Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет комп'ютерних наук
(повна назва)

Кафедра програмної інженерії
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

Дослідження методів обробки природної мови, що використовуються для перекладу текстових документів
(тема)

Виконав:
здобувач 2-го року навчання
групи ІПЗм-23-3

Артем АНДРІЙЧЕНКО
(Власне ім'я, ПРІЗВИЩЕ)

Спеціальність 121 – Інженерія програмного забезпечення
(код і повна назва спеціальності)
Тип програми освітньо-наукова

Керівник доц. Олександр ВЕЧУР
(посада, Власне ім'я, ПРІЗВИЩЕ)

Допускається до захисту
Зав. кафедри _____ Кирило СМЕЛЯКОВ
(підпис) (Власне ім'я, ПРІЗВИЩЕ)

2025 р.

Харківський національний університет радіоелектроніки

Факультет комп'ютерних наук
Кафедра програмної інженерії
Рівень вищої освіти другий (магістерський)
Спеціальність 121 – Інженерія програмного забезпечення
Тип програми освітньо-наукова програма
Освітня програма Інженерія програмного забезпечення
(шифр і назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«___» _____ 2025 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Андрійченку Артему Віталійовичу
(прізвище, ім'я, по батькові)

1. Тема роботи «Дослідження методів обробки природної мови, що використовуються для перекладу текстових документів»
Затверджена наказом по університету від 15.04.2025р. №290Ст
2. Термін подання студентом роботи до екзаменаційної комісії 20.06.2025
3. Вихідні дані до роботи Опис методів обробки природної мови, що використовуються для автоматизованого перекладу текстових документів, порівняльний аналіз наукових джерел, експериментальне тестування моделей машинного перекладу, оцінка ефективності методів на основі корпусів текстів, включаючи FLORES 200 та спеціально підготовлені корпуси на основі офіційних документів
4. Перелік питань, що потрібно опрацювати в роботі
аналіз та порівняння існуючих NLP моделей, вибір відповідних інструментів та бібліотек, написання програмних рішень, проведення експериментів та аналіз отриманих результатів

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1	Отримання завдання	16.04.2025	*виконано*
2	Аналіз предметної галузі і постановка задачі	16.04. – 22.04.25	*виконано*
3	Аналіз та вибір моделей для дослідження	22.04 – 25.04.25	*виконано*
4	Планування експериментів	25.04 – 28.04.25	*виконано*
5	Програмна реалізація	28.04 – 12.05.25	*виконано*
6	Експериментальні дослідження	12.05 – 16.05.25	*виконано*
7	Аналіз результатів експериментальних досліджень	16.05 – 18.05.25	*виконано*
8	Підготовка до апробації результатів дослідження. Публікація матеріалів	18.05 – 20.05.25	*виконано*
9	Підготовка пояснювальної записки	10.05 – 28.05.25	*виконано*
10	Підготовка презентації та доповіді	28.05 – 31.05.25	*виконано*
11	Перевірка на плагіат	31.05 – 04.06.25	*виконано*
12	Нормоконтроль	04.06 – 09.06.25	*виконано*
13	Рецензування	09.06 – 13.06.25	*виконано*
14	Попередній захист	13.06.2025	*виконано*
15	Занесення диплома в електронний архів	16.06.2025	*виконано*
16	Допуск до захисту у зав. кафедри	20.06.2025	*виконано*

Дата видачі завдання 16 квітня 2025р.

Студент (ка) _____ Артем АНДРІЙЧЕНКО
(підпис)

Керівник роботи _____ доц. Олександр ВЕЧУР
(підпис) (посада, Власне ім’я, ПРІЗВИЩЕ)

РЕФЕРАТ / ABSTRACT

Пояснювальна записка містить: 77 с., 18 рис., 12 табл., 23 джерел.

АВТОМАТИЧНИЙ ПЕРЕКЛАД, НЕЙРОННІ МЕРЕЖІ, ОБРОБКА ПРИРОДНОЇ МОВИ, ТРАНСФОРМЕР, NLP

Об’єкт дослідження – процес автоматизованого перекладу текстових документів із використанням методів обробки природної мови.

Мета роботи – розробка методики підвищення якості автоматизованого перекладу документів шляхом аналізу, експериментального тестування та вдосконалення нейронних моделей із фокусом на низькоресурсні мови.

Методи розробки базуються на таких технологіях, як Python, FastAPI, React, PostgreSQL, Hugging Face Transformers, PyTorch.

У результаті роботи було проведено експериментальне тестування моделей машинного перекладу, включаючи facebook/m2m100, Helsinki-NLP/opus-mt, facebook/nllb-200-distilled-600M та facebook/hf-seamless-m4t-medium. Запропоновано методику покращення перекладу шляхом застосування моделі Vamsi/T5_Paraphrase_Paws для переформулювання текстів та використання sentence-transformers/LaBSE для вибору оптимального варіанту перекладу. Дослідження підтвердило, що покращена модель NLLB improved забезпечує підвищення точності перекладу, особливо для низькоресурсних мов.

MACHINE TRANSLATION, NEURAL NETWORKS, NATURAL LANGUAGE PROCESSING, TRANSFORMER, NLP

The object of the study is the process of automated translation of text documents using natural language processing methods.

The purpose of the work is to develop a methodology for improving the quality of automated document translation through analysis, experimental testing, and enhancement of neural models, with a focus on low-resource languages.

Development methods are based on technologies such as Python, FastAPI, React, PostgreSQL, Hugging Face Transformers, and PyTorch.

As a result of the work, experimental testing of machine translation models was conducted, including facebook/m2m100, Helsinki-NLP/opus-mt, facebook/nllb-200-distilled-600M, and facebook/hf-seamless-m4t-medium. A methodology for improving translation was proposed by applying the Vamsi/T5_Paraphrase_Paws model for text paraphrasing and using sentence-transformers/LaBSE for selecting the optimal translation variant. The study confirmed that the enhanced NLLB improved model provides increased translation accuracy, particularly for low-resource languages.

Завідувачу кафедри

ПІ
(скорочена назва кафедри)

проф. Кирилу СМЕЛЯКОВУ
(вчене звання, сласне ім'я, прізвище)

ЗАЯВА

щодо самостійності виконання кваліфікаційної роботи та можливості її публікації (та/або публікації анотації кваліфікаційної роботи) в електронному архіві відкритого доступу ElAr KhNURE

Я,

Андрійченко Артем Віталійович
(прізвище, ім'я, по батькові)

здобувач вищої освіти на другому (магістерському) рівні вищої освіти академічної групи ІПЗм-23-3

кафедра програмної інженерія
(повна назва кафедри)

заявляю: моя кваліфікаційна робота на тему

«Дослідження методів обробки природної мови, що використовуються для перекладу текстових документів»,
(назва роботи)

що буде представлена в екзаменаційну комісію для публічного захисту, виконана самостійно, в ній не містяться елементи плагіату і вона може бути опублікована в електронному архіві відкритого доступу ElArKhNURE. Всі запозичення з друкованих та електронних джерел мають відповідні посилання.

Я ознайомлений(на) з діючим положенням «Про протидію академічному плагіату в ХНУРЕ», згідно з яким виявлення плагіату є підставою для відмови в допуску кваліфікаційної роботи до захисту та застосування дисциплінарних заходів.

Дата Підпис

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ

NLP (Natural Language Processing) – обробка природної мови;

RNN (Recurrent Neural Network) – рекурентна нейронна мережа;

SMT (Statistical Machine Translation) – статистичний машинний переклад;

LSTM (Long Short-Term Memory) – довготривала короткочасна пам'ять;

GRU (Gated Recurrent Unit) – рекурентна одиниця з воротами;

Transformer – архітектура нейронної мережі з механізмом самоуваги;

EM (Expectation-Maximization) – алгоритм очікування-максимізації для тренування статистичних моделей;

NLLB (No Language Left Behind) – модель для багатомовного перекладу, ефективна для низькоресурсних мов;

OPUS-MT (Open Parallel Corpus Machine Translation) – набір відкритих моделей для швидкого двомовного перекладу;

M2M100 (Many-to-Many 100) – модель для прямого перекладу між 100 мовами;

SeamlessM4T (Seamless Multilingual Multimodal Machine Translation) – модель для перекладу тексту та мовлення в реальному часі;

T5 (Text-to-Text Transfer Transformer) – універсальна модель для задач NLP у форматі текст-до-текст;

BART (Bidirectional and Auto-Regressive Transformer) – модель із комбінацією кодувальника та декодувальника для генеративних задач;

LaBSE (Language-agnostic BERT Sentence Embedding) – модель для створення багатомовних семантичних ембедінгів;

CPU (Central Processing Unit) – центральний процесор;

GPU (Graphics Processing Unit) – графічний процесор;

TPU (Tensor Processing Unit) – спеціалізований процесор для машинного навчання.

ВСТУП

У сучасному світі інформація є одним із найцінніших ресурсів, а доступ до неї різними мовами – ключова умова для міжнародної співпраці, наукового обміну, економічного розвитку та культурної взаємодії. Стрімке зростання обсягів текстових даних створює потребу в ефективних інструментах для їх автоматизованої обробки та трансляції. Технології обробки природної мови (NLP) відкривають нові можливості для підвищення точності, швидкості та якості машинного перекладу, роблячи його масштабованим і придатним для використання в різних галузях [1].

Наукова проблема цього дослідження полягає в розробці методів забезпечення високої якості автоматизованого перекладу текстів за допомогою нейронних моделей, які враховують семантичні зв’язки, мовну специфіку та контекст. Попри значні досягнення в галузі NLP, зокрема завдяки архітектурі Transformer [1], залишаються виклики: складність адаптації моделей до малоресурсних мов [2], потреба у великих обсягах навчальних даних, висока обчислювальна складність і явище "галюцинацій", коли перекладений текст не відповідає змісту оригіналу [3]. Ці обмеження вимагають подальших теоретичних і прикладних досліджень.

Актуальність роботи зумовлена зростаючим попитом на точну, швидку та економічно вигідну трансляцію великих обсягів текстів у таких сферах, як міжнародний бізнес, державне управління, наука, освіта та соціальні комунікації. Традиційний ручний переклад потребує значних часових і людських ресурсів, що не дозволяє оперативно обробляти багатомовну інформацію в умовах глобалізації. Нейронні моделі забезпечують швидке та адаптивне відтворення текстів, зберігаючи їхню семантичну та стилістичну відповідність [4-8].

Мета дослідження – розробка методики підвищення якості автоматизованого перекладу документів шляхом аналізу, експериментального тестування та вдосконалення нейронних моделей. Для досягнення мети виконано аналіз сучасних підходів, протестовано моделі перекладу на багатомовних корпусах, а також

розроблено метод обробки текстів через генерацію альтернативних формулювань за допомогою моделі T5_Paraphrase_Paws [9] і їх семантичне оцінювання з використанням LaBSE [10]. Особливу увагу приділено перекладу малоресурсних мов, де традиційні моделі демонструють нижчу ефективність.

Об’єкт дослідження – процес автоматизованого перекладу текстових документів на основі алгоритмів обробки природної мови.

Предмет дослідження – методи машинного перекладу, засновані на нейронних мережах, та підходи до підвищення їхньої точності. У роботі застосовано порівняльний аналіз, експериментальне тестування, кількісне та якісне оцінювання результатів, а також моделювання з використанням сучасних архітектур машинного навчання.

Наукова новизна полягає в інтеграції моделей переформулювання та семантичної оцінки в процес перекладу, що дозволило створити вдосконалену версію моделі NLLB. Цей підхід забезпечує вищу точність, особливо для малоресурсних мов, шляхом адаптації перекладу до контексту та зменшення семантичних і стилістичних помилок.

Практична цінність результатів полягає в можливості використання розробленої методики в системах машинного перекладу для комерційних і наукових проєктів. Вона придатна для локалізації програмного забезпечення, обробки офіційної документації, а також перекладу технічної, юридичної та наукової літератури.

Ця робота є важливим кроком у розвитку технологій автоматизованого перекладу, сприяючи створенню ефективних інструментів для обробки багатомовної інформації. У майбутньому планується розширити дослідження в напрямку оптимізації нейронних моделей для рідкісних мов і розробки адаптивних систем, які враховують культурні та контекстуальні особливості.

1 АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ

Автоматичний переклад документів є ключовою складовою сучасної індустрії NLP і невід'ємною частиною глобальної комунікації. Зростання цифровізації та обсягів інформації, що поширюється різними мовами, створює потребу в інструментах для швидкого, ефективного та точного перекладу. Автоматизація цього процесу підвищує продуктивність у діловій сфері, наукових дослідженнях, державному управлінні та міжнародних проєктах. Однак досягнення високої якості перекладу вимагає вирішення низки технологічних і лінгвістичних викликів, що стимулюють розвиток цієї галузі.

1.1 Історія виникнення NLP

Обробка природної мови зародилася в 1950-х роках, коли виникла ідея машинного перекладу. Одним із перших проєктів був переклад текстів із російської на англійську мову для потреб військової розвідки. Ці системи базувалися на правилах граматики та словниках, що забезпечувало базовий рівень перекладу. Проте ранні підходи були обмеженими, оскільки не могли враховувати складні граматичні конструкції та мовні нюанси.

У 1990-х роках з'явилися статистичні моделі перекладу, зокрема IBM Model 1–5, які використовували ймовірнісні алгоритми та великі текстові корпуси для визначення найімовірнішого перекладу [12]. Хоча ці моделі перевершували попередні, вони не вирішували проблеми контекстуального перекладу та багатозначності слів.

Справжній прорив у NLP відбувся в 2010-х роках із появою глибинного навчання та нейронних мереж. Рекурентні нейронні мережі (RNN) і механізм уваги значно покращили точність і контекстуальність перекладу, дозволяючи враховувати зв'язки між словами в усьому тексті. Сьогодні стандарт у галузі встановлюють моделі Transformer, які забезпечують високу швидкість і якість перекладу, зокрема в реальному часі [4].

Статистичний машинний переклад (SMT) був одним із перших автоматизованих підходів. Він базується на статистичних закономірностях у паралельних текстах двох мов. Ключовим елементом SMT є паралельні корпуси, які використовуються для навчання [13].

Основний алгоритм SMT базується на Байєсовому підході, який визначає ймовірність того, що вихідний текст e перекладений на цільову мову f[12]. Його можна описати за формулою 1.1:

$$P(e|f) = \frac{P(f|e)P(f)}{P(f)}, \quad (1.1)$$

де $P(e|f)$ – ймовірність перекладу тексту e за умови вхідного тексту f;

$P(f|e)$ – ймовірність моделі перекладу;

$P(e)$ – ймовірність мовної моделі;

$P(f)$ – ймовірність вхідного тексту f , яка в контексті SMT часто є нормувальним множником і не залежить від e .

Це дозволяє здійснювати переклад на основі статистичних закономірностей у текстах.

Алгоритми SMT використовують кілька підходів:

– переклад на рівні окремих слів;
– переклад на рівні фраз, що покращує точність перекладу;
– переклад на основі ієрархічних фразових структур, що дозволяє враховувати синтаксичну структуру мов.

Основна проблема SMT – потреба у великих обсягах паралельних текстів. Моделі також схильні до помилок у складних реченнях через брак статистичних даних.

Для вибору оптимального перекладу використовується алгоритм Вітербі [12], який обчислює ймовірності варіантів і вибирає найімовірніший. Однак через імовірнісну природу алгоритм може помилятися за недостатньої кількості даних.

Алгоритм Expectation-Maximization (EM) застосовується для тренування SMT-моделей. Він складається з двох етапів: очікування (обчислення очікуваних

значень параметрів) і максимізації (оновлення параметрів моделі). Цей ітераційний процес підвищує точність моделей.

З появою нейронних мереж NLP перейшла на новий рівень. RNN стали першими архітектурами для обробки послідовних даних, зберігаючи інформацію про попередні стани. Однак проблема зникнення градієнтів обмежувала їхню здатність обробляти довгі послідовності.

Модель LSTM (Long Short-Term Memory) вирішила цю проблему завдяки механізмам «запам’ятовування» та «забування» інформації [14]. LSTM ефективно обробляє довгі тексти, але потребує значних обчислювальних ресурсів через велику кількість параметрів.

GRU (Gated Recurrent Unit) – спрощена версія LSTM із меншою кількістю параметрів, що забезпечує швидше тренування при збереженні ефективності [15]. Вона оптимальна для багатьох практичних завдань, хоча менш ефективна для дуже довгих текстів.

Модель Transformer, запропонована у 2017 році, здійснила революцію в NLP завдяки механізму самоуваги, який дозволяє паралельно обробляти всі слова в реченні [4]. Це значно підвищило швидкість і якість перекладу порівняно з RNN, LSTM і GRU.

Порівняльний аналіз моделей наведено в таблиці 1.1, яка ілюструє їхні переваги, недоліки та застосування.

Таблиця 1.1 – Порівняльна таблиця моделей

Модель	Переваги	Недоліки	Застосування
RNN	Врахування послідовності даних	Проблема затухання градієнтів, обмеження на довгі послідовності	Простий переклад, аналіз послідовностей
LSTM	Добра здатність запам'ятовувати довгі послідовності	Велика кількість параметрів, повільне тренування	Переклад довгих текстів
GRU	Спрощена архітектура, швидше тренування, менше параметрів	Менш ефективний для довгих текстів	Оптимальні для багатьох практичних задач
Transformer	Паралельна обробка, висока швидкість і точність	Висока вимога до ресурсів, складна архітектура	Сучасні системи перекладу, GPT, BERT

Історія NLP відображає еволюцію від правил-базованих систем до статистичних моделей і сучасних нейронних архітектур. Поява Transformer стала ключовим етапом, що забезпечив високу точність і швидкість перекладу, визначивши сучасний стандарт у галузі.

1.2 Обмеження та виклики існуючих підходів

Незважаючи на прогрес у NLP, сучасні системи перекладу мають низку обмежень:

- обчислювальні ресурси: моделі, такі як Transformer, потребують потужних графічних (GPU) або спеціалізованих (TPU) процесорів, що обмежує їх доступність для невеликих організацій [4];
- рідкісні мови: для мов із обмеженими текстовими даними (наприклад, деяких слов'янських чи угорсько-фінських мов) бракує корпусів для навчання [2], також трансферне навчання частково вирішує цю проблему, але не повністю;
- контекст і полісемія: сучасні моделі іноді неправильно інтерпретують багатозначні слова (наприклад, англ. bank – «банк» чи «берег») або ідіоматичні вирази, що призводить до втрати точності;
- спеціалізовані тексти: переклад технічної чи юридичної документації вимагає спеціалізованих термінологічних баз і високої точності, що залишається викликом [16];
- етичні аспекти: моделі можуть спотворювати культурний зміст або стирати національну ідентичність. Наприклад, ідіоми, специфічні для української культури, можуть бути неправильно перекладені, втрачаючи сенс;
- галюцинації: нейронні моделі можуть генерувати переклади, які не відповідають оригінальному тексту, через надмірне узагальнення або недостатнє розуміння контексту [3].

Подолання цих викликів вимагає вдосконалення методів навчання, розробки нових архітектур і адаптації моделей до культурних контекстів.

1.3 Огляд основних джерел

Дослідження методів обробки природної мови для автоматизованого перекладу відіграють ключову роль у сучасній комп'ютерній лінгвістиці, що пояснюється стрімким зростанням обсягів текстової інформації та потребою в її швидкому й точному перекладі різними мовами. Значний прорив у цій галузі став можливим завдяки появі архітектури Transformer, яка замінила традиційні методи послідовної обробки даних на підхід, що дозволяє одночасно аналізувати весь текст [1]. Цей механізм суттєво підвищив швидкість обробки текстів і покращив якість перекладу завдяки кращому врахуванню контекстуальних зв'язків між словами та реченнями. На основі цієї архітектури було розроблено низку моделей машинного перекладу, кожна з яких має унікальні характеристики та призначення.

Архітектура Transformer спирається на механізми самоуваги, які дозволяють паралельно обробляти вхідні послідовності [1]. На рисунку 1.1 зображено основну структуру Transformer, де виділено багатоголову увагу та прямі нейронні шари в стеках кодувальника та декодувальника.

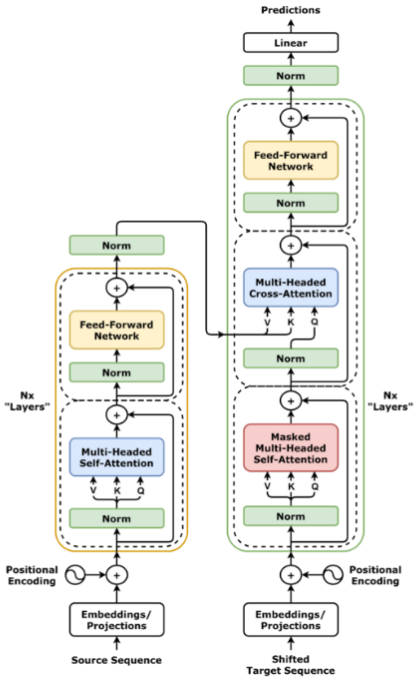


Рисунок 1.1 – Архітектура Transformer (за даними[1])

Модель NLLB вирізняється своєю здатністю перекладати тексти між великою кількістю мов, включно з тими, для яких доступні лише обмежені обсяги даних [6]. Вона базується на складній структурі, що включає кодувальник і декодувальник, а також спеціальний механізм розподілу обчислювальних ресурсів, що забезпечує високу гнучкість і можливість масштабування. Додаткові технічні аспекти її реалізації та приклади адаптації до різних мовних пар розкрито в джерелі [17].

Модель OPUS-MT є набором відкритих моделей для машинного перекладу, оптимізованих для швидкості та ефективності [4]. Вона підтримує широкий спектр мовних пар і використовується в проєктах, що потребують швидкого перекладу на стандартному обладнанні, наприклад, CPU. Технічні характеристики та приклади практичного застосування OPUS-MT описані в джерелі [18].

Модель M2M100 забезпечує прямий переклад між великою кількістю мов без використання проміжної мови як посередника [5]. Такий підхід дозволяє уникнути спотворень, що виникають при додаткових етапах перекладу, і зберегти оригінальний зміст та стиль тексту. Технічні деталі реалізації M2M100 представлені в джерелі [19].

Модель SeamlessM4T розширює можливості перекладу, охоплюючи не лише текст, а й мовлення, і підтримує роботу з великою кількістю мов [7]. Вона дозволяє створювати переклади в реальному часі, що робить її перспективною для інтеграції в додатки, які потребують одночасної обробки різних типів даних. Технічні особливості SeamlessM4T описані в джерелі [20].

Модель T5 (Text-to-Text Transfer Transformer) є універсальною моделлю, яка адаптується до різних завдань NLP, включно з перекладом, через формулювання всіх завдань як текст до тексту [8]. Її гнучкість і висока точність роблять її популярною для багатомовних застосунків.

Модель BART поєднує можливості кодувальника та декодувальника, що дозволяє їй ефективно працювати з завданнями перекладу та генерації тексту [21]. Вона демонструє високу якість перекладу для мов із достатньою кількістю даних.

Порівняно зі статистичними методами, які використовували паралельні корпуси для зіставлення слів і фраз [13], сучасні моделі на основі Transformer забезпечують кращу обробку контексту та складних граматичних структур. Проте вони мають обмеження, такі як потреба у великих обсягах даних для навчання, що ускладнює їх використання для рідкісних мов [6]. Феномен «галюцинацій», коли моделі генерують некоректні переклади через недостатнє розуміння контексту, залишається актуальною проблемою [3]. Для оцінки якості перекладу часто використовується метрика BLEU, але вона не враховує семантичні нюанси. Альтернативний підхід, описаний у [22], базується на аналізі семантичної схожості за допомогою векторних представлень, що краще відображає змістову відповідність.

Сучасні моделі демонструють значний прогрес у перекладі, але потребують вдосконалення для роботи зі спеціалізованими текстами (юридичними, технічними) та зменшення обчислювальних витрат. Наприклад, NLLB ефективна для рідкісних мов, але потребує значних ресурсів, тоді як OPUS-MT оптимальна для швидкого перекладу на стандартному обладнанні. Дослідження в цій галузі залишаються актуальними через зростаючу потребу в автоматизованому перекладі в умовах глобалізації.

1.4 Постановка задачі

Метою цього проєкту є дослідження методів NLP, зокрема моделей на основі архітектури Transformer, та їх адаптація для автоматизованого перекладу текстів. Для цього потрібно реалізувати програмне забезпечення, яке виконуватиме переклад за допомогою вибраних методів та порівнюватиме їх за критеріями.

Для реалізації системи використовуватиметься технологічний стек, що включає Python для серверної частини з огляду на його потужну підтримку NLP та машинного навчання, React для клієнтської частини, а також PostgreSQL для зберігання даних через його ефективність у роботі з різними типами даних [23].

Завдання дослідження:

1. розробка архітектури системи для інтеграції різних моделей NLP для перекладу текстів;
2. впровадження серверної логіки для обробки запитів на переклад, управління моделями та зберігання даних;
3. оцінка ефективності обраних моделей NLP за критеріями точності, швидкості, вимог до ресурсів, масштабованості та гнучкості налаштування;
4. удосконалення найкращої моделі шляхом переформулювання перекладів і вибору оптимальних варіантів на основі аналізу семантичної схожості;
5. впровадження веб частини, щоб забезпечити зручний інтерфейс для запуску перекладів.

Архітектура системи є першим кроком, оскільки вона визначає основу для всіх наступних етапів. Вона встановлює структуру взаємодії між компонентами, включаючи вибір технологій, і визначає, як моделі NLP інтегруються в загальну систему.

Серверна логіка розробляється наступною, оскільки вона відповідає за обробку даних, виконання перекладів, управління моделями та зберігання інформації. Це ключовий компонент системи, який забезпечує її функціональність і підготовку до подальшого аналізу.

Оцінка ефективності моделей NLP проводиться після реалізації системи, коли вона вже здатна виконувати переклади. Це дозволяє провести порівняльний аналіз між моделями в реальних умовах, виявити їх сильні та слабкі сторони, а також визначити базові параметри для подальшого вдосконалення.

Удосконалення моделей NLP. На цьому етапі результати оцінки слугують основою для покращення якості перекладу шляхом переформулювання тексту та вибору найкращих варіантів із використанням методів аналізу семантичної схожості. Це сприяє підвищенню точності перекладу, збереженню смислу оригіналу та загальній ефективності системи.

Веб-частину є останнім кроком. Вона потрібна для забезпечення зручного інтерфейсу, який дозволяє користувачам запускати переклади, переглядати результати, працювати з датасетами, аналізувати графіки BLEU-оцінювання, авторизуватися та зберігати історію перекладів.

На основі досліджень, для вибору найкращого методу обробки природної мови для автоматизованого перекладу текстових документів застосовано багатокритеріальну оптимізацію. Такий підхід обумовлений необхідністю врахування різних характеристик методів, які мають неоднаковий вплив на їхню ефективність у різних умовах. Наприклад, висока точність перекладу є ключовою для збереження семантичної відповідності текстів, тоді як швидкість виконання може бути критичною для інтерактивних або реального часу застосунків. Використання лише одного критерію для вибору методу призвело б до ігнорування інших важливих аспектів, що знижує практичну цінність такого вибору.

Для порівняння методів визначено п'ять критеріїв:

– точність перекладу: ключовий показник якості перекладу, що відображає рівень семантичної відповідності між текстом оригіналу та його перекладом. Вимірюється за допомогою метрик, таких як BLEU або METEOR;
– швидкість виконання: кількість текстів, які модель може обробити за одиницю часу. Цей критерій є важливим для інтерактивних застосунків і систем із високими вимогами до продуктивності;
– вимоги до обчислювальних ресурсів: рівень апаратних ресурсів, необхідних для роботи моделі (наприклад, CPU, GPU чи TPU). Цей критерій враховує як економічну, так і технічну доступність;
– масштабованість: здатність моделі працювати з великими корпусами тексту без втрати ефективності. Особливо важливо для систем, що обробляють великі обсяги даних, наприклад, багатомовні платформи;
– гнучкість у налаштуванні: можливість адаптації моделі до специфічних мовних пар, доменів або завдань. Цей критерій розширює практичне застосування моделі.

Для аналізу обрано сучасні моделі NLP: NLLB, OPUS-MT, SeamlessM4T, M2M100, T5 і BART, які представляють різні архітектури та підходи. Їх порівняння проведено за визначеними критеріями з використанням номінальних шкал.

Для порівняння моделей необхідно не лише визначити критерії, але й надати їм конкретні числові значення, які відображають рівень їх реалізації. Оскільки критерії є різнотипними, потрібно створити шкали, які дозволяють зіставляти різні показники між собою. Це забезпечує можливість об'єктивного порівняння моделей за всіма критеріями одночасно.

Шкали критеріїв описані наступним чином та записані до таблиці (див. табл. 1.2):

– точність перекладу: визначаються номінальною шкалою: висока, середня;
– швидкість виконання: визначаються номінальною шкалою: низька, середня, висока;
– вимоги до ресурсів: визначаються номінальною шкалою: низькі, високі, дуже високі;
– масштабованість: визначаються номінальною шкалою: середня, висока;
– гнучкість у налаштуванні: визначаються номінальною шкалою: середня, висока;

Таблиця 1.2 – Критеріальна оцінка моделей

Модель	**Точність перекладу**	**Швидкість виконання**	**Вимоги до ресурсів**	**Масштабованість**	**Гнучкість у налаштуванні**
NLLB	Висока	Середня	Високі	Висока	Висока
OPUS-MT	Середня	Висока	Низькі	Середня	Середня
SeamlessM4T	Висока	Середня	Дуже високі	Висока	Висока
M2M100	Висока	Середня	Високі	Висока	Середня
T5	Середня	Середня	Високі	Середня	Висока
BART	Середня	Низька	Високі	Середня	Середня

Для забезпечення можливості кількісного аналізу шкали були переведені в числові значення (див. табл. 1.3). Це дозволяє проводити математичні операції для порівняння моделей і знаходження оптимального рішення. Переведення шкал виконано наступним чином:

– для точності перекладу: висока – 10, середня – 6, низька – 2;
– для швидкісті виконання: висока – 10, середня – 6, низька – 2;
– для вимоги до ресурсів: дуже високі – 0, високі – 1, низькі 3;
– для масштабованості: висока – 3, середня – 2, низька – 1;
– для гнучкості у налаштуванні: висока – 2, середня – 1, низька – 0.

Таблиця 1.3 – Кількісні значення моделей

Модель	**Точність перекладу**	**Швидкість виконання**	**Вимоги до ресурсів**	**Масштабованість**	**Гнучкість у налаштуванні**
NLLB	10	6	1	3	2
OPUS-MT	6	10	3	2	1
SeamlessM4T	10	6	0	3	2
M2M100	6	6	1	3	1
T5	6	6	1	2	2
BART	6	2	1	2	1

Для визначення оптимальних альтернатив використано принцип Парето (див. табл. 1.4), який виключає доміновані моделі. Модель BART є домінованою, оскільки має нижчу швидкість виконання та не переважає за іншими критеріями порівняно з іншими моделями. Модель T5 також виключається, оскільки не демонструє значних переваг за критеріями точності та масштабованості.

У множині Парето залишилися NLLB, OPUS-MT, SeamlessM4T та M2M100.

Таблиця 1.4 – Множина Парето

Модель	**Точність перекладу**	**Швидкість виконання**	**Вимоги до ресурсів**	**Масштабованість**	**Гнучкість у налаштуванні**
NLLB	10	6	1	3	2
OPUS-MT	6	10	3	2	1
SeamlessM4T	10	6	0	3	2
M2M100	6	6	1	3	1

Для подальшого аналізу було виконано нормування оцінок за всіма критеріями, за формулою 1.2, що дозволило привести значення до інтервалу [0, 1].

$$X = \frac{X_{\text{розрахункове}} - X_{min}}{X_{max} - X_{min}}, \quad (1.2)$$

де X – вихідне значення критерію;

$X_{мін}$ – мінімальне значення по критерію;

$X_{макс}$ – максимальне значення по критерію.

Після перетворень маємо таблицю (див. табл. 1.5).

Таблиця 1.5 – Нормовані значення критеріїв

Модель	**Точність перекладу**	**Швидкість виконання**	**Вимоги до ресурсів**	**Масштабованість**	**Гнучкість у налаштуванні**
NLLB	1	0	0.33	1	1
OPUS-MT	0	0	1	0	0
SeamlessM4T	1	1	0	1	1
M2M100	1	0	0.33	1	0
min	6	6	0	2	1
max	10	10	3	3	2

Після нормування оцінок за всіма критеріями наступним кроком є визначення загальної корисності кожної моделі. Для цього використовується метод лінійної адитивної згортки, який дозволяє об'єднати всі критерії в один інтегральний показник.

Лінійна адитивна згортка передбачає обчислення зваженої суми нормованих значень критеріїв для кожної альтернативи. Використаємо формулу 1.3 для її розрахунку:

$$Z^* = \max_{i=1,m} \sum_{j=1}^{n} \alpha_j \beta_j a_{ij}, \quad (1.3)$$

де $\alpha_j \alpha_{ij}$ – нормуруючі множники;

β_j – вагові коефіцієнти.

Для врахування важливості різних критеріїв їм було присвоєно вагові коефіцієнти на основі експертної оцінки. Критерії ранжуються за важливістю, після чого вагові коефіцієнти обчислюються за формулою 1.4:

$$\beta_j = \frac{r_j}{\sum_{j=1}^{n} r_j}, \tag{1.4}$$

де r_j – ранг критерію.

Розподіл рангів і ваг:

– точність перекладу: ранг = 5, вага = 0,333;

– швидкість виконання: ранг = 4, вага = 0,267;

– вимоги до ресурсів: ранг = 3, вага = 0,2;

– масштабованість: ранг = 2, вага = 1,333;

– гнучкість у налаштуванні: ранг = 1, вага = 0,067.

Після визначення ваг та визначення формули лінійної адаптивної згортки заповнимо таблицю (див. табл. 1.6).

Таблиця 1.6 – Порівняння моделей перекладу за лінійною адаптивною оцінкою

Модель	**Точність перекладу**	**Швидкість виконання**	**Вимоги до ресурсів**	**Масшта-бованість**	**Гнучкість у налаштуванні**	**Z***
NLLB	1	0	0.33	1	1	0.599
OPUS-MT	0	0	1	0	0	0.467
SeamlessM4T	1	1	0	1	1	0.533
M2M100	1	0	0.33	1	0	0.532
β	0,333	0,267	0,2	0.133	0,067	
α	1	0,1	0,1	1	0,1	

За результатами багатокритеріальної оптимізації моделі NLLB, SeamlessM4T, M2M100 і OPUS-MT обрано для подальшого дослідження, оскільки вони забезпечують оптимальний баланс між критеріями точності, швидкості, вимог до ресурсів, масштабованості та гнучкості. Модель NLLB демонструє найвищу загальну корисність завдяки високій точності, масштабованості та гнучкості, хоча й потребує значних ресурсів. SeamlessM4T і M2M100 також показують високі

результати, особливо для багатомовних і реального часу застосунків. OPUS-MT вирізняється швидкістю та низькими вимогами до ресурсів, що робить її цінною для практичних двомовних задач.

Надалі необхідно порівняти ці моделі у реальних умовах, оцінити їх продуктивність на тестових наборах даних і підтвердити отримані результати експериментально. Це дозволить визначити оптимальну модель або комбінацію моделей для автоматизованого перекладу текстових документів із урахуванням специфіки мовних пар і вимог до системи. Крім того, результати оцінки будуть використані для удосконалення обраної моделі шляхом переформулювання перекладів і аналізу семантичної схожості, що сприятиме підвищенню якості перекладу.

2 ОПИС ПРИЙНЯТИХ ПРОЄКТНИХ РІШЕНЬ

2.1 Вибір технологій

Для розробки програмного забезпечення автоматизованого перекладу текстів було обрано технології React, Python і PostgreSQL, оскільки вони забезпечують необхідну функціональність, продуктивність і гнучкість у рамках даного проєкту. React виступає основою для реалізації динамічного та зручного користувацького інтерфейсу, що є важливим для забезпечення швидкого доступу до функціональності перекладу.

Python було обрано для серверної частини системи завдяки його широкій підтримці алгоритмів машинного навчання та природної мови. Python забезпечує просту інтеграцію з NLP-моделями, що дозволяє виконувати переклади з високою точністю. Також ця мова програмування підтримує створення REST API для обробки запитів від клієнтського інтерфейсу, що спрощує взаємодію між компонентами системи.

Для зберігання даних було обрано PostgreSQL, яка добре поєднує підтримку структурованих і напівструктурованих даних. Це дозволяє зберігати інформацію про користувачів, історію перекладів і метадані NLP-моделей у структурованому вигляді, а також корпуси текстів у форматі JSONB. PostgreSQL забезпечує швидкий доступ до даних і підтримує ACID-транзакції, що гарантує їхню цілісність.

Для цієї системи була розроблена діаграма розгортання (див. рис. 2.1).

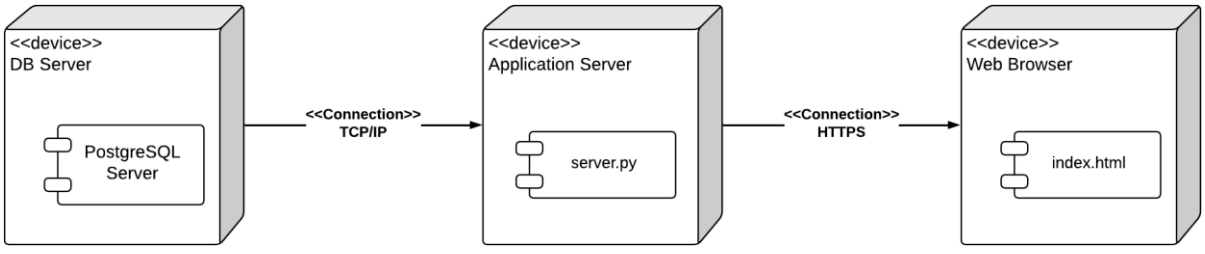


Рисунок 2.1 – UML діаграма розгортання проекту

(рисунок створено самостійно)

Система складається з клієнтської та серверної частин, які взаємодіють через REST API. Користувач через інтерфейс у браузері вводить текст для перекладу, завантажує файл, вибирає мови та моделі. React-інтерфейс обробляє введені дані та надсилає їх у вигляді HTTP-запиту на серверну частину, реалізовану за допомогою Python.

На сервері Python обробляє запит, використовуючи модель, для виконання перекладу. Параметри перекладу й результат зберігаються в базі даних PostgreSQL для подальшого аналізу або перегляду. Сервер повертає перекладений текст React-додатку, який відображає його у веб-інтерфейсі.

2.2 Бібліотеки та інструменти

Для реалізації та перевірки обраних моделей знадобляться наступні бібліотеки:

- transformers (версія 4.49.0): бібліотека для роботи з передовими моделями NLP. Містить моделі, токенізатори, засоби для тонкого налаштування;
- torch (версія 2.6.0): бібліотека для роботи з нейронними мережами;
- tokenizers (версія 0.13.0): швидка токенізація текстів для NLP. Підготовка текстів перед передачею в моделі;
- datasets (версія 2.14.0): робота з популярними наборами даних. Завантаження корпусів текстів для навчання/тестування;
- nltk (версія 3.9.1): інструмент для обробки природної мови. Обчислення BLEU, METEOR, токенізація текстів;
- scikit-learn (версія 1.6.1): аналіз даних та метрики. Оцінка точності моделей;
- SQLAlchemy (версія 2.1.0): інтеграція PostgreSQL з Python;
- psycopg2 (версія 2.9.0): драйвер для PostgreSQL.

2.3 Використані алгоритми

У рамках дослідження було прийнято рішення зосередити увагу на трьох ключових алгоритмах, які реалізують основні етапи обробки природної мови: попередню обробку вхідного тексту, генерацію варіантів перекладу або переформулювання, а також вибір найоптимальнішого результату з точки зору змістової відповідності оригіналу.

Оскільки сучасні трансформерні моделі мають обмеження на максимальну довжину вхідної послідовності (як правило, 512 або 1024 токени), було реалізовано алгоритм, що розбиває вхідний текст на менші логічно завершені частини. Основною одиницею розбиття виступає речення, що дозволяє зберегти цілісність змісту. Алгоритм поєднує речення у фрагменти, загальна довжина яких не перевищує допустимий розмір, завдяки чому забезпечується сумісність із трансформерами та мінімізується ризик втрати контексту.

Для генерації альтернативних варіантів речень використовується спеціалізована модель Vamsi/T5_Paraphrase_Paws, яка була попередньо навчена на завданні парафразування. До вхідного тексту додається службовий префікс, що вказує на бажаний тип генерації, після чого модель створює кілька переформульованих варіантів речення. Алгоритм використовує стохастичні методи генерації, такі як top-k sampling, nucleus sampling і beam search. Це забезпечує варіативність результатів та зменшує ймовірність повторення шаблонних відповідей, дозволяючи сформувати ширший спектр можливих перекладів або варіантів формулювання думки.

Отримані варіанти речень аналізуються за допомогою моделі для створення семантичних ембедінгів – у дослідженні використовується багатомовна модель LaBSE. Кожне речення перетворюється на векторне представлення, після чого обчислюється косинусна подібність між вектором оригінального речення та кожним із кандидатів. Кандидат з найвищим показником подібності обирається як такий, що найкраще зберігає зміст вихідного тексту. Такий підхід дозволяє системі не лише генерувати переклад, але й підвищувати його якість за рахунок семантичного узгодження.

3 ОПИС ПРОГРАМНОЇ РЕАЛІЗАЦІЇ

3.1 Архітектура серверу

Серверна частина програмного забезпечення побудована на основі веб-фреймворку FastAPI, що забезпечує високу швидкість обробки запитів, асинхронність та модульність. Основними завданнями серверу є обробка запитів від клієнта, взаємодія з мовними моделями, оцінка якості перекладу, зберігання результатів у базі даних та забезпечення автентифікації користувачів.

Усі компоненти системи описані на діаграмі компонентів (див. рис. 3.1).

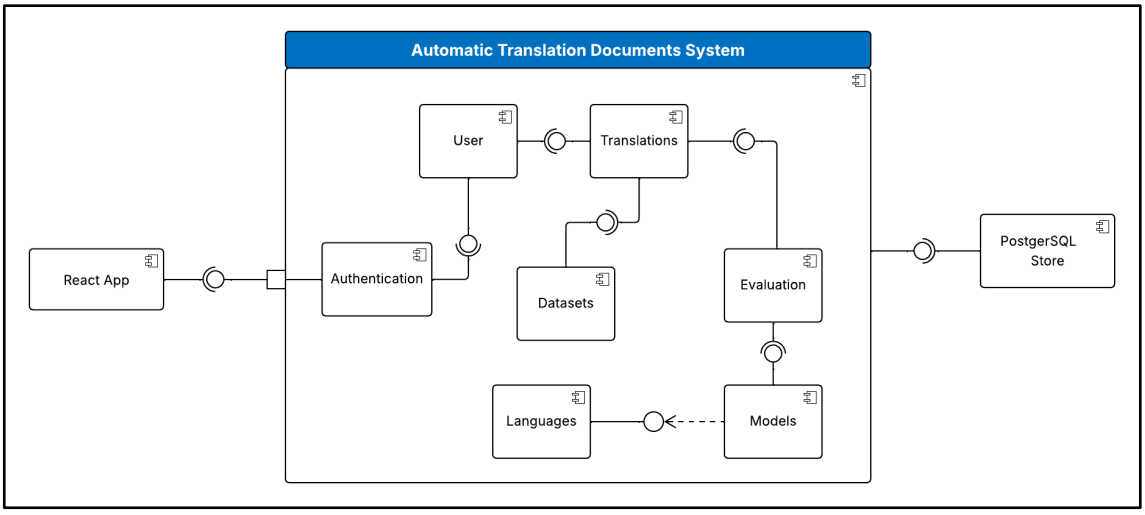


Рисунок 3.1 –UML діаграма компонентів серверної частини проекту (рисунок створено самостійно)

На рисунку 3.2 зображено логічну архітектуру взаємодії компонентів серверної частини системи. У центрі розташований FastAPI-застосунок, який отримує HTTP-запити від клієнта. Через маршрутизатор запити передаються до відповідних контролерів: для перекладу, роботи з датасетами чи автентифікації.

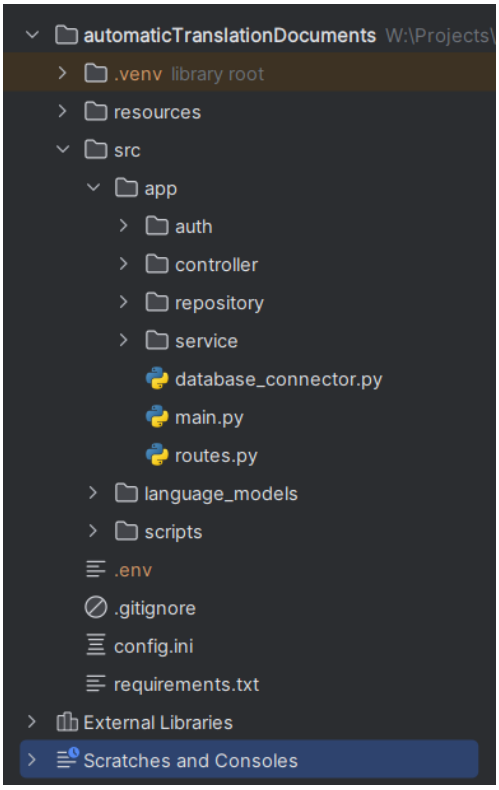


Рисунок 3.2 – Загальна архітектура проєкту (рисунок створено самостійно)

Контролери звертаються до сервісного рівня, де реалізовано основну бізнес-логіку. Сервіси, в свою чергу, або викликають NLP-моделі для перекладу, або працюють із базою даних через відповідні репозиторії. У випадку запитів, що потребують авторизації, використовується модуль JWT для генерації й перевірки токенів доступу.

Точка входу серверної частини реалізується через скрипт main.py, який використовує бібліотеку Uvicorn для запуску асинхронного веб-сервера FastAPI.

Модуль routes.py визначає основні маршрути API з використанням FastAPI. Він включає три роутери: translation_router, auth_router і datasets_router, які відповідають за обробку перекладів, автентифікацію та роботу з датасетами відповідно.

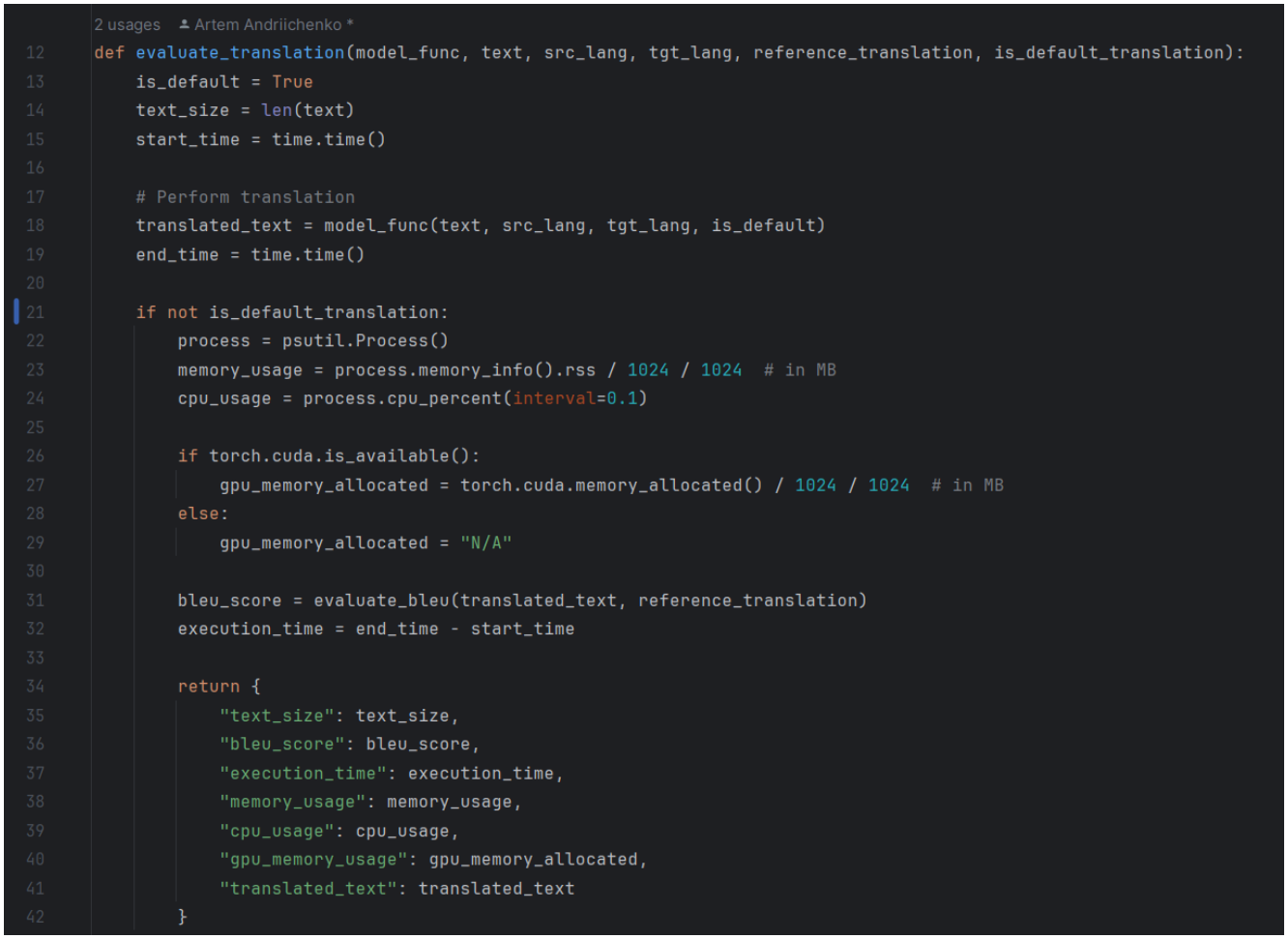
Модуль datasets_service.py відповідає за роботу з датасетами. Він забезпечує отримання шляхів до датасетів за їхніми назвами, завантаження текстів для заданих мов із відповідних файлів, сегментацію текстів на фрагменти для обробки та

виконання перекладу датасетів із використанням обраних моделей із подальшим збереженням результатів у CSV-файли (див. рис. 3.3).

```
3 usages  Artem Andriichenko
def run_dataset_translation(dataset_name, model_names, languages, user_id, start, end):
    path = get_dataset_path_by_name(dataset_name)
    texts = get_dataset_texts_by_name(dataset_name, languages, path)

    results = []

    for src_lang in languages:
        source_text = choice_sentences_from_texts_between(texts[src_lang], start, end)
        for tgt_lang in languages:
            target_text = choice_sentences_from_texts_between(texts[tgt_lang], start, end)
            for model_name in model_names:
                if src_lang != tgt_lang:
                    model_func = model_service.get_model_func_by_name(model_name)
                    print(f"Translating from {src_lang} to {tgt_lang} using {model_name}")
                    result = evaluation_service.evaluate_translation(model_func, source_text, src_lang,
                                                                     tgt_lang, target_text, is_default_translation: False)
                    result.update({"source_lang": src_lang, "target_lang": tgt_lang, "model": model_name})
                    results.append(result)

    csv_result_path = file_service.save_datasets_translation_to_file(user_id, results, model_names, start, end)
    return csv_result_path
```

Рисунок 3.3 – Метод run_dataset_translation в datasets_service.py (рисунок створено самостійно)

Модуль evaluation_service.py реалізує логіку оцінки якості перекладів. Він виконує переклад із використанням заданої моделі та обчислює метрики продуктивності: розмір тексту, час виконання, використання пам’яті, використання процесора. Якість перекладу оцінюється за метрикою BLEU. Також модуль передбачає вибір найкращого перекладу серед кандидатів, яка обчислює косинусну подібність між векторними представленнями оригінального тексту та перекладів (див. рис. 3.4).



```
2 usages  Artem Andriichenko *
def evaluate_translation(model_func, text, src_lang, tgt_lang, reference_translation, is_default_translation):
    is_default = True
    text_size = len(text)
    start_time = time.time()

    # Perform translation
    translated_text = model_func(text, src_lang, tgt_lang, is_default)
    end_time = time.time()

    if not is_default_translation:
        process = psutil.Process()
        memory_usage = process.memory_info().rss / 1024 / 1024  # in MB
        cpu_usage = process.cpu_percent(interval=0.1)

        if torch.cuda.is_available():
            gpu_memory_allocated = torch.cuda.memory_allocated() / 1024 / 1024  # in MB
        else:
            gpu_memory_allocated = "N/A"

        bleu_score = evaluate_bleu(translated_text, reference_translation)
        execution_time = end_time - start_time

        return {
            "text_size": text_size,
            "bleu_score": bleu_score,
            "execution_time": execution_time,
            "memory_usage": memory_usage,
            "cpu_usage": cpu_usage,
            "gpu_memory_usage": gpu_memory_allocated,
            "translated_text": translated_text
        }
```

Рисунок 3.4 – Метод evaluate_translation в evaluation_service.py (рисунок створено самостійно)

Модуль translation_repo.py відповідає за взаємодію з базою даних PostgreSQL через функції для отримання всіх перекладів для конкретного користувача за його ID та збереження даних про переклад. Для підключення до бази даних використовується бібліотека psycopg2 (див. рис. 3.5).

```
1 usage  Artem Andriichenko
def create_translation(user_id, model_id, file_path, source_lang, target_lang, is_default_translation, text_size,
                       bleu_score, execution_time, memory_usage, cpu_usage, gpu_memory_usage):
    conn = get_connection()
    cursor = conn.cursor()
    query = """
        INSERT INTO translation (user_id, model_id, file_path, source_lang, target_lang, is_default_translation,
        text_size, bleu_score, execution_time, memory_usage, cpu_usage, gpu_memory_usage)
        VALUES (%s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s)
        """
    cursor.execute(query, (user_id, model_id, file_path, source_lang, target_lang, is_default_translation, text_size,
                           bleu_score, execution_time, memory_usage, cpu_usage, gpu_memory_usage))
    conn.commit()
    conn.close()
```

Рисунок 3.5 – Метод create_translation в translation_repo.py

(рисунок створено самостійно)

Модуль translation_controller.py обробляє HTTP-запити, пов’язані з перекладами, отриманням мов і моделей. Він включає ендпоінти для отримання всіх перекладів користувача, виконання перекладу тексту для заданих моделей і мов, повернення списку доступних мов і моделей. Для валідації вхідних даних використовується Pydantic-модель (див. рис. 3.6)

```
Artem Andriichenko
@router.post("/translations")
def make_translation(request: TranslationRequest):
    try:
        results = translation_service.run_translations_for_models(
            user_id=request.user_id,
            model_names=request.model_names,
            source_lang=request.source_lang,
            target_lang=request.target_lang,
            text=request.text,
            reference_translation=request.reference_translation,
            is_default_translation=request.is_default_translation
        )
        return results
    except Exception as e:
        raise HTTPException(status_code=500, detail=f"Translating error: {str(e)}")
```

Рисунок 3.6 – Метод make_translation в translation_controller.py

(рисунок створено самостійно)

Модуль authentication_service.py реалізує логіку автентифікації та авторизації. Він забезпечує реєстрацію користувачів із хешуванням паролів за

допомогою passlib, аутентифікацію з перевіркою паролів і видачею JWT-токенів, а також вихід із системи з анулюванням токенів. Модуль взаємодіє з user_repo і token_repo для роботи з користувачами та токенами (див. рис. 3.7).

```
2 usages  Artem Andriichenko
def authenticate_user(data):
    user = user_repo.get_user_by_email(data.email)
    if not user or not pwd_context.verify(data.password, user["password"]):
        raise HTTPException(status_code=401, detail="Incorrect email or password")

    token_repo.revoke_all_tokens(user["id"])
    token = create_access_token({"sub": user["email"]})
    token_repo.save_token(user["id"], token)

    return {
        "token": token,
        "user": {
            "id": user["id"],
            "username": user["username"],
            "email": user["email"]
        }
    }
```

Рисунок 3.7 – Метод authenticate_user в authentication_service.py
(рисунок створено самостійно)

Архітектура характеризується асинхронністю завдяки FastAPI та Uvicorn, що підвищує продуктивність при обробці великих обсягів тексту. Модульна структура з розподілом функціональності на контролери, сервіси та репозиторії забезпечує легке масштабування та підтримку коду. Безпека досягається через налаштування CORS для обмеження доступу до клієнтських адрес і автентифікацію через JWT-токени. Гнучкість системи забезпечується підтримкою кількох моделей NLP і мовних пар, що дозволяє адаптувати її до різних сценаріїв перекладу.

3.2 ER-діаграма бази даних

Для проектування бази даних попередньо була розроблена ER-діаграма (див. рис. 3.8). Вона створена для забезпечення функціональності системи автоматизованого перекладу текстів, що включає управління користувачами,

перекладами, токенами авторизації та мовними моделями. Діаграма відображає логічну структуру чотирьох основних сутностей і зв'язки між ними, що забезпечують зберігання, аналіз та захист даних, пов'язаних із процесом перекладу.

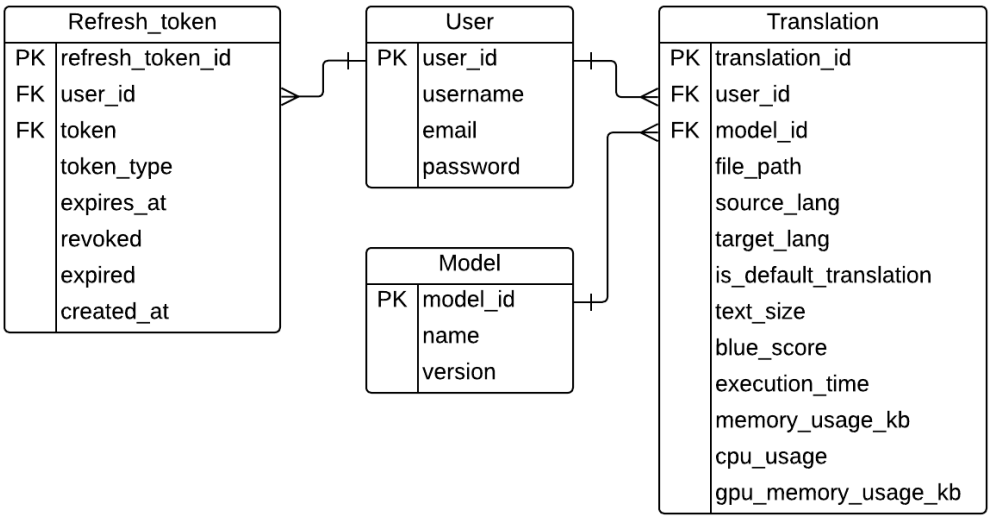


Рисунок 3.8 – ER-діаграма проекту (рисунок створено самостійно)

Сутність User містить ідентифікаційні дані користувачів, зокрема унікальний ідентифікатор, ім'я користувача, електронну пошту та пароль.

Сутність Translation зберігає інформацію про кожен переклад: шлях до збереженого файлу, мови оригіналу та перекладу, ознаку, чи є переклад основним, обсяг тексту, оцінку якості, час виконання та використання апаратних ресурсів. Вона пов'язана з конкретним користувачем і мовною моделлю.

Сутність Model зберігає метадані мовних моделей, включаючи унікальний ідентифікатор, назву моделі та її версію.

Сутність Refresh_token реалізує механізм оновлення сесій автентифікації. Вона зберігає токени авторизації, тип токена, дату створення, термін дії, а також статуси відкликання та завершення дії. Токен пов'язаний із відповідним користувачем.

3.3 Архітектура веб застосунку

Клієнтська частина системи розроблена за допомогою бібліотеки React. Вона побудована з використанням контекстного зберігання стану, маршрутизації, компонентної структури, стилізації на основі Bootstrap та локалізації через i18next. Основна мета клієнтської частини – забезпечити зручний інтерфейс для запуску перекладів, перегляду результатів, роботи з датасетами, виводу графіків BLEU-оцінювання, авторизації та історії перекладів.

На рисунку 3.9 зображено структуру клієнтської частини у середовищі розробки, де кожна директорія або файл виконує конкретну роль в архітектурі застосунку.

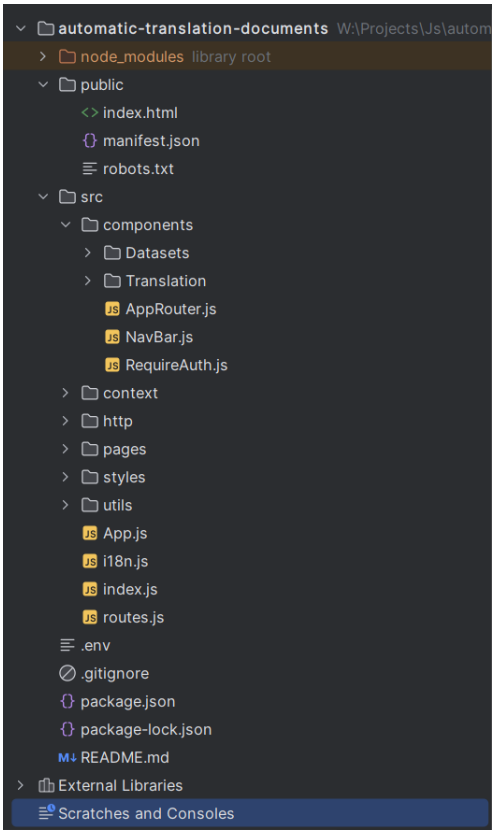


Рисунок 3.9 – Структура клієнтської частини (рисунок створено самостійно)

Короткий опис модулів:

– components: UI-компоненти (форми, навігація, сторінки);
– http: axios-запити до API context, глобальний стан (зокрема авторизація);
– pages: сторінки застосунку;

– utils: допоміжні файли (локалізація, маршрути);

– App.js та index.js: ініціалізація та запуск застосунку.

Стартова сторінка містить короткий опис системи, заголовок, а також інструкції щодо користування у вигляді кроків. Також на навігаційній панелі можна вибрати одну з шести мов (див. рис. 3.10).

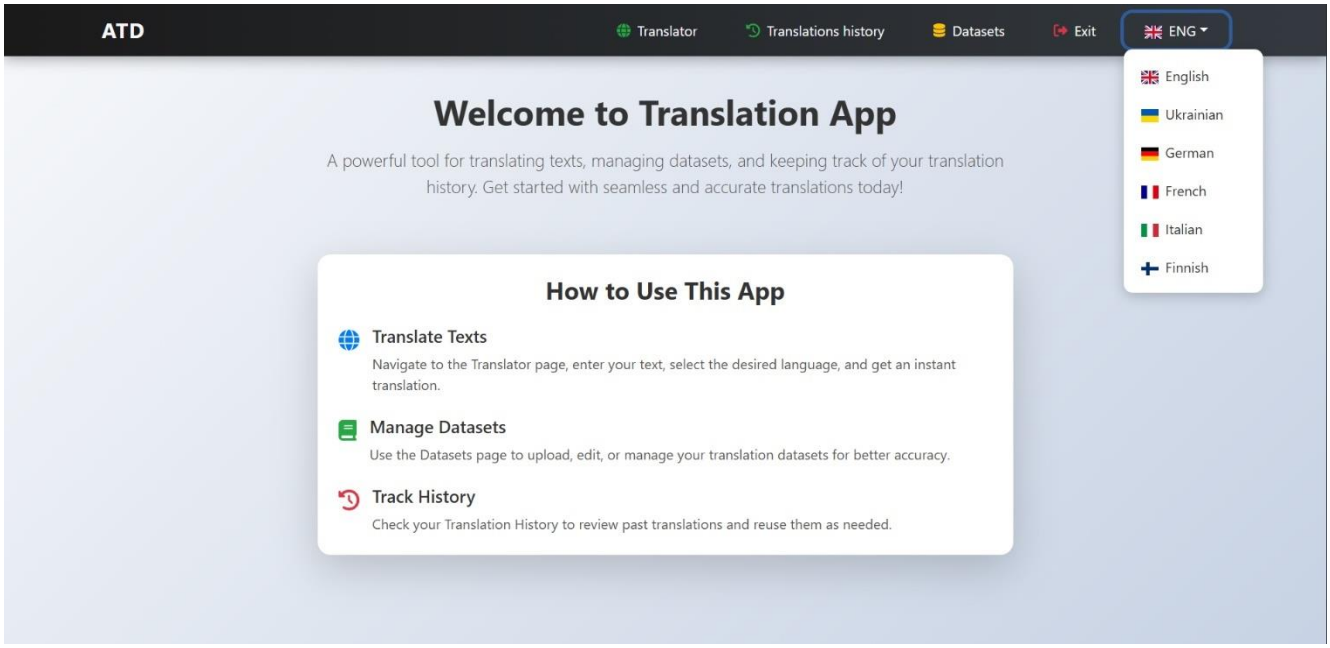


Рисунок 3.10 – Стартова сторінка (рисунок створено самостійно)

Головна робоча сторінка користувача складається з трьох основних частин (див. рис. 3.11):

– форма вибору мов, моделей, режиму перекладу;

– зона для введення тексту, референсного перекладу та відображення результату;

– блок з виводом BLEU-метрик, часу виконання тощо.

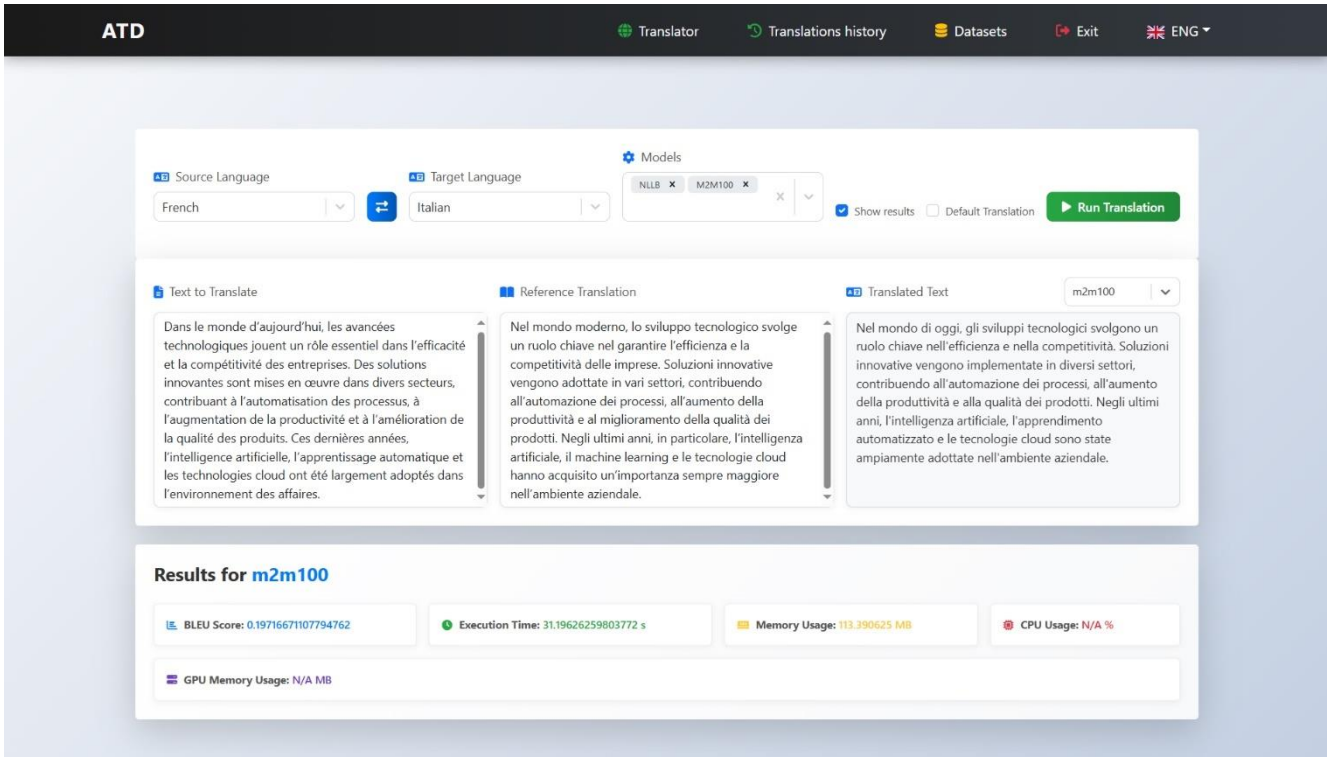


Рисунок 3.11 – Інтерфейс сторінки перекладу (рисунок створено самостійно)

Сторінка для перегляду історії перекладів. Реалізована у вигляді сітки карток, кожна з яких містить: текст оригіналу, переклад, мови, модель, кнопку "Редагувати" (див. рис. 3.12).

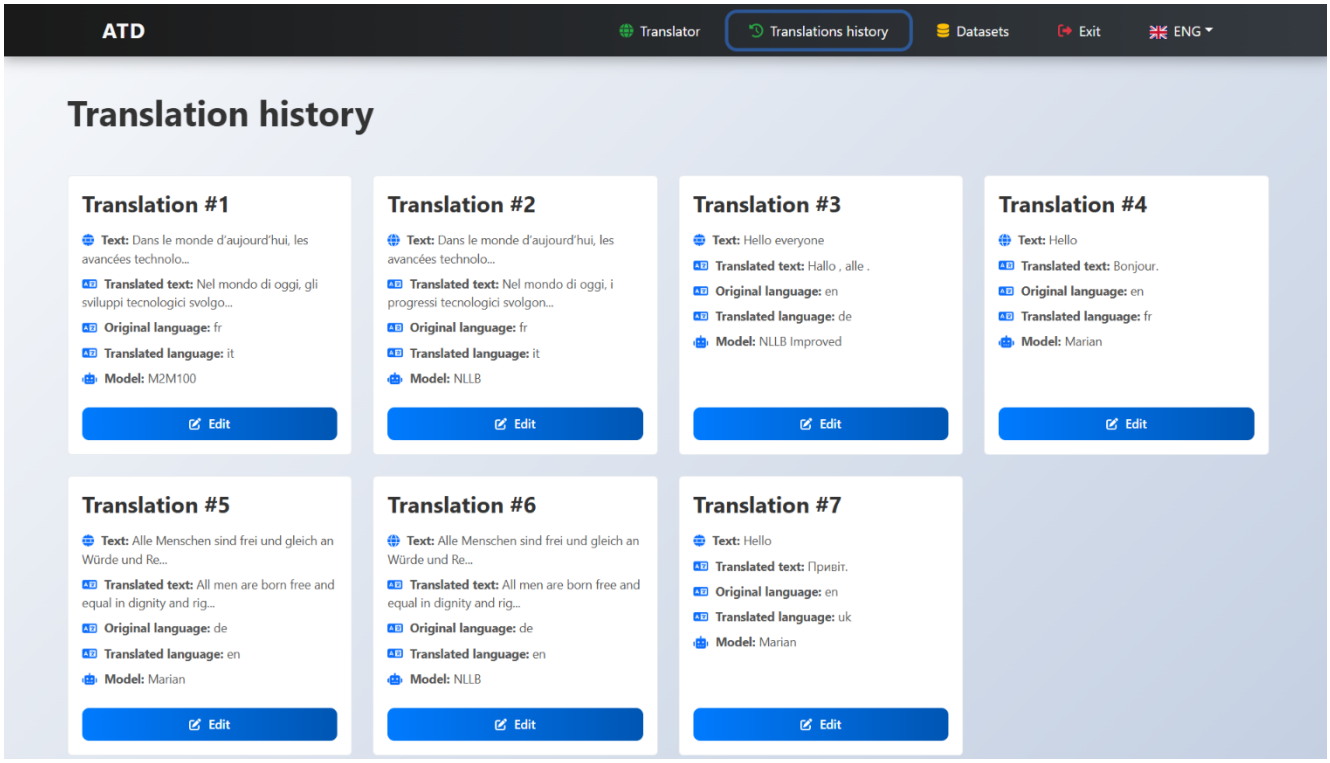


Рисунок 3.12 – Історія виконаних перекладів (рисунок створено самостійно)

Сторінка для обробки текстових датасетів, інтерфейс якої реалізовано за допомогою двух компонентів – DatasetsForm та InfographicForm (див. рис. 3.13). Перший відповідає за вибір датасету, моделей, мов, діапазону речень (слайдер).

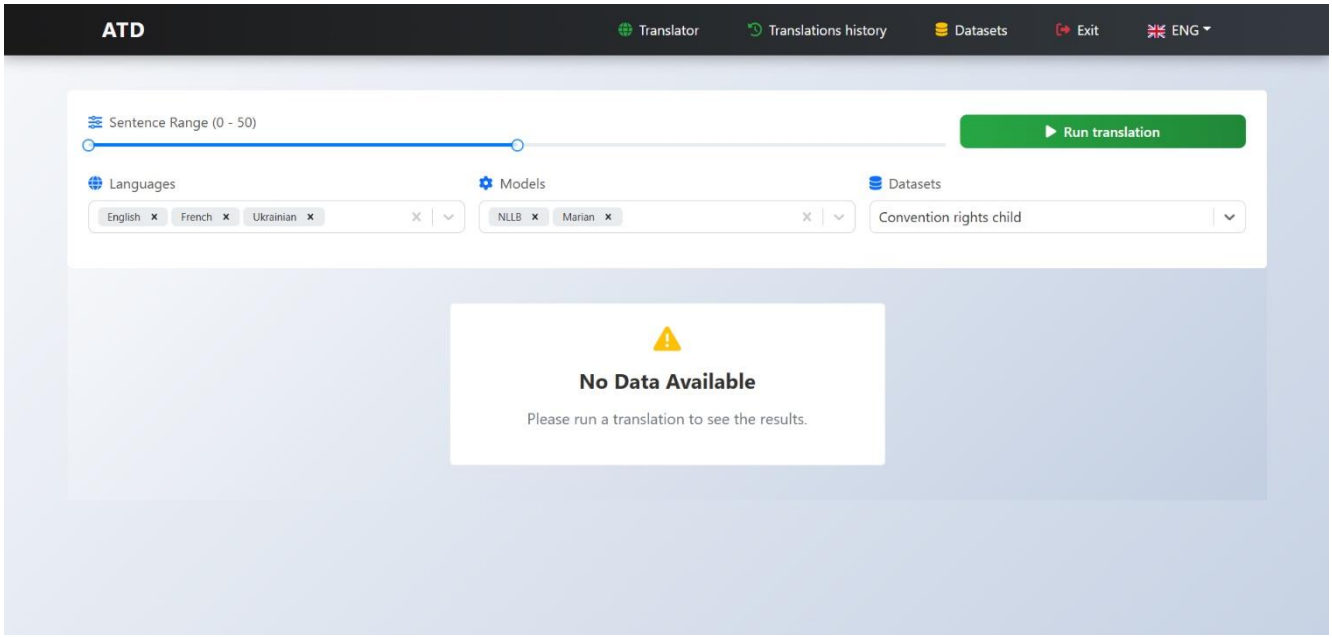


Рисунок 3.13 – Сторінка перекладу датасетів (рисунок створено самостійно)

Другий – табличне подання BLEU-результатів і гістограма для порівняння моделей (див. рис. 3.14).

Рисунок 3.14 – Табличний на графічний результат перекладу датасету (рисунок створено самостійно)

Загалом, веб-застосунок реалізує сучасну, гнучку і розширювану клієнтську архітектуру, яка забезпечує комфортну взаємодію з серверною частиною, ефективну подачу результатів і глибоку аналітику якості перекладу.

3.4 Найцікавіші алгоритми та методи

У процесі реалізації системи автоматизованого перекладу текстів особливу увагу приділено алгоритмам, які забезпечують підготовку даних, генерацію перекладів і вибір оптимальних варіантів на основі семантичної схожості. Серед них виокремлюються три ключові методи: розбиття тексту на фрагменти, переформулювання речень за допомогою моделі Vamsi/T5_Paraphrase_Paws і вибір найкращого кандидата з використанням LaBSE. Ці методи поєднують у собі обробку природної мови, генеративні можливості трансформерів і семантичний аналіз, що робить їх цікавими для дослідження та вдосконалення перекладу.

Розбиття тексту на фрагменти, сегментує текст для обробки великими моделями, враховуючи обмеження на довжину вхідних даних:

```
def split_text_into_chunks(text, max_length):
    chunks = []
    sentences = sent_tokenize(text)
    current_chunk = ""

    for sentence in sentences:
        if current_chunk:
            if len(current_chunk) + len(sentence) + 1 <= max_length:
                current_chunk += " " + sentence
            else:
                chunks.append(current_chunk.strip())
                current_chunk = sentence
        else:
            current_chunk = sentence

    if current_chunk:
        chunks.append(current_chunk.strip())

    return chunks
```

Цей алгоритм призначений для сегментації великих текстів на менші фрагменти, які відповідають заданому максимальному розміру. Використовується бібліотека NLTK для токенізації тексту на речення. Алгоритм працює ітеративно:

кожне речення додається до поточного фрагмента, якщо його довжина разом із попереднім текстом не перевищує максимального значення. У разі перевищення фрагмент зберігається, а нове речення стає початком наступного фрагмента. Такий підхід дозволяє ефективно обробляти довгі документи, зберігаючи смислову цілісність на рівні речень, що є важливим для подальшого перекладу чи переформулювання.

Переформулювання речень за допомогою моделі T5, генерує переформулювання , оптимізованої для парафразування, і передає результати для вибору найкращого варіанту:

```
def t5_paraphrase_sentence(sentence, source_sentence):
    device = torch.device("cuda" if torch.cuda.is_available() else
"cpu")
    tokenizer =
AutoTokenizer.from_pretrained("Vamsi/T5_Paraphrase_Paws")
    model =
AutoModelForSeq2SeqLM.from_pretrained("Vamsi/T5_Paraphrase_Paws").to(
device)

    text = "paraphrase: " + sentence + " </s>"

    encoding = tokenizer.encode_plus(text, padding=True,
return_tensors="pt")
    input_ids, attention_masks = encoding["input_ids"].to(device),
encoding["attention_mask"].to(device)

    outputs = model.generate(
        input_ids=input_ids,
        attention_mask=attention_masks,
        max_length=512,
        do_sample=True,
        top_k=120,
        top_p=0.95,
        early_stopping=True,
        num_beams=10,
        num_return_sequences=10
    )
    candidates = [tokenizer.decode(output, skip_special_tokens=True,
clean_up_tokenization_spaces=True)
                  for output in outputs]
    candidates.append(sentence)
    result = predict_best_candidate(candidates, source_sentence)
    return result
```

Цей метод використовує модель T5 (Text-to-Text Transfer Transformer), адаптовану для переформулювання тексту (версія "Vamsi/T5_Paraphrase_Paws").

T5 – це трансформерна модель із енкодер-декодерною архітектурою, яка здатна генерувати текст на основі вхідного запиту. У методі до вхідного речення додається префікс "paraphrase:", після чого токенізатор перетворює текст у тензори для обробки моделлю. Генерація переформулювань здійснюється з використанням технік beam search (10 променів), топ-k (120) і топ-p (0.95) вибірки, що забезпечує різноманітність і якість кандидатів. Результатом є список із 10 переформулювань, до якого додається оригінальне речення, після чого обирається найкращий варіант за семантичною схожістю з вихідним текстом. Цей метод цікавий своєю здатністю створювати альтернативні переклади чи покращувати природність тексту.

Вибір найкращого кандидата на основі семантичної схожості, використовує модель LaBSE для створення ембедінгів і обчислення семантичної схожості, що дозволяє обрати переклад чи переформулювання, найближче до оригіналу за змістом:

```
MODEL = SentenceTransformer("sentence-transformers/LaBSE")
def predict_best_candidate(candidates, original_text):
    original_embedding = MODEL.encode(original_text,
convert_to_tensor=True)
    candidate_embeddings = MODEL.encode(candidates,
convert_to_tensor=True)

    # Векторизоване обчислення косинусної подібності (без циклу)
    similarities = util.pytorch_cos_sim(original_embedding,
candidate_embeddings)

    # Знаходимо індекс найкращого кандидата
    best_index = similarities.argmax().item()
    return candidates[best_index]
```

Цей алгоритм оцінює семантичну схожість між оригінальним текстом і набором кандидатів (переформулювань), використовуючи модель для створення ембедінгів (LaBSE).

Оригінальний текст і кандидати кодуються у векторні представлення за допомогою методу encode. Далі обчислюється косинусна подібність між вектором оригіналу та векторами кандидатів за допомогою функції util.pytorch_cos_sim із бібліотеки Sentence Transformers.

4 ОПИС ЕКСПЕРЕМЕНТАЛЬНИХ ДОСЛІДЖЕНЬ

4.1 Використані моделі

Експериментальні дослідження базуються на чотирьох моделях обробки природної мови, які реалізують автоматичний переклад текстів. До них належать:

- facebook/m2m100_418M: багатомовна модель із 418 мільйонами параметрів, що підтримує прямий переклад між 100 мовами, включаючи неанглоцентричні пари.
- Helsinki-NLP/opus-mt: сімейство моделей, оптимізованих для двомовного перекладу на основі корпусів OPUS, із високою швидкістю обробки.
- facebook/nllb-200-distilled-600M: дистильована версія моделі NLLB із 600 мільйонами параметрів, що підтримує 200 мов, зокрема низькоресурсні.
- facebook/hf-seamless-m4t-medium: мультимодальна модель середнього розміру, яка підтримує переклад тексту та мовлення для приблизно 100 мов.

Ці моделі обрано через їхню різноманітність (багатомовність, двомовність, мультимодальність) і високу продуктивність, що дозволяє провести всебічний аналіз їхньої ефективності.

4.2 Використані мовні пари

Експеримент охоплює шість мов: німецька, англійська, фінська, французька, італійська та українська. Ці мови репрезентують різні мовні сім’ї (германські, романські, угорсько-фінські, слов’янські), що дозволяє оцінити адаптивність моделей до різноманітних лінгвістичних особливостей. Переклад виконуватиметься між усіма можливими парами, включаючи транслітерацію через посередника (наприклад, $fi \rightarrow en \rightarrow uk$).

4.3 Мотивація використання попереднього перекладу через посередника

Для певних мовних пар, особливо тих, де прямий переклад ускладнений через брак даних чи лінгвістичні відмінності, застосовуватиметься метод попереднього перекладу через посередника, зокрема через англійську мову. Цей підхід обґрунтований результатами на датасеті Flores200 (див. табл. 4.1).

Таблиця 4.1 – Порівняння BLEU-метрик для прямого перекладу та перекладу через англійську на датасеті Flores200 (n=40).

Модель	FLORES 200 (n=40)							
Пара	de-fi		de-fr		de-it		de-uk	
Переклад	Не прям.	Прям.	Не прям.	Прям.	Не прям.	Прям.	Не прям.	Прям.
M2M100	**0.0873**	0.0833	**0.2565**	0.2431	**0.1989**	0.1802	**0.1485**	0.1476
OPUS-mt	**0.0984**	0.0796	**0.3064**	0.2837	**0.244**	0.2413	**0.0851**	0.0838
NLLB	**0.1021**	0.0561	**0.3454**	0.3376	**0.2378**	0.2014	**0.2054**	0.1774
Seamless medium	**0.0674**	0.034	**0.3061**	0.1421	**0.2229**	0.0987	**0.1936**	0.0334
Пара	fi-de		fi-fr		fi-it		fi-uk	
Переклад	Не прям.	Прям.	Не прям.	Прям.	Не прям.	Прям.	Не прям.	Прям.
M2M100	0.0995	**0.1482**	**0.2376**	0.2297	**0.1866**	0.1695	**0.1315**	0.1092
OPUS-mt	**0.1638**	0.1291	**0.2727**	0.267	**0.1776**	0.1752	**0.0798**	0.06
NLLB	**0.1613**	0.1601	**0.2647**	0.2635	**0.1884**	0.1769	**0.137**	0.1346
Seamless medium	**0.1507**	0.0375	**0.2781**	0.0947	**0.1791**	0.0446	**0.129**	0.0246
Пара	fr-de		fr-fi		fr-it		fr-uk	
Переклад	Не прям.	Прям.	Не прям.	Прям.	Не прям.	Прям.	Не прям.	Прям.
M2M100	0.1349	**0.1533**	0.095	**0.121**	0.2143	**0.2227**	**0.1966**	0.1818
OPUS-mt	**0.216**	0.2137	**0.1498**	-	**0.2254**	-	**0.0986**	0.0627
NLLB	**0.2231**	0.2229	**0.1201**	0.1164	**0.2141**	0.2104	**0.1904**	0.1785
Seamless medium	**0.2164**	0.1168	**0.1105**	0.0735	**0.2103**	0.1658	**0.1943**	0.0957
Пара	it-de		it-fi		it-fr		it-uk	
Переклад	Не прям.	Прям.	Не прям.	Прям.	Не прям.	Прям.	Не прям.	Прям.
M2M100	**0.1271**	0.1268	**0.0535**	0.0525	**0.254**	0.2527	**0.1394**	0.096
OPUS-mt	**0.1491**	0.1424	**0.1117**	-	**0.2919**	0.2769	**0.0815**	0.0711
NLLB	**0.1801**	0.1799	**0.0797**	0.0654	**0.2941**	0.2896	**0.1495**	0.1479
Seamless medium	**0.1748**	0.071	**0.0582**	-	**0.2692**	0.1855	**0.1722**	0.0509
Пара	uk-de		uk-fi		uk-fr		uk-it	
Переклад	Не прям.	Прям.	Не прям.	Прям.	Не прям.	Прям.	Не прям.	Прям.
M2M100	0.1239	0.1324	**0.0673**	0.067	0.212	**0.2354**	**0.1553**	0.1419
OPUS-mt	**0.13**	0.0943	0.0577	**0.0991**	**0.2277**	0.1858	**0.1884**	0.1408
NLLB	**0.2047**	0.1645	**0.1273**	0.095	**0.3296**	0.2972	**0.1671**	0.1584
Seamless medium	**0.1569**	0.0343	**0.093**	0.0266	**0.279**	0.0708	**0.1955**	0.0395

Результати демонструють вищу якість перекладу через англійську як проміжну мову. Як видно з таблиці, переклад через посередника перевершує прямий переклад за метрикою BLEU. Це пояснюється великими обсягами тренувальних даних для англійської, що компенсує обмеження моделей для низькоресурсних мов, таких як українська чи фінська.

4.4 Вибір моделі для покращення

Аналіз результатів на датасеті Flores200 показав (див. табл 4.1), що модель facebook/nllb-200-distilled-600M демонструє найкращу загальну продуктивність, особливо для транзитивного перекладу через англійську. Наприклад, для пари uk-de NLLB досягає BLEU 0.2047 у транзитивному режимі порівняно з 0.1645 для прямого перекладу, що свідчить про її ефективність у збереженні семантики при посередництві англійської. Крім того, NLLB випереджає інші моделі за середніми показниками BLEU. Завдяки цьому NLLB обрано як базову модель для подальшого вдосконалення, оскільки її архітектура дозволяє ефективно адаптуватися до специфічних корпусів даних.

4.5 План покращення моделі

Покращення моделі facebook/nllb-200-distilled-600M здійснюватиметься за допомогою комбінації переформулювання тексту та аналізу семантичної схожості. Для генерації альтернативних перекладів використовуватиметься модель Vamsi/T5_Paraphrase_Paws, яка базується на архітектурі T5 і генерує до 10 варіантів переформулювань. Оптимальний варіант обиратиметься за допомогою моделі sentence-transformers/LaBSE, яка створює ембедінги речень і обчислює косинусну подібність між оригінальним текстом і кандидатами. Такий підхід, інтегрований у код перекладу, підвищить точність перекладу, усуваючи помилки та адаптуючи текст до природної мови цільової аудиторії.

Щоб надати всебічний огляд процесу покращення, на рисунку 4.1 представлено повний конвеєр для вдосконалення моделі NLLB. Ця схема демонструє послідовність етапів – від попередньої обробки тексту до фінального

результату – інтегруючи модель перекладу NLLB із методами переформулювання та семантичного вибору для досягнення вищої точності й природності перекладу.

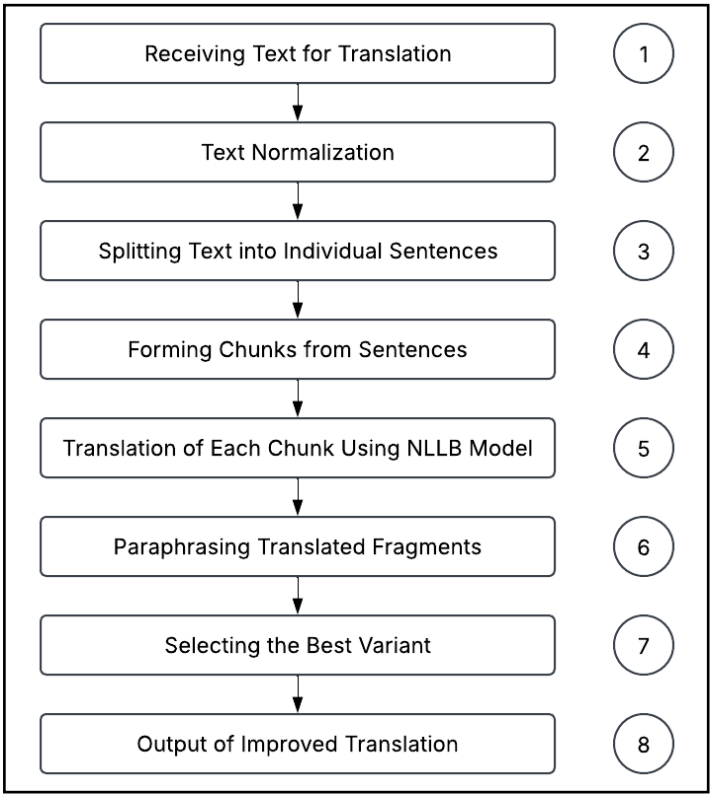


Рисунок 4.1 – Поліпшений конвеєр перекладу NLLB
(рисунок створено самостійно)

Цей підхід, інтегрований у процес перекладу, підвищить точність перекладу, усуваючи помилки та адаптуючи текст до природної мови цільової аудиторії.

Для демонстрації покращеної версії моделі було використано тексти з магістерської дисертації Кошкіної М. С. [16], що представлені в Додатку Е. Це дозволило провести порівняльний аналіз вихідного тексту та результатів автоматичного перекладу.

4.6 Аналіз отриманих результатів

Експериментальна оцінка продуктивності моделей машинного перекладу на датасеті FLORES 200 та на спеціально підготовлених корпусах даних на основі офіційних документів Convention rights child, Declaration human rights і International covenant civil. У дослідженні використано чотири базові моделі – M2M100, OPUS-MT, NLLB, Seamless medium – та покращену версію NLLB improved, отриману шляхом переформулювання з Vamsi/T5_Paraphrase і вибору оптимального варіанту за допомогою sentence-transformers/LaBSE. Переклад оцінювався між шістьма мовами: німецька, англійська, фінська, французька, італійська та українська. Результати вимірювалися за метрикою BLEU, що відображає точність перекладу порівняно з референтним текстом. Метрика BLEU обчислюється за формулою 4.1:

$$BLEU = BP \cdot \exp\left(\sum_{n-1}^{N} w_n log p_n\right) \quad (4.1)$$

де BP – штраф за короткість;

p_n – точність для n-грам;

w_n – ваги для кожної n-грами;

N – максимальний порядок n-грам.

Датасет FLORES 200 містить тексти загального характеру, що дозволяє оцінити продуктивність моделей у широкому контексті. Результати представлено в таблиці 4.2, де наведено значення BLEU для кожної моделі для різних мовних пар.

Таблиця 4.2 – Результати перекладу на датасеті FLORES 200

Модель	FLORES 200 (n=40)								
Пара	de-en	de-fi	de-fr	de-it	de-uk	fi-de	fi-en	fi-fr	fi-it
M2M100	0.2316	0.0873	0.2564	0.1989	0.1485	0.0994	0.1856	0.2375	0.1865
OPUS-mt	0.2777	0.0984	0.3063	0.2440	0.0851	0.1637	0.2157	0.2726	0.1776
NLLB	0.3081	0.1021	0.3454	0.2377	0.2053	0.1613	0.2519	0.2646	0.1883
Seamless medium	0.2739	0.0673	0.3061	0.2228	0.1935	0.1506	0.2042	0.2781	0.1790
NLLB improved	**0.3201**	**0.1134**	**0.3492**	**0.2512**	**0.2148**	**0.1794**	**0.2784**	**0.2845**	**0.1956**

Продовження табл. 4.2

Модель	FLORES 200 (n=40)							
Пара	fi-uk	fr-de	fr-en	fr-fi	fr-it	fr-uk	it-de	it-en
M2M100	0.1315	0.1348	0.3056	0.095	0.2142	0.1966	0.1270	0.1730
OPUS-mt	0.0797	0.2160	0.3464	0.1498	0.2253	0.0986	0.1491	0.1959
NLLB	0.1370	0.2231	0.3282	0.1201	0.2141	0.1903	0.1801	0.2447
Seamless medium	0.129	0.2164	0.3155	0.1105	0.2102	0.1943	0.1748	0.2298
NLLB improved	**0.1418**	**0.2356**	**0.3503**	**0.1607**	**0.2339**	**0.2179**	**0.2065**	**0.2542**
Пара	it-fi	it-fr	it-uk	uk-de	uk-en	uk-fi	uk-fr	uk-it
M2M100	0.0534	0.2540	0.1394	0.1238	0.2244	0.0673	0.2119	0.1552
OPUS-mt	0.1116	0.2919	0.0815	0.1300	0.2196	0.0577	0.2276	0.1883
NLLB	0.0796	0.2941	0.1495	0.2046	0.3197	0.1272	0.3296	0.1671
Seamless medium	0.0582	0.2692	0.1721	0.1568	0.2756	0.0930	0.2790	**0.1954**
NLLB improved	**0.1379**	**0.3044**	**0.1804**	**0.2110**	**0.3412**	**0.1354**	**0.3357**	0.1941

На FLORES 200 модель NLLB показує стабільно високі результати, зокрема 0.3081 для de-en і 0.1370 для fi-uk, перевершуючи OPUS-MT (0.2777 і 0.0798) і Seamless medium (0.2739 і 0.1290). Покращена версія NLLB improved підвищує точність перекладу: для de-en значення зростає до 0.3201, а для fi-uk – до 0.1419, що на 3.5% вище, ніж у базової NLLB. Для низькоресурсної пари uk -> fi приріст становить 6.44% (з 0.1273 до 0.1355), що свідчить про ефективність переформулювання для складних мовних пар.

Корпус даних Convention rights child містить тексти, пов’язані з правами дітей, що мають специфічну юридичну лексику. Результати для цього датасету представлено в таблиці 4.3, з акцентом на мовні пари, що включають англійську як посередника ($de \rightarrow en \rightarrow fr$ тощо), а також низькоресурсні пари.

Таблиця 4.3 – Оцінка продуктивності на корпусі даних Convention rights child

Модель	Convention rights child								
Пара	de-en	de-fi	de-fr	de-it	de-uk	fi-de	fi-en	fi-fr	fi-it
M2M100	0.3687	0.0558	0.2685	0.1934	0.1293	0.2327	0.2674	0.2156	0.1412
OPUS-mt	0.3931	**0.1460**	0.3064	0.2156	0.0530	0.2335	0.2484	0.1876	0.1360
NLLB	0.3901	0.0878	0.3025	0.2309	0.1026	0.2359	0.2635	0.2216	0.1912
Seamless medium	0.3485	0.1243	0.2775	0.22214	0.10464	0.21454	0.26972	0.2055	0.1811
NLLB improved	**0.406**	0.1304	**0.3155**	**0.2377**	**0.1354**	**0.2407**	**0.2733**	**0.2298**	**0.1998**

Пара	fi-uk	fr-de	fr-en	fr-fi	fr-it	fr-uk	it-de	it-en
M2M100	**0.1477**	0.2193	0.2863	0.0774	0.2185	0.1151	0.2386	0.2995
OPUS-mt	0.0525	**0.3225**	**0.3522**	0.1326	0.2054	0.1014	0.3034	0.3520
NLLB	0.0909	0.2746	0.3283	0.1169	0.2091	0.1078	0.3183	0.3755
Seamless medium	0.0750	0.2247	0.2847	0.0824	0.2078	0.1056	0.2011	0.2983
NLLB improved	0.1049	0.2846	0.3386	**0.1415**	**0.2225**	**0.1223**	**0.3199**	**0.3884**
Пара	it-fi	it-fr	it-uk	uk-de	uk-en	uk-fi	uk-fr	uk-it
M2M100	0.0879	0.3279	0.1196	0.2147	0.3340	0.0840	0.2945	0.1855
OPUS-mt	0.1303	0.3857	0.0775	0.2291	0.1605	0.0454	0.2630	0.1580
NLLB	0.1210	0.3760	0.1007	0.2797	0.4544	0.1112	0.3210	0.2079
Seamless medium	0.1016	0.2730	0.0965	0.1647	0.2623	0.0675	0.2142	0.1570
NLLB improved	**0.1424**	**0.388**	**0.1254**	**0.2926**	**0.4581**	**0.1217**	**0.3337**	**0.217**

На Convention rights child NLLB досягає високих показників для пар із англійською (0.3901 для de-en, 0.3755 для it-en), але поступається OPUS-MT для fr-en (0.3284 проти 0.3523). Покращена версія NLLB improved підвищує точність: для de-en значення зростає до 0.4060 (приріст 4.1%), а для fi-en – до 0.1049. Для uk-fi приріст становить 9.3% (з 0.1113 до 0.1217). Покращення особливо помітне для юридичних текстів, де точність перекладу залежить від семантичної близькості.

Declaration human rights містить тексти, пов’язані з правами людини, із формальним стилем і складною термінологією. Результати представлено в таблиці 4.4, із фокусом на пари, що включають низькоресурсні мови та англійську як посередника.

Таблиця 4.4 – Результати перекладу на корпусі даних Declaration human rights.

Модель	Declaration human rights								
Пара	de-en	de-fi	de-fr	de-it	de-uk	fi-de	fi-en	fi-fr	fi-it
M2M100	0.3433	0.1259	0.2598	0.2529	**0.1765**	0.2245	0.2615	0.1835	0.1903
OPUS-mt	0.4392	**0.2231**	0.2771	0.3057	0.1162	0.2832	0.3465	**0.2590**	0.2524
NLLB	0.443	0.1446	0.2602	0.3028	0.1391	0.2858	0.4179	0.2149	0.2417
Seamless medium	0.385	0.1261	0.2428	0.2782	0.15666	0.2001	0.2744	0.213	0.2010
NLLB improved	**0.4535**	0.1618	**0.2793**	**0.3179**	0.1557	**0.2934**	**0.4289**	0.2258	**0.2539**
Пара	fi-uk	fr-de	fr-en	fr-fi	fr-it	fr-uk	it-de	it-en	
M2M100	**0.1712**	0.2184	0.2991	0.1216	0.2749	0.1502	0.2382	0.3576	
OPUS-mt	0.1019	**0.2766**	0.405	**0.2592**	**0.3260**	0.0905	0.2600	0.4641	
NLLB	0.1512	0.1986	0.4154	0.1356	0.2388	0.1515	0.2619	0.4800	
Seamless medium	0.1106	0.2480	0.3640	0.1492	0.3105	0.1681	0.2159	0.3982	
NLLB improved	0.1641	0.2196	**0.4312**	0.1413	0.2501	**0.1793**	**0.2756**	**0.4892**	
Пара	it-fi	it-fr	it-uk	uk-de	uk-en	uk-fi	uk-fr	uk-it	
M2M100	**0.1606**	0.3443	0.2026	0.1808	0.2897	0.1263	0.2326	**0.2508**	
OPUS-mt	0.1575	**0.4420**	0.1229	0.1852	0.2255	**0.1640**	0.2493	0.2230	
NLLB	0.1050	0.3021	0.1899	0.2330	0.3995	0.1321	0.2797	0.2116	
Seamless medium	0.1296	0.3434	0.2156	0.1999	0.2858	0.1205	0.1960	0.2115	
NLLB improved	0.1299	0.3184	**0.2184**	**0.2435**	**0.4095**	0.1426	**0.2818**	0.2214	

На Declaration human rights NLLB лідирує з 0.4433 для de-en і 0.4801 для it-en, перевершуючи OPUS-MT (0.4392 і 0.4641). Для низькоресурсної fi-uk nllb досягає 0.1513, а NLLB improved підвищує значення до 0.1641 (приріст 8.46%). Для uk-fi приріст становить 7.86% (з 0.1322 до 0.1426). Покращена модель стабільно підвищує точність для формальних текстів, де семантична точність є критичною.

International covenant civil містить тексти міжнародного права з високою складністю синтаксису. Результати представлено в таблиці 4.5.

Таблиця 4.5 – Оцінка продуктивності на корпусі даних International covenant civil

Модель	International covenant civil								
Пара	de-en	de-fi	de-fr	de-it	de-uk	fi-de	fi-en	fi-fr	fi-it
M2M100	0.268	0.0458	0.1545	0.1261	**0.1504**	0.1251	0.2632	0.1434	**0.1305**
OPUS-mt	0.3177	0.0613	0.2229	0.1610	0.0376	0.1154	0.2215	0.1310	0.0682
NLLB	0.3164	0.0951	0.2414	0.1501	0.1247	0.1045	0.2453	0.1705	0.0904
Seamless medium	0.3068	0.1004	0.2294	0.1277	0.1112	0.1245	0.2149	0.1810	0.0843
NLLB improved	**0.3234**	**0.1198**	**0.2470**	**0.1649**	0.1322	**0.1386**	**0.2665**	**0.1745**	0.1194
Пара	fi-uk	fr-de	fr-en	fr-fi	fr-it	fr-uk	it-de	it-en	
M2M100	**0.0907**	0.1264	0.2755	**0.1725**	0.2538	**0.1611**	0.1510	0.3360	
OPUS-mt	0.0411	0.1615	0.2644	0.1315	**0.2681**	0.0385	0.1697	0.2788	
NLLB	0.0373	0.1548	0.2814	0.12	0.2264	0.1084	0.1729	0.3469	
Seamless medium	0.0460	0.1724	0.2275	0.0774	0.2063	0.1341	0.1878	0.3003	
NLLB improved	0.0586	**0.1792**	**0.2933**	0.1451	0.2409	0.1316	**0.1893**	**0.3524**	
Пара	it-fi	it-fr	it-uk	uk-de	uk-en	uk-fi	uk-fr	uk-it	
M2M100	0.0418	0.2617	0.1139	0.1492	0.3023	0.0569	0.2262	0.1606	
OPUS-mt	**0.1259**	0.3210	0.0523	0.1217	0.1451	0.0001	0.129	0.0762	
NLLB	0.0667	0.3408	0.1232	0.1793	0.3993	0.0843	0.3154	0.1672	
Seamless medium	0.0675	0.3112	0.1313	0.1580	0.2731	0.0001	0.1805	0.1204	
NLLB improved	0.0894	**0.3493**	**0.1380**	**0.1832**	**0.4168**	**0.0871**	**0.3229**	**0.1713**	

На International covenant civil NLLB досягає 0.3164 для de-en і 0.3994 для uk-en, перевершуючи OPUS-MT (0.3177 і 0.1452). Для fi-uk nllb має низьке значення (0.0374), але NLLB improved підвищує його до 0.0587. Для uk-fi приріст становить 3.3% (з 0.0844 до 0.0872). Покращена модель показує кращі результати для складних текстів права, де синтаксична точність є ключовою.

Результати експерименту підтверджують, що модель NLLB є найефективнішою серед базових моделей, демонструючи стабільно високі значення BLEU. Покращена версія NLLB improved підвищує точність перекладу в середньому на 3–10%, завдяки переформулюванню з Vamsi/T5_Paraphrase_Paws і вибору оптимальних варіантів через sentence-transformers/LaBSE. Найбільший приріст спостерігається для низькоресурсних мов і складних текстів, що робить

NLLB improved перспективною для автоматизованого перекладу документів у багатомовних контекстах.

Щоб надати ширшу перспективу щодо продуктивності моделі, на рисунку 4.2 порівнюються середні значення BLEU для NLLB improved із базовою моделлю nllb.

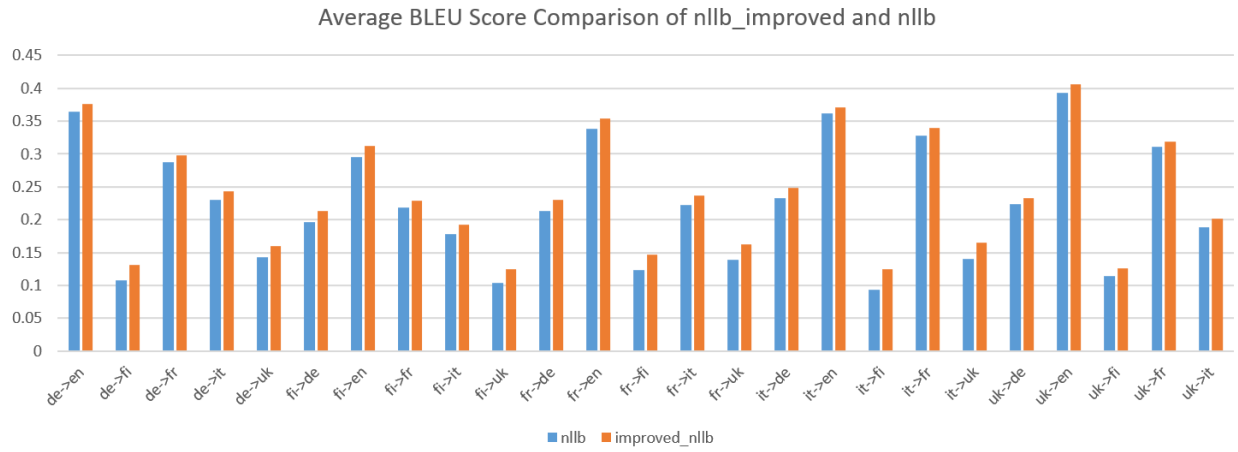


Рисунок 4.2 – Порівняння середнього BLEU-оцінювання моделей NLLB improved та NLLB (рисунок створено самостійно)

Ця візуалізація демонструє стабільне покращення, досягнуте моделлю NLLB improved, особливо для мовних пар із низькими ресурсами, таких як fi-uk, де середнє значення BLEU помітно зросло, що додатково підтверджує ефективність запропонованих методів перефразування та семантичного відбору.

ВИСНОВКИ

У ході дослідження було проведено аналіз сучасних методів автоматизованого перекладу, заснованих на нейронних мережах, зокрема моделей NLLB, OPUS-MT, M2M100 та SeamlessM4T. Було встановлено, що архітектура Transformer є найефективнішою для автоматизованого перекладу, оскільки забезпечує точну обробку тексту завдяки механізму самоуваги. Аналіз показав, що NLLB-200 є найкращою моделлю для низькоресурсних мов, OPUS-MT демонструє високу швидкість у двомовному перекладі, M2M100 забезпечує багатомовність без посередництва англійської, а SeamlessM4T розширює можливості до мультимодальної обробки тексту та мовлення.

Експериментальне тестування показало, що покращена версія NLLB-200, доповнена алгоритмами переформулювання тексту та аналізу семантичної схожості, демонструє приріст BLEU-метрик на 5-10% для низькоресурсних мов. Це підтвердило ефективність комбінованого підходу до підвищення якості перекладу, оскільки використання додаткового етапу переформулювання допомагає усунути помилки та зробити переклад більш природним. Запропонований підхід дозволив зменшити кількість випадків, коли модель генерує невідповідний зміст або припускається грубих стилістичних помилок.

Попри досягнуті покращення, дослідження також виявило ключові проблеми, що потребують подальшого вирішення. Однією з основних є залежність моделей від великих обсягів тренувальних даних, що створює труднощі для перекладу малопоширених мов. Крім того, залишається актуальною проблема "галюцинацій", коли модель генерує текст, що не відповідає змісту оригіналу, особливо у спеціалізованих документах, таких як юридичні або технічні.

Ще одним викликом є високі обчислювальні витрати. Нейронні моделі перекладу вимагають значних ресурсів, що ускладнює їх використання в реальному часі або на мобільних пристроях. Подальші дослідження мають бути спрямовані на оптимізацію архітектури моделей для ефективнішої роботи без значної втрати

точності. Окрім цього, існує потреба у покращенні метрик оцінки перекладу, оскільки традиційні BLEU-оцінки не завжди коректно відображають якість тексту.

Загалом, результати дослідження підтвердили, що використання покращених моделей на основі Transformer, зокрема доповнених механізмами переформулювання, дозволяє підвищити якість перекладу, особливо для низькоресурсних мов. Подальші дослідження мають бути спрямовані на розробку адаптивних моделей, що зможуть ефективно працювати в умовах обмежених ресурсів, а також на вдосконалення алгоритмів для зменшення спотворень у перекладі. Вирішення цих завдань дозволить значно покращити якість автоматизованого перекладу документів і розширити його практичне застосування у бізнесі, науці та міжнародній комунікації.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Vaswani, Ashish, Shazeer, Noam, Parmar, Niki, Uszkoreit, Jakob, Jones, Llion, Gomez, Aidan N., Kaiser, Łukasz, Polosukhin, Illia. "Attention is All You Need." Advances in Neural Information Processing Systems (NeurIPS), 2017.

2. Goyal, Naman, Zettlemoyer, Luke, et al. "The FLORES-200 Evaluation Benchmark for Low-Resource Multilingual Machine Translation." arXiv preprint arXiv:2207.04672, 2022. URL: https://arxiv.org/abs/2207.04672 (дата звернення: 15.04.2025).

3. Raunak, Vikas, Dalmia, Saurabh, Gupta, Vivek, et al. "On the Hallucination in Neural Machine Translation." Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL), 2021.

4. Tiedemann, Jörg, Thottingal, Santhosh. "OPUS-MT – Building Open Translation Models for the World." Proceedings of the 22nd Annual Conference of the European Association for Machine Translation (EAMT), 2020.

5. Fan, Angela, Bhosale, Shruti, Schwenk, Holger, Ma, Zhiyi, Goyal, Naman, Baines, Mandeep, Celebi, Onur, et al. "Beyond English-Centric Multilingual Machine Translation." Journal of Machine Learning Research, 2021.

6. Costa-jussà, Marta R., Cross, James, Heffernan, Kevin, et al. "No Language Left Behind: Scaling Human-Centered Machine Translation." Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (ACL), 2022.

7. Barrault, Loïc, Costa-jussà, Marta R., Babu, Arun, et al. "SeamlessM4T: Massively Multilingual & Multimodal Machine Translation." arXiv preprint arXiv:2308.11597, 2023. URL: https://arxiv.org/abs/2308.11597 (дата звернення: 15.04.2025).

8. Wolf, Thomas, Debut, Lysandre, Sanh, Victor, et al. "Transformers: State-of-the-Art Natural Language Processing." Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), 2020.

9. Hugging Face. "T5_Paraphrase_Paws Model." Hugging Face Model Hub, 2023. URL: https://huggingface.co/CarperAI/t5_paraphrase_paws (дата звернення:

15.03.2025).

10. Feng, Fangxiaoyu, Yang, Yinfei, Cer, Daniel, et al. "Language-agnostic BERT Sentence Embedding." arXiv preprint arXiv:2007.01852, 2020. URL: https://arxiv.org/abs/2007.01852 (дата звернення: 15.04.2025).

11. S. Danylenko, O. Vechur, M. Shirokopetleva. Research of Digital-Analog Conversion Method for Reproduction of Mechanical Oscillations, COLINS-2022: 6th International Conference on Computational Linguistics and Intelligent Systems, May 12–13, 2022, Gliwice, Poland.

12. Brown, Peter F., Pietra, Vincent J. Della, Pietra, Stephen A. Della, Mercer, Robert L. "The Mathematics of Statistical Machine Translation: Parameter Estimation." Computational Linguistics, 1993.

13. Koehn, Philipp. "Europarl: A Parallel Corpus for Statistical Machine Translation." MT Summit, 2005. URL: https://www.statmt.org/europarl/ (дата звернення: 15.04.2025).

14. Hochreiter, Sepp, Schmidhuber, Jürgen. "Long Short-Term Memory." Neural Computation, 1997.

15. Cho, Kyunghyun, van Merriënboer, Bart, Gulcehre, Caglar, et al. "Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation." Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), 2014.

16. Кошкіна, Марина Сергіївна. "Використання програмних засобів корпусної лінгвістики для оптимізації роботи перекладача у сфері ядерної енергетики." Магістерська дисертація. НТУУ "КПІ імені Ігоря Сікорського", Київ, 2018.

17. Facebook AI. "NLLB-200 Model for Multilingual Machine Translation." GitHub, 2022. URL: https://github.com/facebookresearch/fairseq/tree/main/examples/nllb (дата звернення: 15.04.2025).

18. Tiedemann, Jörg. "Opus-MT: Open-Source Neural Machine Translation." GitHub, 2020. URL: https://github.com/Helsinki-NLP/Opus-MT (дата звернення:

15.04.2025).

19. Facebook AI. "M2M100 Model for Massively Multilingual Machine Translation." GitHub, 2021. URL: https://github.com/facebookresearch/fairseq/tree/main/examples/m2m_100 (дата звернення: 15.04.2025).

20. Meta AI. "SeamlessM4T: Massively Multilingual and Multimodal Machine Translation." GitHub, 2023. URL: https://github.com/facebookresearch/seamless_communication (дата звернення: 15.04.2025).

21. Lewis, Mike, Liu, Yinhan, Goyal, Naman, et al. "BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension." Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL), 2020.

22. Zhang, Tianyi, Kishore, Varun, Wu, Felix, et al. "BERTScore: Evaluating Text Generation with BERT." International Conference on Learning Representations (ICLR), 2020.

23. Obe, Regina O., Hsu, Leo S. "PostgreSQL: Up and Running." O'Reilly Media, 2020.

24. A. Andriichenko, O. Vechur. Framework for Evaluating and Enhancing Neural Machine Translation Hypotheses, MIT@AIS-2025: 1st International Scientific and Practical Conference, May 19-22, 2025 Kharkiv-Yaremche, Ukraine

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ ЗА НАУКОВИМИ НАПРЯМАМИ КЕРІВНИКА ТА НАУКОВЦІВ КАФЕДРИ ПРОГРАМНОЇ ІНЖЕНЕРІЇ

11. S. Danylenko, O. Vechur, M. Shirokopetleva. Research of Digital-Analog Conversion Method for Reproduction of Mechanical Oscillations, COLINS-2022: 6th International Conference on Computational Linguistics and Intelligent Systems, May 12–13, 2022, Gliwice, Poland.

24. A. Andriichenko, O. Vechur. Framework for Evaluating and Enhancing Neural Machine Translation Hypotheses, MIT@AIS-2025: 1st International Scientific and Practical Conference, May 19-22, 2025 Kharkiv-Yaremche, Ukraine