилок. Встановлено, що при перекладі тексту можуть спостерігатися лексичні втрати, стилістичні спотворення або зміщення контексту, що суттєво впливає на якість анотації, сформованої або оціненої після перекладу.

Проаналізовано основні метрики оцінювання якості анотації текстів: ROUGE, BLEU, BERTScore і MoverScore. Виявлено, що деякі з них чутливі до стилістичних і лексичних змін, спричинених перекладом, що може призводити до спотворення оцінки якості анотацій.

Підсумовуючи, можна зазначити, що машинний переклад є потужним інструментом для багатомовної обробки текстів, однак його використання в процесі анотування супроводжується низкою викликів, які потребують додаткового аналізу. Сформульована постановка задачі уможливлює перехід до побудови експериментального дослідження для визначення конкретного впливу перекладу на результати анотації текстів і точність оцінювальних метрик.

РОЗДІЛ 2 ПОБУДОВА ЕКСПЕРИМЕНТАЛЬНОГО ПАЙПЛАЙНУ ДЛЯ ОЦІНКИ ВПЛИВУ ПЕРЕКЛАДУ

2.1 Основи автоматичного анотування текстів у задачах сумаризації

Автоматичне анотування текстів, зокрема у формі текстової сумаризації, є однією з ключових задач в обробці природної мови (NLP). Сумаризація полягає у створенні короткого, але змістовного викладу вмісту довшого тексту з максимально можливим збереженням його смислового навантаження. Вона може бути реалізована у двох основних підходах: екстрактивному та абстрактивному. Екстрактивна сумаризація здійснює вибір важливих речень або фрагментів із джерела без зміни їхньої форми, тоді як абстрактивна — генерує нові речення, які переказують зміст, подібно до того, як це робить людина.

Одним із перших проривів у галузі абстрактивної сумаризації стала поява архітектури sequence-to-sequence (seq2seq), яка використовує рекурентні нейронні мережі (RNN) з механізмом уваги. Нижче представлено принцип роботи моделі (рис. 2.1). У роботі Nallapati та ін. [2] запропоновано використання багаторівневих рекурентних моделей зі словниковим обмеженням та контекстною увагою, що дозволило значно покращити якість створених анотацій. Проте такі моделі часто стикалися з труднощами у генерації довготривалих залежностей і семантично цілісного тексту, особливо при складних вхідних документах.

Справжній прорив у задачах автоматичного узагальнення забезпечили трансформерні архітектури, зокрема модель BART (Bidirectional and Auto-Regressive Transformers), яку показано на рисунку 2.2, яка є потужним представником родини seq2seq з попереднім навчанням [3]. BART поєднує ідеї кодувальника з бідірекційним контекстом (як у BERT) та автогенеративного декодувальника (як у GPT), дозволяючи створювати

глибоко узагальнені текстові анотації. Вона демонструє високу продуктивність у задачах як англомовної, так і багатомовної сумаризації.

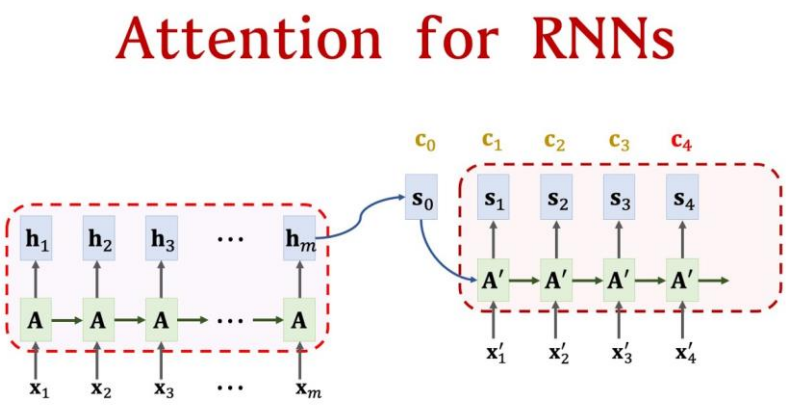


Рисунок 2.1 – Принцип роботи моделі seq2seq [17]

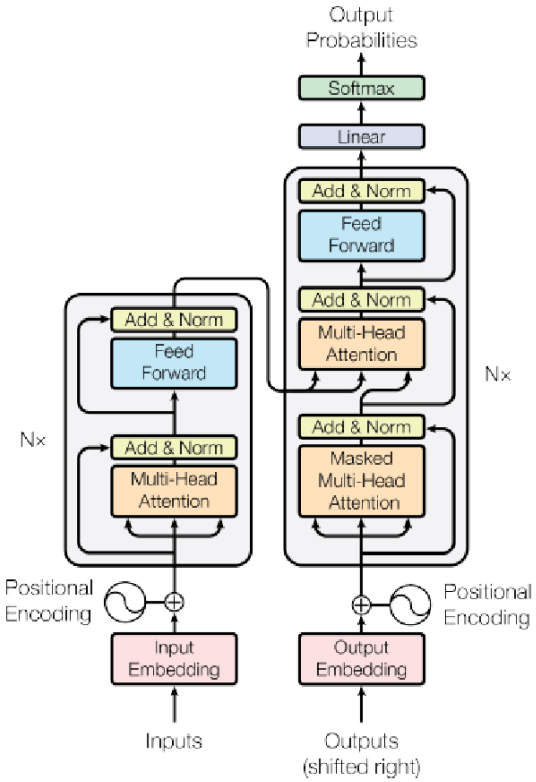


Рисунок 2.2 – Модель BART (Bidirectional and Auto-Regressive Transformers) [18]

Формально, задача абстрактивної сумаризації може бути подана як умовна генерація послідовності, формула 2.1 представлена нижче:

$$\check{y} = arg \max_{y} P(y|x), \tag{2.1}$$

де x — вхідний текст, а y — згенерований короткий виклад. Модель навчається знаходити ймовірнісне наближення до функції $P(y|x)$, що максимізує правдоподібність цільового резюме y за умовою наданого тексту x. Навчання відбувається на корпусах пар “текст–резюме” за допомогою градієнтних методів оптимізації.

Важливим аспектом автоматичного анотування є і його оцінювання, яке часто базується на порівнянні з еталонними (людськими) резюме. Для цього використовуються метрики, які будуть розглянуті в окремому підпункті, зокрема ROUGE, BLEU, BERTScore тощо. Проте точність таких метрик значною мірою залежить від якості як самих референтних даних, так і генерації моделей, зокрема коли йдеться про перекладені тексти.

Сучасне автоматичне анотування текстів у задачах сумаризації значною мірою спирається на потужні архітектури нейронного моделювання, серед яких особливу роль відіграють трансформери. Надійність і гнучкість таких моделей забезпечують можливість їхнього використання у багатоетапних NLP-сценаріях, включаючи трансляцію, парафразування та мультимовну генерацію, що є важливим для дослідження впливу перекладу на якість сумаризації.

2.2 Огляд сучасних інструментів машинного перекладу

Машинний переклад (МП) є однією з найстаріших та найбільш вивчених задач у сфері обробки природної мови. Його мета полягає у

перетворенні тексту з однієї мови на іншу, з максимально можливим збереженням лексичного, синтаксичного і семантичного значення. Протягом десятиліть підходи до автоматизованого перекладу зазнали еволюційних змін — від простих правил до статистичних методів, а згодом і до нейромережевих архітектур, які наразі є основою більшості сучасних перекладацьких систем.

Початкові підходи до МП базувалися на правилах і включали лексичні та граматичні шаблони, що вручну кодувалися лінгвістами. Ці системи були складними в підтримці, нездатними до масштабування і не мали достатньої гнучкості при роботі з реальними текстами. Пізніше, у 1990-х роках, виник підхід статистичного машинного перекладу (SMT), зокрема система Moses [7], яка працювала на основі моделей вирівнювання слів та фраз. У SMT імовірність перекладу джерельного речення x на цільове речення y оцінювалася, як показано у формулі 2.2:

$$\check{y} = arg \max_{y} P(y|x) = arg \max_{y} P(x|y) \cdot P(y), \quad (2.2)$$

де $P(y)$ — мовна модель, яка забезпечує граматичну правильність речення, а $P(x \mid y)$— модель перекладу. Для їх оцінки використовували байєсівський підхід та алгоритми максимізації очікувань (EM).

Згодом SMT витіснено нейронним машинним перекладом (NMT), який вперше набув широкого визнання після публікації Bahdanau et al. [8]. В основі NMT лежить архітектура sequence-to-sequence (seq2seq), яка навчається безпосередньо на паралельних корпусах перекладів. Модель складається з енкодера, що обробляє вхідне речення $x = (x_1, x_2, \ldots, x_n)$, і декодера, який генерує вихідне речення $y = (y_1, y_2, \ldots, y_m)$. Ймовірність послідовності перекладу обчислюється як добуток умовних ймовірностей, що представлено у формулі 2.3:

$$P(y|x) = \prod_{t=1}^{m} P(y_t|y_{<t},\ x). \quad (2.3)$$

Ключовим удосконаленням цієї моделі став механізм уваги (attention), який дає змогу декодеру фокусуватися на релевантних частинах вхідного речення під час генерації кожного токена. Це значно покращило якість перекладу, особливо для довгих речень та складних конструкцій.

Наступним кроком у розвитку NMT стала трансформерна архітектура, запропонована Vaswani et al. [9]. Трансформери відмовилися від рекурентності на користь самоуваги (self-attention), що дозволило значно підвищити паралелізм та ефективність обчислень. У трансформері кожен токен входу взаємодіє з усіма іншими токенами через матрицю уваги. Формально, для матриць запитів Q, ключів K і значень v операція уваги визначається як у формулі 2.4:

$$\text{Attention}\,(Q,K,V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V, \tag{2.4}$$

де d_k — розмірність простору ключів. Саме трансформери стали фундаментом для багатьох сучасних систем перекладу, таких як MarianNMT [10], Google Translate, Facebook Fairseq, DeepL тощо.

MarianNMT, наприклад, є високопродуктивним інструментом машинного перекладу, написаним на C++ з підтримкою GPU-прискорення, що забезпечує як швидкість, так і масштабованість. Цей фреймворк активно використовується у проєкті Helsinki-NLP для створення багатомовних моделей перекладу, включаючи українську мову.

Сучасні інструменти також використовують попереднє навчання на великих корпусах і подальше тонке налаштування (fine-tuning) для конкретних мовних пар. Це уможливлює ефективно переносити знання між мовами, що особливо корисно для мов з обмеженими ресурсами. Крім того, багатомовні моделі на кшталт mBART чи mT5 дозволяють перекладати між десятками мов в одному архітектурному середовищі.

Не менш важливою є задача оцінки якості машинного перекладу. Однією з базових метрик є BLEU (Bilingual Evaluation Understudy), яка обчислює модифіковану точність збігів n-грам між машинним перекладом та одним або кількома референтними перекладами представлено у формулі 2.5:

$$\text{BLEU} = \text{BR} \cdot \exp(\sum_{n=1}^{N} w_n \log p_n), \quad (2.5)$$

де p_n — модифікована точність n-грам, w_n — вагові коефіцієнти, а BP — пенальті за короткі речення. Хоча BLEU є стандартом де-факто, вона часто критикується за чутливість до варіацій формулювань [13].

Тому, сучасні інструменти машинного перекладу — це складні багаторівневі системи, що базуються на трансформерах, навчаються на великих багатомовних корпусах, використовують увагу і тонке налаштування, а також забезпечують високу якість перекладу навіть для малоресурсних мов. Їх інтеграція в інші NLP-завдання, зокрема в автоматичну сумаризацію, відкриває нові перспективи дослідження впливу перекладу на якість анотацій.

Окрім теоретичного аналізу, доцільно здійснити порівняння основних сучасних інструментів машинного перекладу за ключовими характеристиками. Це дозволить нам краще зрозуміти їхні можливості, обмеження та сфери застосування в контексті автоматичного анотування текстів і сумаризації. У таблиці 2.1 показано порівняння популярних систем машинного перекладу: Moses (SMT), OpenNMT, MarianNMT, Helsinki-NLP, Google Translate та DeepL за критеріями архітектури, підтримки мов, відкритості, якості перекладу та швидкодії.

Як можна побачити з таблиці, статистичні підходи, такі як Moses, істотно поступаються нейромережевим системам за точністю перекладу та продуктивністю. MarianNMT, що є основою Helsinki-NLP, забезпечує оптимальний баланс між відкритістю, масштабованістю та якістю. Пропрієтарні системи, як Google Translate та DeepL, демонструють найвищі

результати, однак є закритими і не завжди придатні для наукових досліджень через обмеження у доступі до моделей і даних. Тому для дослідницьких задач у галузі аналізу впливу перекладу на сумаризацію особливо цінними є відкриті системи з підтримкою багатьох мов і можливістю тонкого налаштування, такі як MarianNMT та Helsinki-NLP.

Таблиця 2.1 – Порівняння сучасних систем машинного перекладу

Інструмент	*Тип архітектури*	*Підтримка мов*	*Відкритий код*	*Якість перекладу (BLEU*)*	*Швидкодія*
Moses	Статистичний (SMT)	>40	Так	~20–25	Низька
OpenNMT	Seq2Seq / Transformer	>50	Так	~30–35	Середня
MarianNMT	Transformer	>100 (залежить від моделі)	Так	~35–40	Висока (GPU)
Helsinki-NLP	mBART / MarianNMT	>100	Так	~30–40	Висока
Google Translate	Пропрієтарний Transformer	>130	Ні	~45+	Висока
DeepL	Пропрієтарний Transformer+	~30 (висока якість)	Ні	~50+	Висока

2.3 Обґрунтування вибору корпусу, моделей і метрик

Для досягнення цілей дослідження ретельно підібрано корпус текстів, моделі для автоматичного реферування та машинного перекладу, а також метрики для їх оцінки. Усі ці компоненти взаємопов’язані та визначають якість, достовірність і практичну значущість результатів.

У якості корпусу обрано CNN/DailyMail — широко використовуваний датасет, який містить новинні статті разом із відповідними людськими рефератами. Цей корпус дає змогу проводити повноцінне навчання та оцінку моделей реферування, оскільки має достатню кількість прикладів, охоплює різноманітні теми, а також є загальноприйнятим стандартом у дослідженнях

текстової генерації. Його використання також забезпечує порівнянність результатів із раніше проведеними дослідженнями в галузі [3, 5].

У якості моделі для автоматичного узагальнення текстів обрано модель BART (Bidirectional and Auto-Regressive Transformers) [3]. Вона поєднує в собі властивості автокодерів і трансформерів, дозволяючи ефективно працювати як з маскуванням, так і з автогресивним генератором. BART попередньо навчена на задачах відновлення пошкодженого тексту, що дає їй змогу досягати високої якості узагальнення. Порівняно з іншими моделями (наприклад, Pointer-Generator [4] або PEGASUS [5]), BART забезпечує кращий баланс між гнучкістю генерації та збереженням важливої інформації з оригінального тексту. Також вона добре інтегрується з Hugging Face Transformers API, що спрощує реалізацію пайплайнів.

Для машинного перекладу використано модель Helsinki-NLP, зокрема підмодель, реалізована на основі MarianMT [10]. MarianMT — це оптимізований фреймворк для нейромережевого перекладу, який підтримує понад сотню мовних пар та забезпечує прийнятну якість навіть для таких мовних комбінацій, як англійська↔українська. Враховуючи завдання перекладу новинних текстів і рефератів без втрати контексту та стилістичних ознак, саме нейромережевий підхід, заснований на трансформерах [9], є найбільш ефективним.

Щодо метрик оцінки якості, обрано три: ROUGE, BLEU та BERTScore. ROUGE (Recall-Oriented Understudy for Gisting Evaluation) [14] є найпоширенішою метрикою в задачах реферування, оскільки оцінює перетини між згенерованим і референтним текстами на рівні n-грам. BLEU (Bilingual Evaluation Understudy) [12] традиційно використовується в оцінці машинного перекладу, але також застосовується для оцінки узагальнення — зокрема в ситуаціях, де важливо зберегти точне формулювання. Водночас, обидві ці метрики базуються на лексичних збігах, що є їхнім недоліком у випадках, коли текст перефразовується. Тому додатково включено BERTScore [16], яка базується на контекстуальних векторних

представленнях, що уможливлює оцінювання семантичної подібності навіть при відсутності точних збігів у формулюваннях.

Комбінація ROUGE, BLEU та BERTScore дає змогу охопити як лексичні, так і семантичні аспекти якості текстів після перекладу й узагальнення. Кожна з цих метрик акцентує увагу на різних аспектах відповідності, що уможливлює комплексно оцінити вплив машинного перекладу на якість реферування.

У таблиці 2.2 представлено коротке порівняння обраних моделей і метрик, що демонструє їхні основні характеристики в контексті завдання дослідження:

Таблиця 2.2 – Порівняння обраних моделей та метрик оцінки

Компонент	*Назва*	*Тип*	*Переваги*	*Недоліки*
Корпус	CNN / DailyMail	Новини	Великий обсяг, реальні людські реферати	Обмежений стилістикою новин
Модель реферування	BART	Абстрактивна	Висока якість генерації, трансформерна архітектура	Вимоглива до ресурсів
Перекладач	Marian MT	Нейромереже-вий	Якісний переклад, підтримка рідкісних мов	Можливі втрати стилістичних особливостей
Метрика	ROUGE	Лексична	Простота реалізації, швидкість	Не враховує синонімію
Метрика	BLEU	Лексична	Відома, стандарту MT	Надто чутлива до варіацій формулювань
Метрика	BERT Score	Семантична	Враховує контекст та синонімію	Вища обчислюваль-на складність

З урахуванням усіх зазначених факторів, саме такий вибір корпусу, моделей і метрик уможливлює максимально повно дослідити вплив машинного перекладу на якість автоматичного реферування, зберігаючи баланс між точністю, продуктивністю та семантичною глибиною аналізу.

Висновки до розділу 2

У другому розділі детально розглянуто теоретичні та практичні аспекти побудови експериментального пайплайну для оцінки впливу машинного перекладу на результати автоматичного анотування текстів у задачах сумаризації. Спочатку визначено ключові підходи до автоматичного реферування, зокрема екстрактивне та абстрактивне узагальнення, а також проаналізовано переваги останнього в контексті створення змістовних та перефразованих рефератів, що відображають глибинний зміст оригіналу. Розглянуто архітектурні особливості сучасних моделей узагальнення, серед яких значну увагу приділено трансформерним моделям, зокрема BART, що демонструє провідні результати у відповідних задачах.

Окрему увагу приділено сучасним інструментам машинного перекладу. У цьому контексті здійснено огляд розвитку технологій від статистичних до нейромережевих підходів, з акцентом на важливість якісного перекладу при роботі з автоматично згенерованими текстами. Обґрунтовано доцільність використання нейромережевих перекладачів, зокрема MarianMT, який базується на архітектурі трансформера і підтримує широкий спектр мов, включно з українською.

На завершення, обґрунтовано вибір корпусу CNN/DailyMail, що широко використовується у дослідженнях із сумаризації, а також підібрано відповідні моделі та метрики для оцінки якості результатів. Обрані метрики — ROUGE, BLEU та BERTScore — дозволяють оцінити тексти з різних точок зору: від лексичних збігів до семантичної відповідності. Такий комплексний підхід дає змогу об’єктивно проаналізувати, як машинний переклад може впливати на зміст, структуру та якість автоматично згенерованих рефератів.

Таким чином, сформовано надійну теоретичну й технічну основу для проведення подальших експериментів, метою яких є кількісне та якісне дослідження впливу машинного перекладу на результати абстрактивного реферування.

РОЗДІЛ 3 ЕКСПЕРИМЕНТАЛЬНИЙ АНАЛІЗ ТА ІНТЕРПРЕТАЦІЯ РЕЗУЛЬТАТІВ

3.1 Опис експериментального пайплайну та сценаріїв перекладу

У цьому підрозділі описано процес підготовки корпусу даних та здійснення необхідної передобробки, яка передувала запуску експериментального пайплайну оцінки впливу машинного перекладу на якість автоматичного анотування текстів. Для дослідження використано корпус англомовних новинних текстів, кожен з яких супроводжується короткою референтною анотацією. Кожен приклад у корпусі складався з двох частин: основного тексту новини, що подавався як суцільний абзац із кількох речень, та відповідної до нього референтної анотації у вигляді кількох тез, що узагальнювали ключові події та факти.

На першому етапі підготовки даних здійснено базову фільтрацію: видалено порожні приклади, а також ті, в яких відсутня або основна новина, або анотація. Враховано обмеження на довжину тексту — надто короткі або надто об’ємні новини виключалися, щоб забезпечити збалансованість корпусу та уникнути крайнощів, які могли б спотворити результати автоматичного узагальнення. Для зручності обробки обмежено загальну кількість прикладів, що підлягали аналізу, з урахуванням обчислювальних можливостей.

Далі проведено базову передобробку тексту. Вона включала нормалізацію лапок, апострофів і спеціальних символів, видалення зайвих пробілів, а також збереження єдиного формату подання кожного прикладу: текст новини подавався як один рядок, а референтна анотація — як окремий блок. Усі приклади зберігалися у структурованому вигляді для подальшого автоматизованого опрацювання (рис. 3.1).

```
Текст: The plot where Lizzie Borden and her family are buried in Fall River, Massachusetts,
was vandalized . The final resting place of Lizzie Borden and the father and stepmother she
was accused of killing was vandalized over the weekend. The Borden family burial monument at
Oak Grove Cemetery in Fall River, Massachusetts was defaced with black and green paint. The
vandalism was discovered on Monday, the day after a Lifetime miniseries, The Lizzie Borden
Chronicles, premiered on Sunday night. Borden went on trial for brutally murdering her father
and stepmother with a hatchet in their Fall River home in 1892. She was acquitted, but her
innocence is questioned even today. The murder and trial inspired the schoolyard rhyme:
'Lizzie Borden took an ax and gave her mother 40 whacks. 'When she saw what she had done,
she gave her father 41'. The house where the murders occurred is still standing. It is
located next to the Bristol Superior Court House, the venue where the murder trial of
former New England Patriots player Aaron Hernandez is taking place. Borden went on trial
for brutally murdering her father and stepmother with a hatchet in their home in 1892 .
The historic Lizzie Borden house, where the killings of her father and stepmother took
place, still stands today . The murder and trial surrounding Borden made her a cult figure
and led to an appearance on the Simpsons . This is not the first time the plot, which also
contains Lizzie's sister Emma, has been vandalized, the Herald News reported. Cemetery director
Tony Eaton said: 'I'd like to catch the guys who did it. 'I don't know what the reason is for
putting that on a grave stone. 'It just irks me... not just when it's Lizzie Borden, but anyone.'
 In addition to graffiti, the Borden grave has also been covered with pennies, flowers, notes,
amulets, chicken claws, feathers, blood and urine in the past. Borden, who was acquitted of
murder, died from complications following gall bladder surgery at the age of 66 . Christina
Ricci is playing Borden in a Lifetime miniseries, The Lizzie Borden Chronicles, that began on
Sunday . Historian and author Shelley Dziedzic said the start of the show may have led to the
graffiti, according to the Boston Globe. Dziedzic said: 'My thinking is it's probably related
to the Lifetime series that debuted. 'Anytime the programs are on television there's a huge influx
in visitors.' The new show, which stars Christina Ricci, is 'an intense and fictionalized account
of actual events and people surrounding Lizzie's life after her controversial acquittal of the
horrific double murder of her father and stepmother in 1892', according to Lifetime, . Ricci's
character is an 'exonerated figure' who 'lives a life awash in newfound celebrity filled with
scandalous love affairs'. Borden died from complications following gall bladder surgery at the age of 66. ...

Реф. анотація: Lizzie Borden and her family are buried in Fall River, Massachusetts .
Their plot at Oak Grove Cemetery was defaced with black and green paint .
Speculated graffiti is related to Lifetime's The Lizzie Borden Chronicles .
Borden was acquitted after going on trial for the two murders in 1892 .
Inspired rhyme: 'Lizzie Borden took an ax and gave her mother 40 whacks'
Historic murder house is adjacent to where Aaron Hernandez is on trial .
```

Рисунок 3.1 – Приклад фільтрації тексту

На цьому етапі підготовлено кілька варіантів корпусу для експерименту: базовий корпус оригінальних англомовних текстів, а також версії, в яких тексти або анотації перекладені українською мовою і, за потреби, назад англійською. Це дозволяло у подальших етапах дослідження оцінити вплив перекладу на змістовну відповідність між узагальненням та референтною анотацією. сформований корпус і здійснена передобробка забезпечили надійне підґрунтя для якісного аналізу зміни метрик якості внаслідок перекладу.

3.2 Реалізація пайплайну автоматичної сумаризації та перекладу

Перейдемо до опису реалізація експериментального пайплайну, який поєднує автоматичну сумаризацію текстів із подальшим машинним перекладом для оцінки впливу перекладу на якість узагальнення. Пайплайн побудовано у середовищі Python з використанням бібліотек для обробки природної мови та моделей глибокого навчання. Реалізація охоплює кілька послідовних етапів, які дозволяють провести порівняння результатів сумаризації за різних сценаріїв перекладу.

На першому етапі створено функціонал для автоматичного узагальнення англомовного тексту. Для цього використовувалася сучасна нейромережева модель абстрактивної сумаризації, здатна перетворити повноцінну новину у короткий зміст. На вхід моделі подавалися очищені новинні тексти, а на виході отримувалися сформовані нею узагальнення, які потім можна порівнювати з референтними.

Наступним кроком у реалізації пайплайну стало підключення моделей машинного перекладу. Реалізовано переклад як самих текстів новин, так і згенерованих анотацій у різних напрямках: з англійської на українську, з української назад на англійську, а також переклад лише анотацій або лише основного тексту. Це дозволило створити кілька окремих сценаріїв експерименту: (1) пряме узагальнення англомовного тексту, (2) узагальнення після перекладу новини українською, (3) оцінка узагальнення, яке перекладене з англійської на українську, а потім назад на англійську, та інші комбінації (рис. 3.2).

Для реалізації кожного етапу написані окремі функції, які автоматизували обробку корпусу, виклик відповідних моделей, збереження результатів та підготовку їх до подальшої оцінки. Особливу увагу приділено збереженню відповідності між оригінальними прикладами, перекладеними

версіями та отриманими узагальненнями, що критично важливо для подальшого обчислення метрик якості.

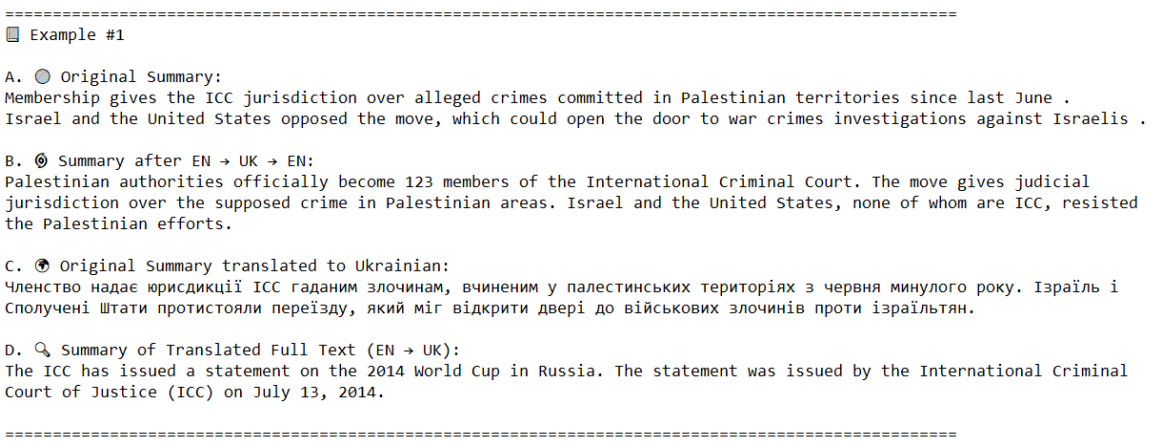


Рисунок 3.2 – Реалізація етапу сумаризації тексту

Кожен сценарій обробки виконаний на обмеженій кількості прикладів з корпусу з урахуванням ресурсомісткості моделей. Отримані результати — як згенеровані анотації, так і їх перекладені версії — зберігалися у структурованому форматі для подальшої автоматичної оцінки метриками якості. Таким чином, реалізований пайплайн забезпечив повноцінне та гнучке середовище для дослідження впливу перекладу на процес автоматичного анотування текстів.

3.3 Методика оцінювання якості узагальнень

Методика оцінювання якості узагальнень, реалізована в межах даного дослідження, базується на використанні трьох провідних автоматичних метрик — ROUGE, BLEU та BERTScore. Кожна з них виконує свою специфічну функцію в системі аналізу результатів і дає змогу отримати комплексне уявлення про точність, повноту й семантичну релевантність

згенерованих анотацій. Ці метрики є загальновизнаними у сфері обробки природної мови, оскільки дозволяють не лише об'єктивно порівнювати узагальнення між собою, а й оцінювати їхню відповідність референтним варіантам. У контексті нашого дослідження, де оцінюється вплив машинного перекладу на якість сумаризації, особливо важливо враховувати як поверхневі лексичні відповідності, так і змістовні (семантичні) збіги, які можуть зберігатися або змінюватися внаслідок трансформацій під час перекладу.

Перша метрика, ROUGE (Recall-Oriented Understudy for Gisting Evaluation), використовується для вимірювання ступеня збігу між еталонною (референтною) анотацією та узагальненням, згенерованим моделлю. Основу метрики становить підрахунок кількості однакових n-грам, що з’являються в обох текстах. У нашому випадку застосовувалися такі її варіанти: ROUGE-1 (уніграмний збіг), ROUGE-2 (біграмний збіг) та ROUGE-L (довжина найбільшої спільної підпослідовності). Формально ROUGE-1 визначається за формулою 3.1:

$$\text{ROUGE} - 1 = \frac{\sum_{n-gram \in Reference} \min(Count_{gen},\ Count_{ref})}{\sum_{n-gram \in Reference} Count_{ref}}, \qquad (3.1)$$

де $Count_{gen}$ — кількість появ n-грам у згенерованому тексті, а $Count_{ref}$ — кількість появ цих же n-грам у референтному тексті [14]. ROUGE є чутливою до порядку слів і добре ілюструє, наскільки узагальнення відтворює ключові одиниці тексту, однак не завжди здатна охопити семантичну відповідність, особливо при синонімії або парафразах.

Метрика BLEU (Bilingual Evaluation Understudy), хоч і спочатку розроблялась для оцінки якості машинного перекладу, широко використовується і в задачах сумаризації, особливо коли потрібно порівнювати лексичні структури різних варіантів узагальнень. BLEU розраховує точність збігів n-грам і включає штраф за занадто короткі

узагальнення, що стимулює моделі генерувати повніші фрази. Її математична формула 3.2 має вигляд:

$$\text{BLEU} = \text{BR} \cdot \exp\left(\sum_{n=1}^{N} w_n \cdot \log p_n\right), \tag{3.2}$$

де p_n — точність n-грам (частка збігів), w_n — ваги для кожного порядку n-грам (зазвичай рівні), а BR — штраф за стислість (brevity penalty), який визначається як у системній формулі 3.3:

$$\text{BR} = \begin{cases} 1, & \text{якщо } c > r \\ e^{\left(1-\frac{r}{c}\right)}, & \text{якщо } c > r \end{cases}, \tag{3.3}$$

де c — довжина згенерованого тексту, а r — довжина референтного [15]. Таким чином, BLEU більш строго підходить до оцінки точності формулювань і корисний для аналізу тих змін, які вносяться в результаті перекладу, навіть якщо вони зберігають основну ідею.

Для подолання обмежень, пов’язаних із чутливістю ROUGE та BLEU до лексичних розбіжностей, до аналізу включено метрику BERTScore. Ця метрика базується на використанні трансформерної моделі BERT для порівняння глибинних контекстуальних векторів кожного токена у згенерованому тексті та референтній анотації. Вона використовує косинусну подібність між ембеддингами слів, що уможливлює виявляти семантичну близькість навіть у тих випадках, коли формальні збіги між словами відсутні. У формулі 3.4 BERTScore розраховує Precision, Recall і F1-міру, що робить її надійним інструментом для оцінки змістовної відповідності:

$$\text{BERTScore} - F_1 = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall}. \tag{3.4}$$

Завдяки своїй архітектурі BERTScore дає змогу аналізувати узагальнення з точки зору глибокої смислової відповідності, що особливо

важливо у випадках, коли переклад змінює поверхневу структуру, але зберігає суть [16].

Усі зазначені метрики реалізовано з використанням бібліотеки evaluate від HuggingFace, що забезпечує зручний та уніфікований інтерфейс для оцінювання результатів. Ця реалізація уможливлює одночасно обробляти великі масиви результатів, підраховувати середні значення, будувати розподіли та готувати дані до подальшого візуального аналізу. Кожен варіант обробки даних — пряме узагальнення англомовного тексту, узагальнення після прямого перекладу на українську, а також узагальнення після перекладу з української назад на англійську — оцінювався окремо для того, щоб порівняти якість генерованих анотацій у кожному з експериментальних сценаріїв.

Методика оцінювання якості узагальнень у цьому дослідженні є багаторівневою та всебічною: вона поєднує класичні підходи на основі збігів n-грам і сучасні методи аналізу семантичної відповідності, що уможливлює глибоко оцінити вплив машинного перекладу на результати сумаризації з різних боків — від формального до змістового рівня.

3.4 Порівняльний аналіз результатів для різних сценаріїв перекладу

У цьому підрозділі здійснено порівняльний аналіз результатів для трьох різних сценаріїв перекладу, які реалізовані для оцінки впливу машинного перекладу на якість анотацій. Для кожного з 20 прикладів отримано три типи обробки: по-перше, резюме, яке перекладене з англійської на українську, а потім назад на англійську (EN→UK→EN); по-друге, оригінальне резюме, перекладене з англійської на українську (EN→UK summary); по-третє, резюме, згенероване на основі повного тексту, який

попередньо перекладено з англійської на українську (EN→UK full text → summary). Усі результати порівняні з оригінальними англомовними резюме за допомогою ROUGE-метрик, що дозволило оцінити збереження лексичних і структурних елементів.

Аналіз отриманих метрик , які ми спостерігаємо на рисунку 3.3 продемонстрував суттєву різницю між трьома підходами.

```
================================================================================================================
📊 ROUGE Scores for: EN→UK→EN Summary
rouge1: 0.2290
rouge2: 0.0702
rougeL: 0.1658
rougeLsum: 0.1897
================================================================================================================
📊 ROUGE Scores for: EN→UK Summary→UK
rouge1: 0.0597
rouge2: 0.0042
rougeL: 0.0597
rougeLsum: 0.0604
================================================================================================================
📊 ROUGE Scores for: Summary of (EN→UK) Full Text
rouge1: 0.0659
rouge2: 0.0016
rougeL: 0.0517
rougeLsum: 0.0589
```

Рисунок 3.3 – Аналіз ROUGE-метрик

Найкращі показники спостерігалися у випадку обробки EN→UK→EN, де середнє значення ROUGE-1 становило 0.2290, ROUGE-2 — 0.0702, ROUGE-L — 0.1658, а ROUGE-Lsum — 0.1897. Це свідчить про те, що при зворотному перекладі більшість ключових лексичних елементів та структура резюме зберігаються відносно добре. Хоча відбуваються певні втрати, вони не є критичними, і загальний зміст залишається наближеним до оригінального. Збереження біграм (ROUGE-2) також вказує на те, що в таких випадках переклад не сильно впливає на локальну когерентність тексту.

Зовсім інша картина спостерігається у другому сценарії — EN→UK summary. У цьому випадку оцінювання проводилося між англійським оригіналом і його перекладом українською мовою, що автоматично обумовлює низькі значення ROUGE. ROUGE-1 дорівнює лише 0.0597,

ROUGE-2 — 0.0042, ROUGE-L — 0.0597, а ROUGE-Lsum — 0.0604. Такий результат є очікуваним, адже метрики ROUGE не пристосовані для міжмовного порівняння, і навіть при якісному перекладі, відмінності у словоформах, синтаксисі та порядку слів значно знижують кількість збігів, які ці метрики фіксують. Проте навіть на інтуїтивному рівні видно, що інформаційне наповнення між оригінальним резюме та його перекладом у цьому випадку частково зберігається, але лексична форма не дає змогу це відобразити через ROUGE.

Третій сценарій — створення резюме з перекладеного українською повного тексту — продемонстрував ще нижчі результати. ROUGE-1 тут становив 0.0659, ROUGE-2 — лише 0.0016, ROUGE-L — 0.0517, а ROUGE-Lsum — 0.0589. Попри те, що оцінювання відбувалося між текстами однією мовою (англійською), метрики виявилися нижчими, ніж у випадку прямого перекладу резюме. Це пояснюється тим, що процес перекладу повного тексту, а потім узагальнення вже українського варіанту, значно змінює структуру, акценти та лексичні вибори. Отримане резюме, хоч і перебуває в тій самій мовній площині, має дуже низький ступінь подібності до оригінального англомовного резюме. Зміни у викладі думок, синтаксичних структурах та використаних термінах призводять до істотних втрат у метриках, особливо у ROUGE-2, яка майже нульова. Це свідчить про порушення цілісності контексту та втрату локальних зв’язків, які важливі для формування точного резюме.

Результати експерименту показують, що найменш руйнівним для якості резюме є сценарій зворотного перекладу (EN→UK→EN), де інформаційні та структурні втрати залишаються мінімальними. Переклад резюме на іншу мову або створення нового резюме з перекладеного повного тексту дає значно гірші результати за ROUGE-метриками, що свідчить про втрату ключової інформації та спотворення оригінального змісту. Це підкреслює важливість обережного використання машинного перекладу у завданнях, пов’язаних із текстовою анотацією, особливо коли мова йде про

міжмовні сценарії або про створення нових текстів на основі перекладених джерел.

3.5 Візуалізація, інтерпретація результатів та лінгвістичні спостереження

Візуалізація та інтерпретація отриманих результатів дає змогу глибше зрозуміти вплив різних сценаріїв машинного перекладу на якість текстових анотацій, а також висвітлити супутні лінгвістичні особливості, пов'язані з трансформацією тексту.

Графіки ROUGE-мір показують суттєву різницю між сценаріями (рис. 3.4).

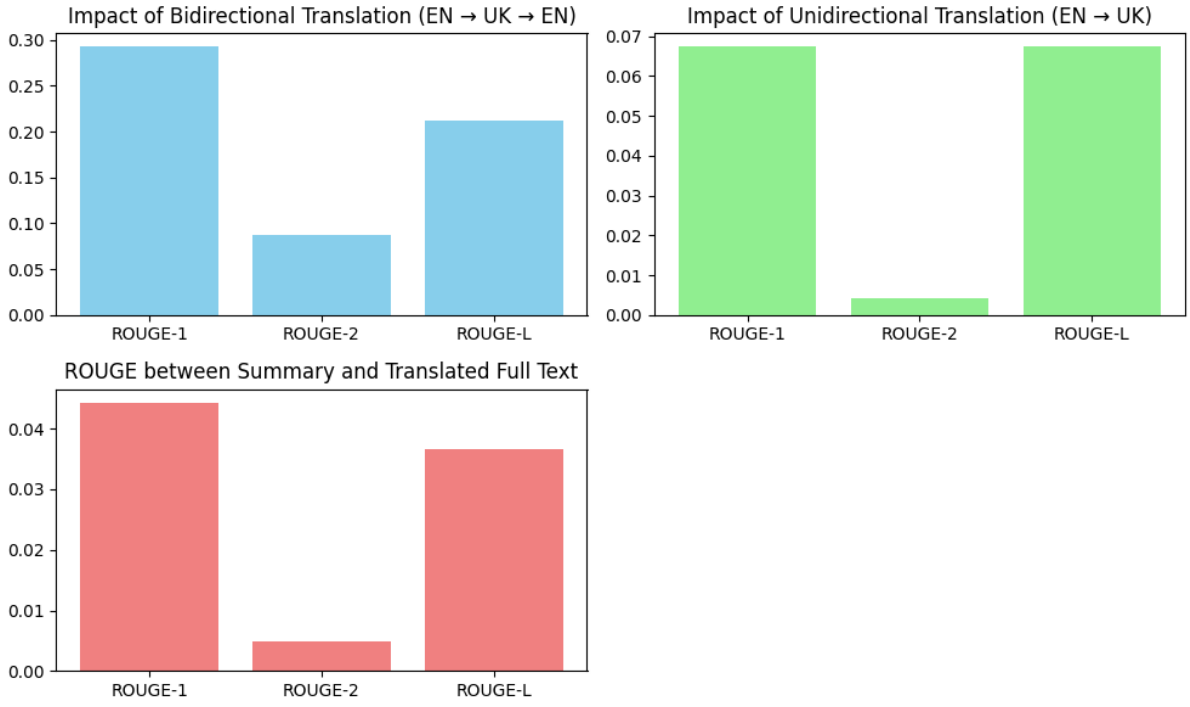


Рисунок 3.4 – Графіки ROUGE-метрик

Найвищі значення спостерігаються для бінаправленого перекладу (EN→UK→EN), де ROUGE-1 становить 0.2290, ROUGE-2 — 0.0702, а ROUGE-L — 0.1658. Це свідчить про те, що незважаючи на проходження тексту через два перекладацькі етапи, семантична структура та лексичні одиниці залишаються досить наближеними до оригіналу. Інакше кажучи, BART-модель здатна згенерувати резюме, яке зберігає подібність до оригінального, навіть після подвійного перекладу. Натомість у випадку одностороннього перекладу (EN→UK), де порівнюється англомовне резюме з його українською версією, показники різко знижуються: ROUGE-1 дорівнює лише 0.0597, ROUGE-2 — 0.0042, а ROUGE-L — 0.0597. Це свідчить про суттєві лексичні й синтаксичні розбіжності, що виникають унаслідок міжмовного переходу. Схожа картина спостерігається і для третього варіанту — коли резюме формується з перекладеного повного тексту українською мовою, а потім порівнюється з англійським оригіналом. Тут значення ROUGE-1 становить 0.0659, а ROUGE-2 — 0.0016, що ще раз підкреслює серйозні втрати при трансформації змісту через переклад повного тексту.

Дані BLEU-метрики представлено на рисунку 3.5.

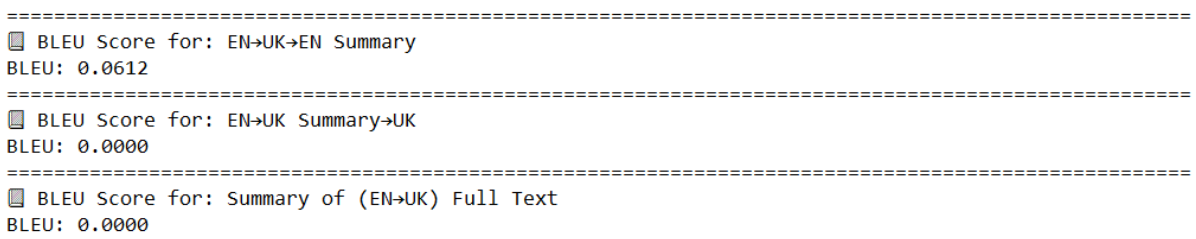


Рисунок 3.5 – Результати BLEU-метрик

Вони підтверджують ці висновки: лише бінаправлений переклад дає ненульовий результат — BLEU дорівнює 0.0612, тоді як обидва інші сценарії

(EN→UK summary та summary з перекладеного тексту) мають BLEU 0.0000. Це означає майже повну відсутність точного лексичного збігу між оригінальними та перетвореними резюме, що логічно з огляду на міжмовну природу змін.

Натомість BERTScore демонструє дещо іншу картину, оскільки враховує семантичну близькість, а не лише лексичну (рис. 3.6).

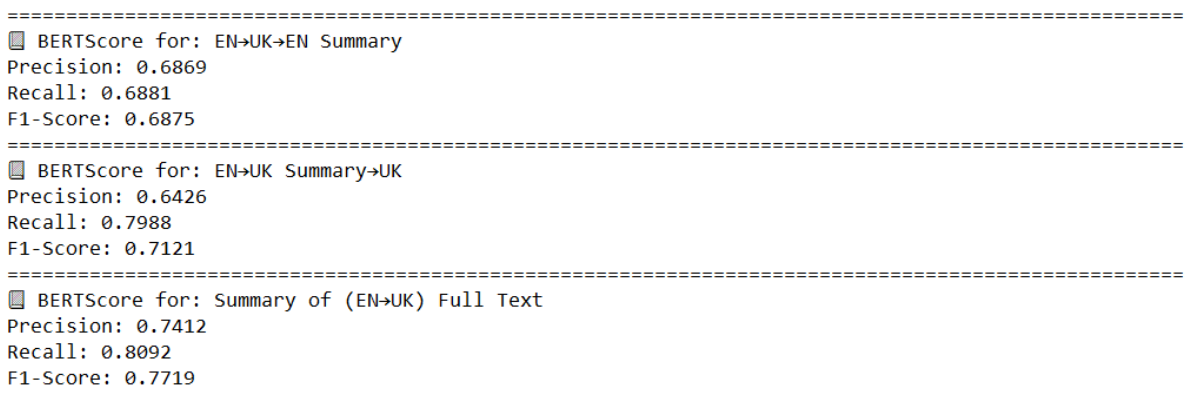


Рисунок 3.6 – Результати BERTScore

Найвищий F1-результат спостерігається у третьому сценарії (summary з перекладеного повного тексту) — 0.7719, що вказує на те, що попри лексичну відмінність, загальний зміст резюме залишається відносно близьким до оригіналу. Друге місце займає EN→UK summary (F1 = 0.7121), а найнижчий показник має саме EN→UK→EN summary (0.6875). Це дещо парадоксально, оскільки ROUGE-метрики демонструють протилежну тенденцію. Така розбіжність пояснюється тим, що BERTScore опирається на глибокі контекстуальні подібності, тоді як ROUGE і BLEU — на формальну текстову структуру.

Додаткове лінгвістичне спостереження ґрунтується на аналізі частин мови (POS-tagging). Результати свідчать про те, що бінаправлений переклад

(EN→UK→EN) зберігає граматичну структуру речень найбільш близькою до оригіналу (рис. 3.7).

	NOUN	VERB	DET	PROPN	ADP	ADJ	SCONJ	PUNCT	SPACE	\
Original	8	4	2	5	4	2	0	5	1	
EN→UK→EN	9	5	4	6	5	2	0	6	0	
From Translated Text	4	1	1	8	1	0	0	12	4	

	CCONJ	PRON	AUX	ADV	NUM	PART	X
Original	0	1	2	1	0	0	0
EN→UK→EN	1	2	2	0	1	1	0
From Translated Text	0	0	0	0	0	0	0

Рисунок 3.7 – Результати аналізу частин мови (POS-tagging)

Наприклад, кількість іменників (NOUN) і власних назв (PROPN) навіть зросла, що може свідчити про збагачення змісту або зміну стилістики при перекладі. Водночас у резюме, створеному з перекладеного повного тексту, спостерігається значне спрощення лексико-граматичної структури: майже повна відсутність дієслів, прикметників та займенників. Це може свідчити про втрату контексту, узагальнення або механічний стиль генерації, що характерний для таких перекладних текстів.

Загалом візуалізація та багаторівневий аналіз результатів показують, що вплив машинного перекладу на якість анотації залежить не лише від кількості перекладацьких етапів, а й від того, яка саме частина процесу (повний текст чи резюме) підлягає трансформації. Найменш руйнівним для структури та змісту тексту виявився сценарій EN→UK→EN, хоча і він не є повністю нейтральним. Найбільше втрат інформації й лексичної цілісності спостерігається при генерації резюме з перекладеного повного тексту, що вказує на необхідність обережного використання перекладних даних у задачах автоматичного реферування.

Висновки до розділу 3

У третьому розділі здійснено повноцінний експериментальний аналіз, спрямований на оцінювання впливу машинного перекладу на якість автоматично згенерованих резюме. Першочергово сформовано корпус із 20 прикладів на базі датасету CNN/DailyMail, який піддано низці трансформацій. Проведено ретельну передобробку текстових даних, включно з токенізацією, нормалізацією та форматуванням для подальшої обробки в пайплайні.

На наступному етапі реалізовано комплексний пайплайн, що включав три основні компоненти: генерацію резюме англомовною моделлю BART, переклад повного тексту або резюме за допомогою моделі машинного перекладу (Helsinki-NLP) та порівняння результатів з оригіналом. Оцінювання виконувалося за кількома незалежними сценаріями: бінаправлений переклад резюме (EN→UK→EN), односторонній переклад самого резюме (EN→UK), а також створення резюме на основі перекладеного повного тексту.

Методика оцінювання якості узагальнень включала три групи метрик: ROUGE (для аналізу перекриття фраз), BLEU (для лексичного збігу) та BERTScore (для оцінки семантичної подібності). Доповненням до цього аналізу стали лінгвістичні спостереження на основі POS-тегування, які дозволили виявити зміни в граматичній структурі речень після перекладу.

Порівняльний аналіз продемонстрував, що найвищі показники ROUGE і BLEU спостерігаються для бінаправленого перекладу резюме (EN→UK→EN), що свідчить про мінімальні втрати лексичної подібності при такій трансформації. Значно гірші результати зафіксовано у випадках одностороннього перекладу та особливо при створенні резюме з перекладеного повного тексту, де спостерігався майже повний розрив між змістом і формою. Водночас BERTScore показав цікаву тенденцію:

незважаючи на низькі формальні збіги, семантична подібність залишалась на високому рівні, особливо при генерації резюме на основі перекладеного тексту, що може свідчити про збереження загального змісту при суттєвій зміні мовного оформлення.

Візуалізація результатів у вигляді графіків дозволила наочно продемонструвати різницю між сценаріями та підкреслити тенденції, які важко помітити лише з числових значень. Лінгвістичний аналіз на основі частин мови показав, що бінаправлений переклад зберігає структурну складність і лексичне багатство тексту, тоді як резюме з перекладеного тексту демонструє значне спрощення граматичної побудови та втрату лексичних категорій.

Проведене дослідження підтвердило, що вплив машинного перекладу на автоматичне узагальнення не є однозначним і залежить як від напрямку перекладу, так і від того, на якому етапі процесу здійснюється трансформація. Найменші втрати якості спостерігаються при бінаправленому перекладі коротких текстів (резюме), тоді як найгірші результати демонструє створення узагальнень на основі перекладеного повного тексту. Це дає змогу зробити практичні висновки щодо обережного використання машинного перекладу в задачах багатоетапної обробки тексту, де важливо зберігати точність, смислову цілісність і лексичну відповідність.

.

РОЗДІЛ 4 ФУНКЦІОНАЛЬНО-ВАРТІСНИЙ АНАЛІЗ СИСТЕМИ ОЦІНКИ ВПЛИВУ МАШИННОГО ПЕРЕКЛАДУ НА ЯКІСТЬ АНОТУВАННЯ

У цьому розділі буде здійснено функціонально-вартісний аналіз (ФВА) створеної системи оцінки впливу машинного перекладу на якість автоматичного анотування текстів, зосередженої на процесах машинного перекладу та автоматичної генерації резюме. Розроблена система передбачає інтеграцію сучасних моделей глибокого навчання, зокрема трансформерних архітектур для узагальнення текстів (на прикладі BART) і моделей типу MarianMT/Helsinki-NLP для перекладу між англійською та українською мовами. Основна увага приділяється розробці програмного рішення, яке забезпечує повноцінний експериментальний пайплайн — від збору даних до візуалізації метрик оцінки якості.

Функціонально-вартісний аналіз дає змогу оцінити доцільність створення та використання подібної системи як з технічної, так і з економічної точки зору. Він дозволяє визначити ключові функції, які виконує система, співвіднести їх із вартістю ресурсів, витрачених на реалізацію кожної функції, та оцінити загальну ефективність розробки. Це особливо важливо для дослідницьких і навчальних проектів, у яких поєднується складна технічна реалізація, обчислювальні ресурси та необхідність отримання достовірних результатів для аналітичних висновків.

ФВА буде проведено з урахуванням як технічних аспектів (вибір моделей, обчислювальна складність, ефективність реалізованого пайплайну), так і організаційних (необхідність фахових знань, часові витрати на підготовку корпусу, запуск експериментів, обробку результатів). Крім того, буде розглянуто потенційні шляхи оптимізації системи — як з точки зору зниження витрат, так і в напрямку підвищення точності оцінювання та автоматизації процесів. У підсумку аналіз дозволить зробити висновки про

функціональну повноту розробленого рішення, доцільність його використання в подібних дослідженнях, а також окреслити можливості подальшого розвитку та впровадження в інші задачі обробки природної мови.

4.1 Постановка задачі проектування

У даній роботі функціонально-вартісний аналіз застосовується для техніко-економічного обґрунтування створеної системи, призначеної для оцінки впливу машинного перекладу на якість автоматичного анотування текстів. Це дослідницьке середовище поєднує в собі етапи автоматичної генерації резюме за допомогою сучасних трансформерних моделей, виконання перекладу в одному або двох напрямках (англійська ↔ українська), а також обчислення й візуалізацію метрик якості узагальнення. У зв’язку з цим завдання проєктування охоплює створення гнучкої, масштабованої та відтворюваної системи, яка дозволяє швидко змінювати конфігурації експериментів, додавати нові моделі, змінювати мови перекладу та адаптувати метрики під різні дослідницькі цілі.

Оскільки якість аналізу залежить від взаємодії всіх компонентів системи — починаючи від збору корпусу й закінчуючи інтерпретацією отриманих результатів — важливо, щоб кожна підсистема узгоджувалась із загальними вимогами до точності, швидкості та ефективності. Проєктування відбувалося з урахуванням комплексної реалізації усіх необхідних функцій: автоматизоване завантаження і підготовка даних, виклик попередньо навчених моделей перекладу та сумаризації, обчислення таких метрик як ROUGE, BLEU, BERTScore, а також статистичний і лінгвістичний аналіз результатів (наприклад, POS-tagging).

Основні технічні та функціональні вимоги до ПЗ включають:

- сумісність із персональними комп’ютерами та підтримка виконання в середовищах Google Colab або локальних Python-оточеннях;
- забезпечення стабільної роботи при обробці десятків прикладів із повноцінним циклом сумаризації та перекладу;
- можливість масштабування системи до більших корпусів без потреби в суттєвому переписуванні коду;
- простота у використанні й модифікації (наприклад, через параметризовані скрипти або конфігураційні файли);
- доступність та відкритість використаних інструментів (використання моделей із відкритих бібліотек Hugging Face);
- мінімізація витрат часу та ресурсів на повторне проведення експериментів;
- можливість візуалізації результатів у зручному форматі (графіки, таблиці).

4.2 Обґрунтування функцій програмного продукту

Головна функція F_0 — розробка програмного продукту, який дозволяє здійснювати комплексну оцінку впливу машинного перекладу на якість анотування текстів шляхом реалізації пайплайну, що включає автоматичну сумаризацію, переклад текстів у різних напрямках, обчислення метрик якості узагальнення (ROUGE, BLEU, BERTScore), а також проведення лінгвістичного аналізу (наприклад, за допомогою POS-tagging). Така система має на меті забезпечити наукову обґрунтованість результатів і гнучкість при проведенні експериментів із перекладом і сумаризацією.

Взявши за основу дану головну функцію, можна виокремити наступні підфункції:

1) F_1 – вибір мови програмування;
2) F_2 – вибір бібліотек для реалізації NLP-моделей;
3) F_3 – вибір середовища розробки.

Кожна функція має декілька варіантів реалізації.

Функція F_1.

1. Python.
2. R.

Функція F_2.

1. Hugging Face Transformers.
2. Fairseq.

Функція F_3.

1. Google Colab.
2. Jupyter Notebook.

Кожна функція має декілька можливих варіантів реалізації. Варіанти реалізації основних функцій наведені у морфологічній карті системи (рис. 4.1).

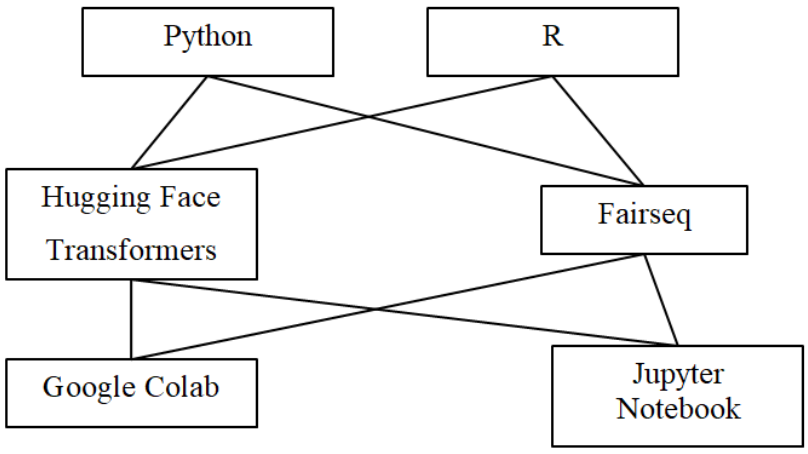


Рисунок 4.1 — Морфологічна карта системи

На основі морфологічної карти побудуємо позитивно-негативну матрицю, яка наведена у таблиці 4.1.

Таблиця 4.1 — Позитивно-негативна матриця

Функція	*Варіант реалізації*	*Переваги*	*Недоліки*
F_1	1	Величезна екосистема для NLP, підтримка сучасних моделей, активна спільнота	Високе навантаження на ресурси, складність масштабування великих проєктів
	2	Добре підходить для статистичного аналізу, потужні засоби візуалізації	Обмежена підтримка сучасних моделей глибокого навчання
F_2	1	Простота інтеграції моделей сумаризації та перекладу, великий вибір моделей	Вимоги до ресурсів GPU, складність налаштувань для кастомізації
	2	Висока продуктивність при обробці тексту, ефективність у перекладі	Менше готових моделей, слабша підтримка сумаризації
F_3	1	Хмарне середовище, доступність GPU, не потребує локальної установки	Обмеження за часом сесії, залежність від підключення до інтернету
	2	Зручне середовище для аналізу даних, підтримка візуалізацій	Обмежена функціональність для складних проєктів

При аналізі позитивно-негативної матриці можна зробити такі висновки щодо вибору функціональних рішень.

Функція F_1: оптимальним є варіант 1 — Python, завдяки великій кількості бібліотек та потужним можливостям для реалізації NLP-проєктів. Варіант 2 — R — хоч і добре підходить для статистичного аналізу, але має обмежену підтримку сучасних трансформерних моделей, тому надалі не враховується.

Функція F_2: найбільш доцільним є варіант 1 — Hugging Face Transformers, який забезпечує комплексну підтримку як перекладу, так і

сумаризації, а також активне оновлення та документацію. Варіант 2 — Fairseq — може бути використаний у спеціалізованих випадках, але має обмежену кількість готових моделей.

Функція F_3: доцільно зберегти обидва варіанти. Google Colab ідеально підходить для хмарної роботи з GPU, експериментів та швидкого прототипування. Jupyter Notebook є хорошим вибором для локального середовища з інтерактивною візуалізацією та аналізом.

Таким чином, до подальшого розгляду беруться такі варіанти реалізації програмного продукту (ПП):

$$F_1 1 - F_2 1 - F_3 1,$$
$$F_1 1 - F_2 1 - F_3 2.$$

Ці варіанти забезпечують оптимальний баланс між функціональністю, ефективністю та зручністю для реалізації програмного забезпечення, здатного проводити глибокий аналіз впливу машинного перекладу на якість анотацій у задачах автоматичного узагальнення текстів.

4.3 Обґрунтування системи параметрів програмного продукту

На основі попереднього аналізу функцій програмного продукту та обраних варіантів їх реалізації формуються ключові параметри, які відображають рівень якості, продуктивності та технічної доцільності реалізованої системи. Дані параметри необхідні для подальшого розрахунку коефіцієнта технічного рівня програмного забезпечення, що дозволяє визначити оптимальність обраного підходу з позиції ефективності та ресурсозатрат.

Для цілей кількісної оцінки доцільно використовувати такі параметри:

1. X_1 — ефективність виконання коду (швидкодія). Визначається кількістю операцій, що можуть бути оброблені за одиницю часу. Цей параметр прямо впливає на загальну продуктивність системи, особливо при роботі з великими текстовими корпусами.
2. X_2 — обсяг необхідної оперативної пам’яті. Характеризує вимоги до апаратного забезпечення, особливо у процесах перекладу, сумаризації та обчислення метрик.
3. X_3 — тривалість навчання моделей. Визначає час, потрібний для повного навчального циклу моделі, що дозволяє оцінити ефективність обчислювального процесу.
4. X_4 — обсяг програмного коду. Кількість рядків коду, яка дозволяє оцінити складність розробки, підтримки та масштабування системи.

Значення кожного параметра диференціюються на три рівні – гірший, середній та кращий, що відображають можливі варіанти реалізації з точки зору ефективності та доцільності використання, показано у таблиці 4.2.

Таблиця 4.2 – Значення системи параметрів для оцінки технічного рівня

Назва параметра	*Умовні позначення*	*Одиниці виміру*	*Значення параметра*		
			Гірші	*Середні*	*Кращі*
Швидкодія мови програмування	X1	оп/мс	3000	9000	14000
Об’єм пам’яті	X2	Гб	6	10	18
Час навчання моделей	X3	с	1500	800	300
Обсяг програмного коду	X4	Кількість рядків коду	1800	1100	700

4.4 Аналіз експертного оцінювання

Після детального обговорення та аналізу кожний експерт оцінює ступінь важливості кожного параметра для конкретної цілі — розробка програмного продукту, який забезпечує максимальну точність та ефективність при виконанні завдань машинного перекладу, аналізу тексту та оцінювання результатів за допомогою різноманітних метрик.

Значимість кожного параметра визначається методом попарного порівняння. Оцінку проводить експертна комісія з 7 осіб. Визначення коефіцієнтів значимості передбачає:

- визначення рівня значимості параметра шляхом присвоєння різних рангів;
- перевірку придатності експертних оцінок для подальшого використання;
- визначення оцінки попарного пріоритету параметрів;
- обробку результатів та визначення коефіцієнта значимості.

Результати експертного ранжування наведені у таблиці 4.3.

Для перевірки степені достовірності експертних оцінок визначаються наступні параметри:

1. Сума рангів кожного з параметрів і загальна сума рангів:

$$R_i = \sum_{j=1}^{N} r_{ij} R_{ij} = \frac{Nn(n+1)}{2} = 70, , \quad (4.1)$$

де N – число експертів;

n – кількість параметрів.

Таблиця 4.3 — Результати експертного ранжування

Позначення параметра	*Назва параметра*	*Ранг параметра за оцінкою експерта*							*Сума рангів* R_i	*Відхилення* Δ_i	Δ_i^2
		1	2	3	4	5	6	7			
X1	Швидкодія мови програмування	2	2	1	3	2	1	2	13	-4.5	20.25
X2	Об’єм оперативної пам’яті	1	1	2	1	1	2	2	10	-7.5	56.25
X3	Час навчання моделей	3	4	3	3	4	4	3	24	6.5	42.25
X4	Обсяг програмного коду	4	3	4	3	3	3	3	23	5.5	30.25
Разом		10	10	10	10	10	10	10	70		149

2. Середня сума рангів:

$$T = \frac{1}{n} R_{ij} = 17{,}5 \ . \tag{4.2}$$

3. Відхилення суми рангів кожного параметра від середньої суми рангів:

$$\Delta_i = R_i - T. \tag{4.3}$$

Сума відхилень по всім параметрам повинна дорівнювати 0.

4. Загальна сума квадратів відхилення:

$$S = \sum_{i=1}^{N} \Delta_i^2 = 137. \tag{4.4}$$

Коефіцієнт узгодженості обчислюється за формулою:

$$W = \frac{12S}{N^2(n^3-n)} = \frac{12 \cdot 197}{7^2(4^3-4)} = 0{,}754 > W_k = 0{,}56. \quad (4.5)$$

Ранжування можна вважати достовірним, тому що знайдений коефіцієнт узгодженості перевищує нормативний, котрий дорівнює 0,67.

Скориставшись результатами ранжування, буде проведено попарне порівняння всіх параметрів та занесення результатів у таблиці 4.4.

Таблиця 4.4 — Попарне порівняння параметрів

Параметри	*Експерти*							*Кінцева оцінка*	*Числове значення*
	1	2	3	4	5	6	7		
X_1 і X_2	>	>	<	>	>	<	=	>	1.5
X_1 і X_3	<	<	<	=	<	<	<	<	0.5
X_1 і X_4	<	<	<	=	<	<	<	<	0.5
X_2 і X_3	<	<	<	<	<	<	<	<	0.5
X_2 і X_4	<	>	<	<	<	<	<	<	0.5
X_3 і X_4	<	<	<	=	>	>	=	>	1.5

Числове значення, що визначає ступінь переваги і- го параметра над jтим визначається по формулі:

$$a_{ij} = \begin{cases} 1.5 \text{ при } X_i > X_j \\ 1.0 \text{ при } X_i = X_j \\ 0.5 \text{ при } X_i < X_j \end{cases}. \quad (4.6)$$

З отриманих числових оцінок переваги складемо матрицю А=$\| a_{ij} \|$.

Для кожного параметра зробимо розрахунок вагомості за наступними формулами:

$$WK_{\text{Bi}} = \frac{b_i}{\sum_{i=1}^{n} b_i}, \quad (4.7)$$

$$b_i = \sum_{i=1}^{N} a_{ij}. \quad (4.8)$$

Відносні оцінки розраховуються декілька разів доти, поки наступні значення не будуть незначно відрізнятися від попередніх (менше 2%). На другому і наступних кроках відносні оцінки розраховуються за наступними формулами:

$$K_{\text{Bi}} = \frac{b_i'}{\sum_{i=1}^{n} b_i'}, \quad (4.9)$$

$$b_i' = \sum_{i=1}^{N} a_{ij} b_j. \quad (4.10)$$

Зобразимо результати розрахунків вагомості параметрів на табл. 4.5:

Таблиця 4.5 — Результати розрахунків вагомості параметрів

Параметри X_i	*Параметри* X_j				*Перша ітер.*		*Друга ітер.*	
	X_1	X_2	X_3	X_4	b_i	K_{Bi}	$b^1{}_i$	$K^1{}_{Bi}$
X_1	1	1,5	0,5	0,5	3,5	0,22	12,25	0,21
X_2	0,5	1	0,5	0,5	2,5	0,16	9,25	0,15
X_3	1,5	1,5	1	1,5	5,5	0,34	21,25	0,36
X_4	1,5	1,5	0,5	1	4,5	0,28	16,25	0,28
Разом					16	1	59	1

Після другої ітерації різниця значень коефіцієнтів вагомості не перевищує 2%, тому подальші ітерації не потребуються.

4.5 Аналіз рівня варіантів реалізації функцій

Визначено рівень якості кожного варіанту виконання основних функцій окремо.

Абсолютні значення параметрів X_2 (необхідний обсяг пам'яті), X_3 (час навчання моделі) та X_4 (точність моделі) відповідають технічним вимогам умов функціонування даного ПП.

Абсолютне значення параметра X_1 (потенційний обсяг програмного коду) обрано не найгіршим.

Коефіцієнт технічного рівня для кожного варіанта реалізації ПП розраховується наступним чином (таблиця 4.6):

$$K_K(j) = \sum_{i=1}^{n} K_{вi,j} B_{i,j}, \quad (4.11)$$

де n – кількість параметрів;

$K_{вi}$ – коефіцієнт вагомості і–го параметра;

B_i – оцінка і–го параметра в балах.

Таблиця 4.6 – Розрахунок показників рівня якості варіантів реалізації основних функцій ПП

Основні функції	*Варіанти реалізації функції*	*Параметри*	*Абсолютне значення параметра*	*Бальна оцінка параметра*	*Коефіцієнт вагомості параметра*	*Коефіцієнт рівня якості*
F1	А	X1	1500	7	0,16	1,05
F2	А	X2	2,8	8	0,28	2,12
F3	А	X3	1,7	3	0,2	0,5
	Б	X4	93,5	8	0,36	2,88

За даними з таблиці 4.6 за формулою:

$$K_K = K_{\text{ТУ}}[F_{1k}] + K_{\text{ТУ}}[F_{2k}] + \ldots + K_{\text{ТУ}}[F_{zk}], \quad (4.12)$$

визначено рівень якості кожного з варіантів:

$$\text{К}_{\text{К1}} = 1{,}05 + 2{,}12 + 0{,}5 = 3{,}67,$$
$$\text{К}_{\text{К2}} = 1{,}05 + 2{,}12 + 2{,}88 = 6{,}05.$$

Помітно з розрахунків, кращим виявився варіант 2, для нього коефіцієнт технічного рівня має більше значення.

4.6 Економічний аналіз варіантів розробки ПП

Для визначення вартості розробки програмного продукту спочатку необхідно розрахувати його трудомісткість. Кожен із розглянутих варіантів реалізації включає два окремі етапи: розробку проєктної частини програмного продукту та створення програмної оболонки.

Перше завдання за рівнем новизни належить до групи А, друге – до групи Б. За складністю алгоритмів завдання 1 віднесено до групи 1, тоді як завдання 2 – до групи 3.

Для реалізації першого завдання використовується довідкова інформація, а для другого – інформація у вигляді вхідних даних.

Проведемо розрахунок норм часу на розробку та програмування для кожного з завдань.

Загальна трудомісткість обчислюється як:

$$\text{Т}_{\text{О}} = \text{Т}_{\text{Р}} \cdot \text{К}_{\text{П}} \cdot \text{К}_{\text{СК}} \cdot \text{К}_{\text{М}} \cdot \text{К}_{\text{СТ}} \cdot \text{К}_{\text{СТ.М}}, \quad (4.13)$$

де T_P – трудомісткість розробки ПП, $K_П$ – поправочний коефіцієнт, $K_{СК}$ – коефіцієнт на складність вхідної інформації, $K_М$ – коефіцієнт рівня мови програмування, $K_{СТ}$ – коефіцієнт використання стандартних модулів і прикладних програм, $K_{СТ.М}$ – коефіцієнт стандартного математичного забезпечення.

Для першого завдання, виходячи із норм часу для завдань розрахункового характеру ступеню новизни А та групи складності алгоритму 1, трудомісткість дорівнює: $T_P = 30$ людино-днів. Поправочний коефіцієнт, який враховує вид нормативно-довідкової інформації для першого завдання: $K_П = 1.75$. Поправочний коефіцієнт, який враховує складність контролю вхідної та вихідної інформації для всіх семи завдань рівний 1: $K_{СК} = 1$. Оскільки при розробці першого завдання використовуються стандартні модулі, врахуємо це за допомогою коефіцієнта $K_{СТ} = 0.9$. Тоді загальна трудомісткість програмування першого завдання дорівнює:

$$T_1 = 30 \cdot 1.75 \cdot 1 \cdot 0.9 = 47{,}25 \text{ людино} - \text{днів}.$$

Проведемо аналогічні розрахунки для подальших завдань.

Для другого завдання (використовується алгоритм третьої групи складності, степінь новизни Б), тобто $T_р = 27$ людино-днів, $K_П = 0.9$, $K_{СК} = 1$, $K_{СТ} = 0.8$:

$$T_2 = 27 \cdot 0.9 \cdot 0.8 = 19{,}44 \text{ людино} - \text{днів}.$$

Складаємо трудомісткість відповідних завдань для кожного з обраних варіантів реалізації програми, щоб отримати їхню трудомісткість:

$$T_I = (47{,}25 + 19{,}44 + 3{,}96 + 19{,}44) \cdot 8 = 720{,}72 \text{ людино-годин},$$

$$T_{II} = (47{,}25 + 19{,}44 + 6{,}24 + 19{,}44) \cdot 8 = 738{,}96 \text{ людино-годин}.$$

Найбільш високу трудомісткість має варіант ІІ.

У розробці беруть участь один програміст з окладом 21000 грн. та один аналітик в області даних з окладом 16000 грн. Визначено середню зарплату за годину за формулою:

$$С_Ч = \frac{М}{T_m \cdot t} грн., \quad (4.14)$$

де М – місячний оклад працівників, T_m – кількість робочих днів на місяць, t – кількість робочих годин в день.

$$С_ч = \frac{21000+16000}{2 \cdot 18 \cdot 8} = 128{,}47 \text{ грн.}$$

Розрахунок заробітньої плати відбувається за формулою:

$$С_{ЗП} = С_ч \cdot T_i \cdot К_Д, \quad (4.15)$$

де $С_ч$ – величина погодинної оплати праці програміста, T_i – трудомісткість відповідного завдання; $К_Д$ – норматив, який враховує додаткову заробітну плату.

Зарплата розробників за варіантами становить:

$$\text{I. } С_{ЗП} = 128{,}47 \cdot 720{,}72 \cdot 1{,}2 = 111109{,}09 \text{ грн},$$
$$\text{II. } С_{ЗП} = 128{,}47 \cdot 738{,}96 \cdot 1{,}2 = 113921{,}03 \text{ грн}.$$

Відрахування на єдиний соціальний внесок становить 22%:

$$\text{I. } С_{ВІД} = С_{ЗП} \cdot 0{,}22 = 111109{,}09 \cdot 0{,}22 = 24443{,}99 \text{ грн},$$
$$\text{II. } С_{ВІД} = С_{ЗП} \cdot 0.22 = 113921{,}03 \cdot 0.22 = 25062{,}62 \text{ грн}.$$

Тепер визначимо витрати на оплату однієї машино-години ($С_М$).

Так як одна ЕОМ обслуговує одного програміста з окладом 21000 грн., з коефіцієнтом зайнятості 0,2 то для однієї машини отримаємо:

$$С_Г = 12 \cdot М \cdot К_З = 12 \cdot 21000 \cdot 0{,}2 = 50400 \text{ грн.}$$

З урахування додаткової заробітньої плати:

$$С_{ЗП} = С_Г \cdot (1 + К_З) = 50400 \cdot (1 + 0.2) = 60480 \text{ грн.}$$

Відрахування на соціальний внесок:

$$С_{ВІД} = С_{ЗП} \cdot 0.22 = 60480 \cdot 0{,}22 = 13305{,}6 \text{ грн.}$$

Амортизаційні відрахування розраховуємо при амортизації 25% та вартості ЕОМ – 24000 грн.

$$С_А = К_{ТМ} \cdot К_А \cdot Ц_{ПР} = 1.4 \cdot 0.25 \cdot 24000 = 8400 \text{ грн.}$$

де $К_{ТМ}$– коефіцієнт, який враховує витрати на транспортування та монтаж приладу у користувача; $К_А$– річна норма амортизації; $Ц_{ПР}$– договірна ціна приладу.

Витрати на ремонт та профілактику розраховуються за формулою:

$$С_Р = К_{ТМ} \cdot Ц_{ПР} \cdot К_Р = 1.4 \cdot 24000 \cdot 0.08 = 2688 \text{ грн,}$$

де $К_р$ – відсоток витрат на поточні ремонти.

Ефективний годинний фонд часу ПК за рік розраховуємо за формулою:

$$Т_{ЕФ} = (Д_К - Д_В - Д_С - Д_Р) \cdot t_З \cdot К_В = (365 - 104 - 12 - 16) \cdot 8 \cdot 0.4 = 745{,}6 \text{ години},$$

де $Д_К$ – календарна кількість днів у році; $Д_В$, $Д_С$ – відповідно кількість вихідних та святкових днів; $Д_Р$ – кількість днів планових ремонтів устаткування; t –кількість робочих годин в день; $К_В$ – коефіцієнт використання приладу у часі протягом зміни.

Витрати на оплату електроенергії розраховуються за формулою:

$$С_{ЕЛ} = Т_{ЕФ} \cdot N_C \cdot К_З \cdot Ц_{ЕН} = 745{,}6 \cdot 0{,}3 \cdot 0{,}5 \cdot 9{,}43 = 1054{,}65 \text{ грн},$$

де N_C – середньо-споживча потужність приладу; $К_З$ – коефіцієнт зайнятості приладу; $Ц_{ЕН}$ – тариф за 1 КВт-годин електроенергії.

Накладні витрати розраховуються за формулою:

$$С_Н = Ц_{ПР} \cdot 0{,}67 = 24000 \cdot 0{,}67 = 16080 \text{ грн}.$$

Тоді річні експлуатаційні витрати розраховуватимуться за формулою:

$$С_{ЕКС} = С_{ЗП} \cdot С_{ВІД} \cdot С_А \cdot С_Р \cdot С_{ЕЛ} \cdot С_Н, \quad (4.16)$$

$$С_{ЕКС} = 63360 + 13939{,}2 + 8400 + 2688 + 1054{,}65 + 16080 = 105521{,}85 \text{ грн}.$$

Собівартість однієї машино-години ЕОМ дорівнюватиме:

$$С_{М-Г} = \frac{С_{ЕКС}}{Т_{ЕФ}} = 105521{,}85 / 745{,}6 = 141{,}53 \text{ грн/год}.$$

Оскільки в даному випадку всі роботи, які пов‘язані з розробкою програмного продукту ведуться на ЕОМ, витрати на оплату машинного часу, в залежності від обраного варіанта реалізації, складає:

$$С_М = С_{М-Г} \cdot Т, \quad (4.17)$$

$$I.\ С_М = 141{,}53 \cdot 720{,}72 = 102003{,}5 \text{ грн.}$$
$$II.\ С_М = 141{,}53 \cdot 738{,}96 = 104585 \text{ грн.}$$

Накладні витрати складають 67% від заробітної плати:

$$С_Н = С_{ЗП} \cdot 0{,}67, \quad (4.18)$$

$$I.\ С_Н = 120120{,}96 \cdot 0{,}67 = 80481{,}04 \text{ грн,}$$
$$II.\ С_Н = 123160{,}985 \cdot 0{,}67 = 82517{,}86 \text{ грн.}$$

Отже, вартість розробки ПП за варіантами становить:

$$С_{ПП} = С_{ЗП} \cdot С_{ВІД} \cdot С_М \cdot С_Н, \quad (4.19)$$

$$I.С_{ПП} = 120120{,}96 + 26426{,}61 + 102003{,}5 + 80481{,}04 = 329032{,}11 \text{ грн.,}$$
$$II.С_{ПП} = 123160{,}985 + 27095{,}42 + 104585 + 82517{,}86 = 337359{,}27 \text{ грн.}$$

4.7 Вибір кращого варіанту ПП техніко-економічного рівня

Коефіцієнт техніко-економічного рівня розраховано за формулою:

$$К_{ТЕР\,j} = К_{К\,j} \cdot К_{Ф\,j}, \quad (4.20)$$

$$К_{ТЕР1} = 3{,}96 \,/\, 329032{,}11 \;=\; 1{,}20353 \cdot 10^{-5},$$

$$К_{ТЕР2} = 6{,}24 - 337359{,}27 \;=\; 1{,}84966 \cdot 10^{-5}.$$

Найбільш ефективним є другий варіант реалізації програмного продукту з коефіцієнтом техніко-економічного рівня $К_{ТЕР2} = 1{,}84966 \cdot 10^{-5}$.

Отже, за результатами проведеного функціонально-вартісного аналізу програмного комплексу, що знаходиться на етапі розробки, здійснено порівняння двох попередньо відібраних варіантів реалізації. На основі отриманих даних встановлено, що другий варіант є найбільш доцільним і оптимальним з погляду технічних характеристик, економічної доцільності та зручності впровадження. У цьому варіанті передбачається використання мови програмування Python, яка відзначається високою гнучкістю, широким спектром бібліотек для роботи з даними та машинним навчанням. Реалізація основної логіки системи ґрунтується на використанні готових бібліотек, що дозволяє значно скоротити час розробки та зменшити обсяг програмного коду. Крім того, середовище Google Colab, яке було обрано для розробки, забезпечує віддалений доступ до обчислювальних ресурсів, зручний інтерфейс та широкі можливості для тестування, візуалізації результатів і спільної роботи. Такий підхід гарантує високу ефективність створення програмного продукту, його гнучкість, масштабованість і доступність для кінцевого користувача, що є ключовими критеріями у процесі розробки інноваційного програмного забезпечення.

Висновки до розділу 4

У четвертому розділі було проведено ґрунтовне функціонально-вартісне дослідження програмного продукту, що призначений для оцінки

впливу машинного перекладу на якість анотування текстів у задачах автоматичного реферування. В межах цього розділу послідовно виконано постановку задачі проектування, що дозволило чітко окреслити функціональну мету програмної системи, її призначення, сферу застосування та основні вимоги до реалізації.

Подальше обґрунтування функцій програмного продукту дало змогу визначити перелік необхідних операційних можливостей, які мають бути реалізовані для забезпечення коректної роботи системи, таких як завантаження корпусу текстів, генерація рефератів за допомогою моделей машинного навчання, переклад текстів та порівняльна оцінка результатів за допомогою лексико-семантичних метрик.

Розглянуто систему параметрів, що описують технічні характеристики програмного забезпечення: швидкодія, використання оперативної пам’яті, тривалість навчання моделей та обсяг програмного коду. Значення цих параметрів були узагальнені в табличній формі із класифікацією на гірші, середні та кращі, що дало змогу здійснити подальшу формалізовану оцінку.

В Аналіз експертного оцінювання передбачено залучення фахівців до визначення ступеня важливості кожного параметра за методом попарного порівняння. Результати опрацювання думок експертів дозволили отримати узагальнену систему пріоритетів і ранжувати параметри відповідно до їх впливу на якість програмного продукту.

У розділі також виконано оцінювання різних варіантів реалізації функціоналу. Проведено порівняльний аналіз на основі зібраних параметрів і вагових коефіцієнтів, що дало змогу обчислити рівень техніко-економічної ефективності кожного варіанта. Зокрема, розглянуто рішення, реалізовані на різних мовах програмування, з використанням як локального, так і хмарного середовища, з урахуванням рівня автоматизації, складності реалізації та витрат часу на розробку.

Особливу увагу приділено економічному аналізу альтернативних рішень, де враховано витрати на розробку, впровадження та експлуатацію.

Порівняння витрат у контексті реалізації функціональних можливостей дозволило зробити обґрунтований висновок щодо доцільності вибору конкретного технологічного стеку.

Отже, результати функціонально-вартісного аналізу дозволили обґрунтовано та системно підійти до вибору рішення, що забезпечує оптимальне співвідношення витрат, функціональності та технічного рівня реалізації програмного продукту.

ВИСНОВКИ

У межах цієї дипломної роботи здійснено комплексне дослідження впливу машинного перекладу на якість анотування текстів, зокрема в контексті автоматичної сумаризації. Проблематика дослідження охоплює як технічні, так і лінгвістичні аспекти застосування сучасних систем машинного перекладу в багатомовних NLP-завданнях.

У першому розділі сформульовано актуальність задачі, проаналізовано розвиток технологій автоматичного перекладу, зокрема нейронних підходів, та виявлено ключові труднощі, які виникають під час поєднання процесів перекладу й узагальнення інформації. Визначено актуальні метрики для оцінки якості анотацій, зокрема ROUGE, BLEU та BERTScore, а також поставлено основну дослідницьку мету — виявити, яким чином машинний переклад змінює результати автоматичного анотування.

Другий розділ присвячено побудові експериментального пайплайну, що охоплює процеси генерації summary, перекладу (EN → UK, UK → EN), а також оцінки отриманих результатів за допомогою згаданих метрик. Обґрунтовано вибір корпусу CNN/DailyMail, моделі сумаризації BART, перекладачів Helsinki-NLP та інструментів оцінки якості. Це дало змогу побудувати надійну експериментальну базу для подальшого аналізу.

У третьому розділі реалізовано практичну частину дослідження — здійснено серію експериментів, у межах яких порівнювалися результати сумаризації до та після перекладу текстів і/або резюме. Зібрано статистику для кількох сценаріїв, зокрема: переклад лише summary, переклад повного тексту, повний цикл EN → UK → EN. Проведено кількісну й якісну інтерпретацію результатів. Виявлено, що навіть незначні трансформації внаслідок перекладу можуть призводити до зниження значень оцінних метрик, а також до лексичних та структурних змін у результатах. Окрему

увагу приділено аналізу втрати смислових одиниць і трансформації синтаксису.

У четвертому розділі здійснено функціонально-вартісний аналіз програмного рішення, що реалізує експериментальний пайплайн. Визначено ключові функції, параметри якості реалізації, а також виконано експертне оцінювання для встановлення пріоритетів серед параметрів (наприклад, швидкодія, обсяг пам’яті, час навчання, розмір коду). Порівняно можливі варіанти реалізації та проведено економічний аналіз. Вибір кращого варіанту ґрунтується на поєднанні технічних і економічних критеріїв — оптимальним виявився підхід з використанням мови Python, хмарного середовища Google Colab та готових бібліотек Hugging Face.

Проведене дослідження підтверджує, що використання машинного перекладу в ланцюгу обробки текстів значною мірою впливає на якість автоматичних узагальнень. Це має важливе значення для задач багатомовного аналізу, де переклад не лише доповнює, а й змінює інформаційний зміст тексту. Отримані результати можуть бути використані під час розробки багатомовних систем анотування, машинного перекладу або сумаризації, де потрібно враховувати наслідки проміжних трансформацій тексту.

Загалом робота демонструє важливість комплексного підходу до дослідження взаємодії мовних технологій, а також доводить доцільність поєднання експериментального аналізу з техніко-економічною оцінкою при створенні сучасних програмних продуктів у сфері NLP.

ПЕРЕЛІК ПОСИЛАНЬ

1. Jurafsky D., Martin J. H. *Speech and Language Processing*. — 3rd ed. — Draft, 2022.

2. Nallapati R., Zhai F., Zhou B. *Abstractive Text Summarization Using Sequence-to-Sequence RNNs and Beyond* // arXiv:1602.06023 [cs.CL], 2016.

3. Lewis M. et al. *BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension* // arXiv:1910.13461 [cs.CL], 2019.

4. See A., Liu P. J., Manning C. D. *Get To The Point: Summarization with Pointer-Generator Networks* // arXiv:1704.04368 [cs.CL], 2017.

5. Zhang J. et al. *PEGASUS: Pre-training with Extracted Gap-sentences for Abstractive Summarization* // arXiv:1912.08777 [cs.CL], 2019.

6. Hutchins W. J., Somers H. L. *An Introduction to Machine Translation*. — Academic Press, 1992.

7. Koehn P. et al. *Moses: Open Source Toolkit for Statistical Machine Translation* // Proceedings of the ACL, 2007.

8. Bahdanau D., Cho K., Bengio Y. *Neural Machine Translation by Jointly Learning to Align and Translate* // arXiv:1409.0473 [cs.CL], 2014.

9. Vaswani A. et al. *Attention is All You Need* // Advances in Neural Information Processing Systems, 2017.

10. Junczys-Dowmunt M. et al. *Marian: Fast Neural Machine Translation in C++* // Proceedings of ACL, 2018.

11. Toral A., Way A. *What Level of Quality can Neural Machine Translation Attain on Literary Text?* // Translation Quality Assessment. – Springer, 2018.

12. Papineni K. et al. *BLEU: a method for automatic evaluation of machine translation* // Proceedings of ACL, 2002.

13. Freitag M., Al-Onaizan Y., Sankaran B. *Bleu might be guilty but references are not innocent* // arXiv preprint arXiv:2004.06063, 2020.

14. Lin C. *ROUGE: A Package for Automatic Evaluation of Summaries* // Proceedings of the Workshop on Text Summarization Branches Out, 2004.

15. Papineni K. et al. *BLEU: a Method for Automatic Evaluation of Machine Translation* // Proceedings of ACL, 2002.

16. Zhang T. et al. *BERTScore: Evaluating Text Generation with BERT* // arXiv preprint arXiv:1904.09675, 2019.

17. Зображення трансформера [Електронний ресурс]. – Режим доступу: https://i.ytimg.com/vi/B3uws4cLcFw/maxresdefault.jpg, вільний.

18. Kus, A. Architecture of Transformer Model [Електронний ресурс] // ResearchGate. – Режим доступу: https://www.researchgate.net/profile/Anil-Kus/publication/383282857/figure/fig1/AS:11431281272973346@1724311618995/Architecture-of-Transformer-Model.png, вільний.

ДОДАТОК А ЛІСТИНГ ПРОГРАМИ

```
from datasets import load_dataset
from transformers import pipeline
import re
import evaluate
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
from rouge_score import rouge_scorer
from bert_score import score
import difflib
from collections import Counter
from transformers import BartTokenizer, BartForConditionalGeneration
import spacy
from collections import Counter

# Завантаження датасету
dataset = load_dataset("cnn_dailymail", "3.0.0", split="test")

# Вибір перших 5000 прикладів
examples = dataset.select(range(5000))

# Виведення прикладу для перевірки
for i, example in enumerate(examples):
    print(f"Приклад {i+1}")
    print("Текст:", example['article'], "...")
    print("Реф. анотація:", example['highlights'])
    print("-" * 80)

# Сумаризація
summarizer = pipeline("summarization", model="facebook/bart-large-cnn")

# Переклад EN → UK
translator_en_uk = pipeline("translation", model="Helsinki-NLP/opus-mt-en-
uk")

# Переклад UK → EN
```

```
translator_uk_en = pipeline("translation", model="Helsinki-NLP/opus-mt-uk-en")

# Тестовий текст
original_text = """The Palestinian Authority officially joins the International Criminal Court.
It becomes the 123rd member of the Hague-based tribunal and is now subject to its founding treaty, the Rome Statute.
The move gives the court jurisdiction over alleged crimes committed in Palestinian territories, opening the door to possible war crimes investigations against both Israelis and Palestinians.
Israel and the United States opposed the Palestinians' efforts to join the body.
The Palestinians have also formally accepted the court's jurisdiction retroactively to June 13, 2014, a move that could target Israel's actions during last summer's war in Gaza."""

# Функція для розбиття на речення
def simple_sentence_split(text):
    sentences = re.split(r'(?<=[.!?]) +', text.strip())
    return [s for s in sentences if s]

# A. Сумаризація оригіналу
original_summary = summarizer(original_text, max_length=130, min_length=30, do_sample=False)[0]['summary_text']
print("A. Original Summary:\n", original_summary)
print("="*100)

# B. Двосторонній переклад (EN => UK -> EN)
sentences = simple_sentence_split(original_text)
translated_uk_sentences = [translator_en_uk(sentence, max_length=400)[0]['translation_text'] for sentence in sentences]
translated_ukr = " ".join(translated_uk_sentences)

sentences_ukr = simple_sentence_split(translated_ukr)
translated_back_sentences = [translator_uk_en(sentence, max_length=400)[0]['translation_text'] for sentence in sentences_ukr]
translated_back = " ".join(translated_back_sentences)

summary_back = summarizer(translated_back, max_length=130, min_length=30, do_sample=False)[0]['summary_text']
```

```
print("B. Summary after EN → UK → EN:\n", summary_back)
print("="*100)

# C. Односторонній переклад (EN -> UK)
summary_translated_uk = translator_en_uk(original_summary,
max_length=400)[0]['translation_text']
print("C. Original Summary translated to Ukrainian:\n",
summary_translated_uk)
print("="*100)

# D. Порівняння summary з повним текстом (перекладений повний текст ->
summary)
summary_of_translated_full = summarizer(translated_ukr, max_length=130,
min_length=30, do_sample=False)[0]['summary_text']
print("D. Summary of Translated Full Text (EN → UK):\n",
summary_of_translated_full)
print("="*100)

# 1. Завантаження перших 20 прикладів
dataset = load_dataset("cnn_dailymail", "3.0.0", split="test[:20]")

def summarize(text, max_length=400, min_length=30):
    return summarizer(text, max_length=max_length, min_length=min_length,
do_sample=False)[0]['summary_text']

def translate_sentences(text, translator):
    # sentence splitter на основі крапки
    sentences = re.split(r'(?<=[.!?])\s+', text.strip())
    translated = []
    for s in sentences:
        if s.strip():
            out = translator(s, max_length=400)
            translated.append(out[0]['translation_text'])
    return " ".join(translated)

# 2. Обробка кількох прикладів
for i, example in enumerate(dataset):
    print(f"\n{'='*100}")
    print(f"📄 Example #{i+1}")
```

```
    original_text = example['article']
    original_summary = example['highlights']

    # A. Оригінальна сумаризація (використаємо існуючу summary)
    print("\nA. □ Original Summary:")
    print(original_summary)

    # B. Двосторонній переклад (EN → UK → EN)
    translated_ukr = translate_sentences(original_text, translator_en_uk)
    translated_back = translate_sentences(translated_ukr,
translator_uk_en)
    try:
        summary_back = summarize(translated_back)
    except:
        summary_back = "[□ Summarization failed]"

    print("\nB. 🔄 Summary after EN → UK → EN:")
    print(summary_back)

    # C. Переклад оригінальної summary EN → UK
    translated_summary_uk = translate_sentences(original_summary,
translator_en_uk)
    print("\nC. 🌍 Original Summary translated to Ukrainian:")
    print(translated_summary_uk)

    # D. Summary перекладеного тексту (EN → UK)
    translated_text_uk = translate_sentences(original_text,
translator_en_uk)
    try:
        summary_of_translated_uk = summarize(translated_text_uk[:800])
    except:
        summary_of_translated_uk = "[□ Summarization failed]"

    print("\nD. 🔍 Summary of Translated Full Text (EN → UK):")
    print(summary_of_translated_uk)

print("\n✅ Завершено обробку всіх прикладів.")

# Завантаження ROUGE метрики
rouge = evaluate.load("rouge")
```

```
# Збір результатів
originals = []
back_translations = []
translated_summaries = []
summaries_of_uk_text = []

for example in dataset:
    original_summary = example['highlights']
    original_text = example['article']

    # Двосторонній переклад → сумаризація
    translated_ukr = translate_sentences(original_text, translator_en_uk)
    translated_back = translate_sentences(translated_ukr,
translator_uk_en)
    try:
        summary_back = summarize(translated_back)
    except:
        summary_back = ""

    # Перекладена summary (EN → UK)
    translated_summary_uk = translate_sentences(original_summary,
translator_en_uk)

    # Summary перекладеного повного тексту (EN → UK)
    translated_text_uk = translate_sentences(original_text,
translator_en_uk)
    try:
        summary_of_translated = summarize(translated_text_uk[:800])
    except:
        summary_of_translated = ""

    originals.append(original_summary)
    back_translations.append(summary_back)
    translated_summaries.append(translated_summary_uk)
    summaries_of_uk_text.append(summary_of_translated)

# Обчислення метрик
results = {
    "EN→UK→EN Summary": rouge.compute(predictions=back_translations,
references=originals),
```

```
    "EN→UK Summary→UK": rouge.compute(predictions=translated_summaries,
references=originals),
    "Summary of (EN→UK) Full Text":
rouge.compute(predictions=summaries_of_uk_text, references=originals),
}

# Вивід результатів
for label, score in results.items():
    print("=" * 100)
    print(f"📊 ROUGE Scores for: {label}")
    for k, v in score.items():
        print(f"{k}: {v:.4f}")

# Ініціалізація токенайзера та моделі для BART
tokenizer = BartTokenizer.from_pretrained('facebook/bart-large-cnn')
model = BartForConditionalGeneration.from_pretrained('facebook/bart-large-
cnn')

# Функція для сумаризації
def summarize(text, max_length=130, min_length=30):
    inputs = tokenizer.encode(text, return_tensors='pt', truncation=True,
max_length=1024)
    summary_ids = model.generate(inputs, max_length=max_length,
min_length=min_length, length_penalty=2.0, num_beams=4,
early_stopping=True)
    summary = tokenizer.decode(summary_ids[0], skip_special_tokens=True)
    return summary

# Функція для обчислення ROUGE-метрик
def compute_rouge(prediction, reference):
    scorer = rouge_scorer.RougeScorer(["rouge1", "rouge2", "rougeL"],
use_stemmer=True)
    scores = scorer.score(reference, prediction)
    return scores

# Завантаження набору даних
dataset = load_dataset('cnn_dailymail', '3.0.0', split='test[:20]')

# Вибір метрик для збереження
rouge_metrics = {
```

```
    "rouge1": [],
    "rouge2": [],
    "rougeL": [],
    "rougeLsum": []
}

# Аналіз усіх прикладів
for i, example in enumerate(dataset):
    original_text = example['article']
    original_summary = example['highlights']

    # A. Оригінальна сумаризація (використаємо існуючу summary)
    original_summary_pred = original_summary

    # B. Двосторонній переклад (EN → UK → EN)
    translated_ukr = translate_sentences(original_text, translator_en_uk)
    translated_back = translate_sentences(translated_ukr,
translator_uk_en)
    summary_back = summarize(translated_back)

    # C. Переклад оригінальної summary EN → UK
    translated_summary_uk = translate_sentences(original_summary,
translator_en_uk)

    # D. Summary перекладеного тексту (EN → UK)
    translated_text_uk = translate_sentences(original_text,
translator_en_uk)
    summary_of_translated_uk = summarize(translated_text_uk[:800])

    # Обчислення ROUGE для кожного варіанту
    rouge_original = compute_rouge(original_summary_pred,
original_summary_pred)  # reference та prediction однакові
    rouge_en_uk_en = compute_rouge(summary_back, original_summary)
    rouge_en_uk = compute_rouge(translated_summary_uk, original_summary)
    rouge_translated = compute_rouge(summary_of_translated_uk,
original_summary)

    # Збереження результатів
    rouge_metrics["rouge1"].append({
        "original": rouge_original["rouge1"].fmeasure,
        "en_uk_en": rouge_en_uk_en["rouge1"].fmeasure,
```

```
        "en_uk": rouge_en_uk["rouge1"].fmeasure,
        "translated": rouge_translated["rouge1"].fmeasure
    })
    rouge_metrics["rouge2"].append({
        "original": rouge_original["rouge2"].fmeasure,
        "en_uk_en": rouge_en_uk_en["rouge2"].fmeasure,
        "en_uk": rouge_en_uk["rouge2"].fmeasure,
        "translated": rouge_translated["rouge2"].fmeasure
    })
    rouge_metrics["rougeL"].append({
        "original": rouge_original["rougeL"].fmeasure,
        "en_uk_en": rouge_en_uk_en["rougeL"].fmeasure,
        "en_uk": rouge_en_uk["rougeL"].fmeasure,
        "translated": rouge_translated["rougeL"].fmeasure
    })

# Обчислення середніх значень
average_rouge1 = np.mean([item["original"] for item in
rouge_metrics["rouge1"]])
average_rouge2 = np.mean([item["original"] for item in
rouge_metrics["rouge2"]])
average_rougeL = np.mean([item["original"] for item in
rouge_metrics["rougeL"]])

# Створення DataFrame для виведення середніх значень
df_rouge = pd.DataFrame({
    "Metric": ["ROUGE-1", "ROUGE-2", "ROUGE-L"],
    "EN → UK → EN": [np.mean([item["en_uk_en"] for item in
rouge_metrics["rouge1"]]),
                     np.mean([item["en_uk_en"] for item in
rouge_metrics["rouge2"]]),
                     np.mean([item["en_uk_en"] for item in
rouge_metrics["rougeL"]])],
    "EN → UK": [np.mean([item["en_uk"] for item in
rouge_metrics["rouge1"]]),
                np.mean([item["en_uk"] for item in
rouge_metrics["rouge2"]]),
                np.mean([item["en_uk"] for item in
rouge_metrics["rougeL"]])],
    "Summary of Translated Full Text": [np.mean([item["translated"] for
item in rouge_metrics["rouge1"]]),
```

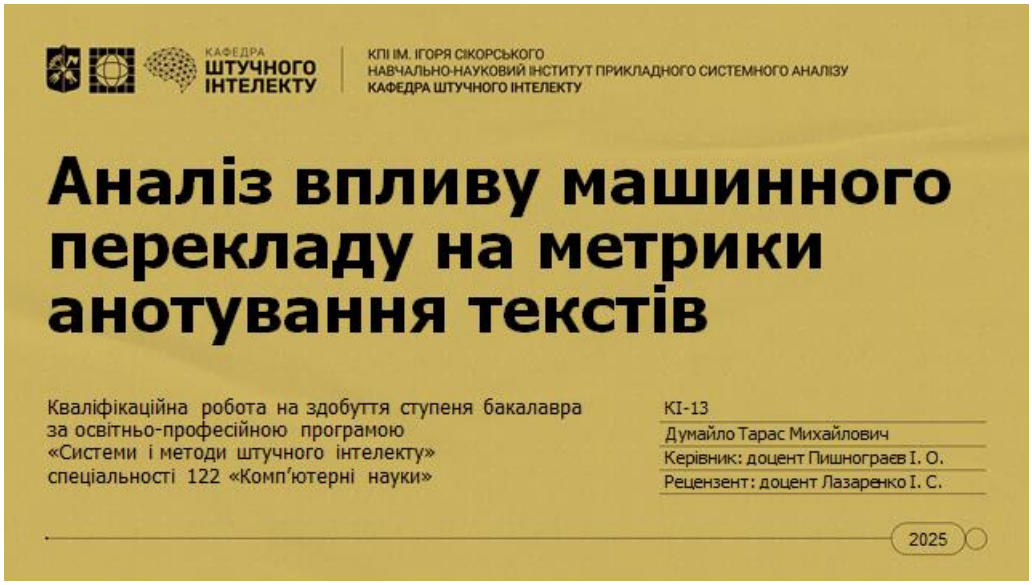
```
                                        np.mean([item["translated"] for
item in rouge_metrics["rouge2"]]),
                                        np.mean([item["translated"] for
item in rouge_metrics["rougeL"]])],
})

print("\n📊 ROUGE Scores Summary:")
print(df_rouge)

# Побудова графіків
plt.figure(figsize=(10, 6))

# Графік для впливу перекладу в обидва боки (EN → UK → EN)
plt.subplot(2, 2, 1)
plt.bar(["ROUGE-1", "ROUGE-2", "ROUGE-L"], df_rouge["EN → UK → EN"],
color='skyblue')
plt.title("Impact of Bidirectional Translation (EN → UK → EN)")

# Графік для впливу одностороннього перекладу (EN → UK)
plt.subplot(2, 2, 2)
plt.bar(["ROUGE-1", "ROUGE-2", "ROUGE-L"], df_rouge["EN → UK"],
color='lightgreen')
plt.title("Impact of Unidirectional Translation (EN → UK)")

# Графік для ROUGE між summary та перекладеним текстом
plt.subplot(2, 2, 3)
plt.bar(["ROUGE-1", "ROUGE-2", "ROUGE-L"], df_rouge["Summary of Translated
Full Text"], color='lightcoral')
plt.title("ROUGE between Summary and Translated Full Text")

plt.tight_layout()
plt.show()

# Ініціалізація BLEU
bleu = evaluate.load("bleu")

# BLEU метрики
bleu_results = {
    "EN→UK→EN Summary": bleu.compute(predictions=back_translations,
references=[[ref] for ref in originals]),
```

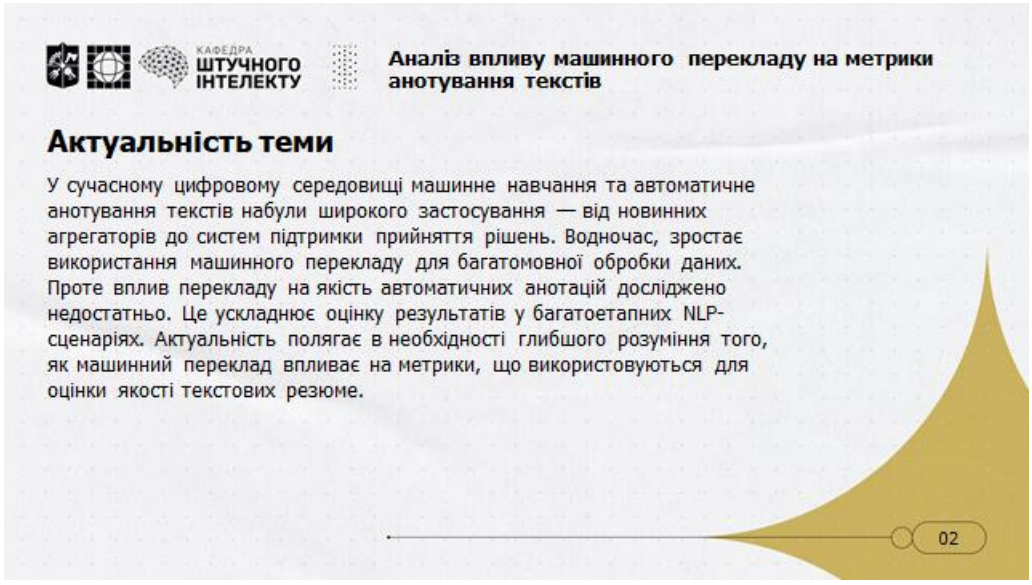
```
    "EN→UK Summary→UK": bleu.compute(predictions=translated_summaries,
references=[[ref] for ref in originals]),
    "Summary of (EN→UK) Full Text":
bleu.compute(predictions=summaries_of_uk_text, references=[[ref] for ref
in originals]),
}

# Вивід BLEU результатів
for label, score in bleu_results.items():
    print("=" * 100)
    print(f"📄 BLEU Score for: {label}")
    print(f"BLEU: {score['bleu']:.4f}")

from bert_score import score

# Функція для обчислення BERTScore
def compute_bertscore(predictions, references):
    P, R, F1 = score(predictions, references, lang="en", verbose=True)
    return P.mean().item(), R.mean().item(), F1.mean().item()

# BERTScore для кожної пари summary
bertscore_results = {
    "EN→UK→EN Summary": compute_bertscore(back_translations, originals),
    "EN→UK Summary→UK": compute_bertscore(translated_summaries,
originals),
    "Summary of (EN→UK) Full Text":
compute_bertscore(summaries_of_uk_text, originals),
}

# Виведення результатів BERTScore
for label, score in bertscore_results.items():
    print("=" * 100)
    print(f"📄 BERTScore for: {label}")
    print(f"Precision: {score[0]:.4f}")
    print(f"Recall: {score[1]:.4f}")
    print(f"F1-Score: {score[2]:.4f}")

# Завантаження англійської моделі
nlp = spacy.load("en_core_web_sm")
```

```
# Функція для підрахунку POS тегів
def count_pos(text):
    doc = nlp(text)
    pos_counts = Counter([token.pos_ for token in doc])
    return dict(pos_counts)

# Для зберігання результатів
pos_results = {
    "original": [],
    "en_uk_en": [],
    "translated_text": []
}

for i, example in enumerate(dataset):
    original_summary = example['highlights']

    # A. Оригінальна summary
    pos_results["original"].append(count_pos(original_summary))

    # B. Summary після EN → UK → EN
    translated_ukr = translate_sentences(example['article'],
translator_en_uk)
    translated_back = translate_sentences(translated_ukr,
translator_uk_en)
    try:
        summary_back = summarize(translated_back)
    except:
        summary_back = ""
    pos_results["en_uk_en"].append(count_pos(summary_back))

    # C. Summary з перекладеного повного тексту
    translated_text_uk = translate_sentences(example['article'],
translator_en_uk)
    try:
        summary_of_translated = summarize(translated_text_uk[:800])
    except:
        summary_of_translated = ""

pos_results["translated_text"].append(count_pos(summary_of_translated))

import pandas as pd
```

```
from collections import defaultdict

def average_pos_counts(pos_dicts):
    total = defaultdict(float)
    for d in pos_dicts:
        for k, v in d.items():
            total[k] += v
    avg = {k: total[k]/len(pos_dicts) for k in total}
    return avg

avg_original = average_pos_counts(pos_results["original"])
avg_back = average_pos_counts(pos_results["en_uk_en"])
avg_translated = average_pos_counts(pos_results["translated_text"])

# Об'єднання у DataFrame
df_pos = pd.DataFrame({
    "Original": avg_original,
    "EN→UK→EN": avg_back,
    "From Translated Text": avg_translated
}).fillna(0)

df_pos = df_pos.astype(int).T  # транспонування
print(df_pos)
```

ДОДАТОК Б ЕЛЕКТРОННА ПРЕЗЕНТАЦІЯ





КАФЕДРА ШТУЧНОГО ІНТЕЛЕКТУ

Аналіз впливу машинного перекладу на метрики анотування текстів

Об'єкт, предмет, мета, проблема, завдання

Об'єкт дослідження — процес автоматичного анотування текстів у задачах текстової сумаризації.

Предмет дослідження — вплив використання машинного перекладу на результати оцінювання анотацій за допомогою метрик.

Мета — дослідити, як змінюються значення метрик якості анотацій при використанні машинного перекладу на різних етапах.

Проблема — спотворення якості автоматично згенерованих анотацій при залученні перекладацьких моделей, що може призводити до хибних висновків про продуктивність систем.

Завдання:
- Побудувати експериментальний пайплайн з перекладом і анотуванням.
- Провести обчислення метрик (ROUGE, BLEU, BERTScore).
- Проаналізувати зміни при різних сценаріях перекладу.

КАФЕДРА ШТУЧНОГО ІНТЕЛЕКТУ

Аналіз впливу машинного перекладу на метрики анотування текстів

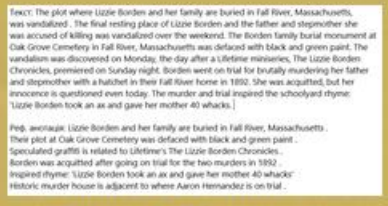
Обрані технології та бібліотеки

Для реалізації експерименту використано сучасні інструменти:
- **Мова програмування**: Python
- **Бібліотеки машинного навчання**:
  transformers — для генерації резюме (модель BART)
  datasets, torch, evaluate, bert_score, nltk, sacrebleu — для оцінювання якості
- **Машинний переклад**: Helsinki-NLP (моделі EN↔UK)
- **Середовище виконання**: Google Colab

КАФЕДРА ШТУЧНОГО ІНТЕЛЕКТУ

Огляд датасету

Для дослідження використано **CNN/DailyMail Dataset** — популярний корпус англомовних новинних текстів та резюме.
- Кожен приклад містить повний текст новини та її коротке анотаційне зведення.
- Обрано 20 прикладів вручну для глибшого аналізу.
- Перевага датасету — наявність якісно підготовлених людських резюме.


Текст: The plot where Lizzie Borden and her family are buried in Fall River, Massachusetts, was vandalized . The final resting place of Lizzie Borden and the father and stepmother she was accused of killing was vandalized over the weekend. The Borden family burial monument at Oak Grove Cemetery in Fall River, Massachusetts was defaced with black and green paint. The vandalism was discovered on Monday, the day after a Lifetime miniseries, The Lizzie Borden Chronicles, premiered on Sunday night. Borden went on trial for brutally murdering her father and stepmother with a hatchet in their Fall River home in 1892. She was acquitted, but her innocence is questioned even today. The murder and trial inspired the schoolyard rhyme: 'Lizzie Borden took an ax and gave her mother 40 whacks.'

Реф. анотація: Lizzie Borden and her family are buried in Fall River, Massachusetts .
Their plot at Oak Grove Cemetery was defaced with black and green paint .
Speculated graffiti is related to Lifetime's The Lizzie Borden Chronicles .
Borden was acquitted after going on trial for the two murders in 1892 .
Inspired rhyme: 'Lizzie Borden took an ax and gave her mother 40 whacks'
Historic murder house is adjacent to where Aaron Hernandez is on trial .

Рисунок 1 – Приклад тексту

КАФЕДРА ШТУЧНОГО ІНТЕЛЕКТУ

Аналіз впливу машинного перекладу на метрики анотування текстів

Попередня обробка даних

Пайплайн обробки включає:
- Попереднє очищення та токенізацію тексту.
- Генерацію резюме за допомогою моделі BART.
- Переклад повного тексту або резюме у напрямках EN→UK, UK→EN.
- Створення трьох варіантів:
  - EN → UK → EN (повне зворотне перетворення)
  - EN → UK (переклад тільки резюме)
  - Summary з перекладеного повного тексту

Рисунок 2 – Обробка даних

06

КАФЕДРА ШТУЧНОГО ІНТЕЛЕКТУ

Аналіз впливу машинного перекладу на метрики анотування текстів

Архітектура моделі

Архітектура експерименту:
- Модель сумаризації: BART (facebook/bart-large-cnn) — генеративна трансформерна архітектура.
- Переклад: Helsinki-NLP, моделі MarianMT.
- Оцінювання:
  - ROUGE (для збігу підрядків)
  - BLEU (n-грам збіги)
  - BERTScore (контекстуальна подібність)

07

КАФЕДРА ШТУЧНОГО ІНТЕЛЕКТУ

Аналіз впливу машинного перекладу на метрики анотування текстів

Результати роботи

У результаті проведеного аналізу:
- Найвищу якість зберігає варіант без перекладу.
- Переклад повного тексту з наступною сумаризацією дає кращі результати, ніж переклад резюме.
- Зворотній переклад EN→UK→EN спотворює метрики найбільше.
- ROUGE та BLEU демонструють чутливість до формальних змін.
- BERTScore краще відображає семантичну подібність після перекладу.

Рисунок 3 – ROUGE

Рисунок 4 – BLEU

Рисунок 5 – BERTScore

08

КАФЕДРА ШТУЧНОГО ІНТЕЛЕКТУ

Аналіз впливу машинного перекладу на метрики анотування текстів

Результати роботи

У результаті проведеного аналізу:

- Найвищу якість зберігає варіант без перекладу.
- Переклад повного тексту з наступною сумаризацією дає кращі результати, ніж переклад резюме.
- Зворотній переклад EN→UK→EN спотворює метрики найбільше.
- ROUGE та BLEU демонструють чутливість до формальних змін.
- BERTScore краще відображає семантичну подібність після перекладу.

Рисунок 3 – ROUGE

Рисунок 4 – BLEU

Рисунок 5 – BERTScore

КАФЕДРА ШТУЧНОГО ІНТЕЛЕКТУ

Аналіз впливу машинного перекладу на метрики анотування текстів

Висновки

Проведене дослідження показало, що машинний переклад може значно впливати на результати автоматичного анотування та оцінку якості. Використання класичних метрик без урахування перекладацьких впливів може спричинити викривлення підсумків. Пайплайн, реалізований у цій роботі, дає змогу гнучко тестувати сценарії обробки текстів і може бути використаний для подальших досліджень у сфері багатомовного NLP. Отримані висновки корисні для практичного застосування в аналітичних системах, де якість коротких текстових зведень критично важлива.