Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Інформаційно-аналітичних технологій та менеджменту
(повна назва)
Кафедра Інформатики
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

ДОСЛІДЖЕННЯ ТА РЕАЛІЗАЦІЯ МЕТОДУ ПЕРЕКЛАДУ ТЕКСТУ НА ЗОБРАЖЕННЯХ ЗІ ЗБЕРЕЖЕННЯМ ОРИГІНАЛЬНОГО ВИГЛЯДУ ЗОБРАЖЕННЯ
(тема)

Виконав:
здобувач 2 року навчання,
групи ІНФм-24-2
Харченко В.В
(прізвище, ініціали)

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

Тип програми освітньо-професійна

Освітня програма Інформатика
(повна назва освітньої програми)

Науковий керівник доц. Любченко В. А.
(посада, прізвище, ініціали)

Допускається до захисту

Завідувач кафедри інформатики _____ Кобилін О. А.
(підпис) (прізвище, ініціали)

2025 р.

Харківський національний університет радіоелектроніки

Факультет Інформаційно-аналітичних технологій та менеджменту

Кафедра Інформатики

Рівень вищої освіти другий (магістерський)

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

Тип програми освітньо-професійна

Освітня програма Інформатика
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри

_____
(підпис)
«____»_____ 2025 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

здобувачеві Харченку Владиславу Вадимовичу
(прізвище, ім'я, по батькові)

1. Тема роботи Дослідження та реалізація методу перекладу тексту на зображеннях зі збереженням оригінального вигляду зображення

затверджена наказом університету від 14 листопада 2025 року № 1045Ст
2. Термін подання здобувачем роботи до екзаменаційної комісії 22 листопада 2025 р.
3. Вихідні дані до роботи методи детекції та розпізнавання тексту на зображеннях, методи машиного перекладу, відновлення зображення та вставка тексту, літературні джерела щодо OCR-систем, бібліотеки для реалізації компонентів системи, інтерфейс користувача, датасети зображень з текстом для тестування моделей

4. Перелік питань, що потрібно опрацювати в роботі
1. Аналіз сучасних методів детекції та розпізнавання тексту на зображеннях
2. Аналіз методів машиного перекладу та візуального відстворення тексту на зображеннях.
3. Аналіз літературних джерел щодо апробації OCR-систем та методів перекладу.
4. Формування алгоритму обробки зображень для перекладу тексту зі збереженням стилю.
5. Розробка програмного застосунку для автоматичного перекладу на зображеннях з можливістю вибору OCR-моделі, сервісу перекладу, та налаштувань.

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри) актуальність проблеми перекладу тексту на зображеннях, об'єкт та мета дослідження, постановка задачі, блок-схеми алгоритмів детекції тексту, розпізнавання символів та візуального відтворення перекладу, приклади зображень з датасетів, ілюстрація головного екрану розробленого застосунку із веб-інтерфейсом Gradio та параметрами налаштування OCR, перекладу та вставки, порівняльні таблиці та графіки точності OCR-систем та якості перекладу, приклади результатівроботи системи з перекладеним текстом на різних типах зображення, висновки, перспективи розвитку та апробація роботи

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

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

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Строк / терміни виконання етапів роботи	Примітка
1	Отримання завдання на кваліфікаційну роботу	29.09.2025	
2	Аналіз завдання, підбір літератури	30.09.25-07.10.25	
3	Аналіз літератури з досліджуваної проблеми	08.10.25-14.10.25	
4	Аналіз існуючих OCR та методів перекладу	15.10.25-20.10.25	
5	Дослідження методу перекладу зображення	21.10.25-27.10.25	
6	Програмна реалізація	28.10.25-05.11.25	
7	Обґрунтування отриманих результатів	06.11.25-11.11.25	
8	Оформлення пояснювальної записки	12.11.25-14.11.25	
9	Перевірка на нормоконтроль	20.11.25	
10	Перевірка на плагіат	21.11.15	
11	Рецензування	22.11.25	
12	Підготовка презентації та доповіді	23.11.25	
13	Занесення роботи в електронний архів	24.11.25	
14	Попередній захист кваліфікаційної роботи	01.12.25	

Дата видачі завдання 29 вересня 2025 р.

Здобувач _____
(підпис)

Керівник роботи _____ доц. Любченко В. А.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка до кваліфікаційної роботи: 92 с., 12 табл., 27 рис., 1 дод., 68 джерел.

ВІЗУАЛЬНА РЕКОНСТРУКЦІЯ, ДЕТЕКЦІЯ ТЕКСТУ, ЗБЕРЕЖЕННЯ СТИЛЮ ЗОБРАЖЕННЯ, МАШИННИЙ ПЕРЕКЛАД, ПЕРЕКЛАД ТЕКСТУ НА ЗОБРАЖЕННЯХ, РЕНДЕРИНГ ТЕКСТУ, РОЗПІЗНАВАННЯ ТЕКСТУ, CRAFT, CRNN, EASYOCR, IMAGE INPAINTING, PADDLEOCR, SURYA.

Об’єктом дослідження є процес автоматичного перекладу текстової інформації на зображеннях із збереженням оригінального стилю.

Предмет дослідження – методи детекції, розпізнавання, машинного перекладу та візуального відтворення тексту з адаптивними параметрами.

Метою дослідження є створення модульної системи автоматичного перекладу тексту на зображеннях.

Проведено аналіз підходів до детекції тексту, OCR, машинного перекладу та візуальної реконструкції. Виконано експериментальне порівняння чотирьох OCR-систем на зображеннях із трьох датасетів текстових фрагментів.

Наукова новизна полягає у систематичному порівнянні OCR і перекладачів для різних мов з визначенням точності й швидкості, у запропонованих адаптивних параметрах для роботи з різнотипними зображеннями та у створенні модульної архітектури з можливістю локального виконання.

У результаті систематизовано теоретичні основи задачі, визначено переваги та обмеження методів, створено працюючу систему та сформовано рекомендації щодо вибору оптимальних компонентів залежно від вимог до точності, швидкості.

ABSTRACT

Explanatory note to the qualification work: 92 pages, 12 tables, 27 figures, 1 appendix, 68 sources.

CRAFT, CRNN, EASYOCR, IMAGE INPAINTING, IMAGE STYLE PRESERVATION, MACHINE TRANSLATION, PADDLEOCR, SURYA, TEXT DETECTION, TEXT RENDERING, TEXT RECOGNITION, TEXT TRANSLATION ON IMAGES, VISUAL RECONSTRUCTION.

The object of the study is the process of automatic translation of textual information in images while preserving the original visual style. The subject of the study is the methods of text detection, recognition, machine translation, and visual reconstruction with adaptive processing parameters. The aim of the study is to develop a modular system for automatic translation of text in images.

An analysis of approaches to text detection, OCR, machine translation, and visual reconstruction has been carried out. An experimental comparison of four OCR systems on images from three datasets on text fragments has been performed.

The scientific novelty lies in the systematic comparison of OCR systems and translation tools for different languages with evaluation of accuracy and speed, in the proposed adaptive parameters for handling diverse image types, and in the development of a modular architecture capable of local execution.

As a result of the research, the theoretical foundations of the task have been systematized, the advantages and limitations of existing methods have been identified, a functional system has been created, and recommendations have been formulated regarding the choice of optimal components depending on accuracy and processing speed requirements.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

ШІ – штучний інтелект

API – Application Programming Interface (програмний інтерфейс додатка)

ASTER – Attentional Scene Text Recognizer (розпізнавач сценового тексту з механізмом уваги)

BLEU – Bilingual Evaluation Understudy (метрика оцінки якості перекладу)

CER – Character Error Rate (рівень помилок на рівні символів)

CLAHE – Contrast Limited Adaptive Histogram Equalization (адаптивне вирівнювання гістограми з обмеженням контрасту)

CNN – Convolutional Neural Networks (згорткові нейронні мережі)

COMET – Crosslingual Optimized Metric for Evaluation of Translation (крос-лінгвальна метрика оцінки перекладу)

CPU – Central Processing Unit (центральний процесор)

CRAFT – Character Region Awareness for Text Detection (детекція тексту з аналізом символьних регіонів)

CRNN – Convolutional Recurrent Neural Network (згорткова рекурентна нейронна мережа)

CTC – Connectionist Temporal Classification (коннекціоністська темпоральна класифікація)

DB – Differentiable Binarization (диференційована бінаризація)

EAST – Efficient and Accurate Scene Text Detector (ефективний детектор сценового тексту)

FPS – Frames Per Second (кадрів за секунду)

GAN – Generative Adversarial Network (генеративно-змагальна мережа)

GPU – Graphics Processing Unit (графічний процесор)

JSON – JavaScript Object Notation (текстовий формат обміну даними)

LLM – Large Language Model (велика мовна модель)

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

MSER – Maximally Stable Extremal Regions (максимально стабільні екстремальні регіони)

NMT – Neural Machine Translation (нейронний машинний переклад)

OCR – Optical Character Recognition (оптичне розпізнавання символів)

PSNR – Peak Signal-to-Noise Ratio (пікове співвідношення сигнал/шум)

ResNet – Residual Network (залишкова нейронна мережа)

SDK – Software Development Kit (набір засобів розробки програмного забезпечення)

SSIM – Structural Similarity Index Measure (індекс структурної подібності)

STN – Spatial Transformer Network (мережа просторової трансформації)

SWT – Stroke Width Transform (перетворення товщини штриху)

TrOCR – Transformer-based Optical Character Recognition (OCR на основі трансформерів)

UI – User Interface (інтерфейс користувача)

VGG – Visual Geometry Group (архітектура нейронної мережі)

WER – Word Error Rate (рівень помилок на рівні слів)

ВСТУП

Обробка візуальної інформації є одним із найголовніших напрямів сучасної інформатики та штучного інтелекту. Щоденно створюються мільярди цифрових зображень, значна частина яких містить текстову інформацію різними мовами. Туристи фотографують меню в іноземних містах, міжнародні компанії локалізують рекламні матеріали, користувачі потребують швидкого перекладу текстової інформації з документів та фотографій. Ручна обробка таких зображень є трудомістким процесом. Саме тому розроблення інтелектуальних систем автоматичного перекладу тексту на зображеннях набуває особливої актуальності.

Переклад тексту на зображеннях є комплексною задачею, яка вимагає не лише розпізнавання та перекладу текстової інформації, але й відтворення перекладеного тексту зі збереженням шрифту, кольору, орієнтації, та особливо стилістики тексту на зображенні. Такі технології застосовуються у мобільних застосунках для миттєвого перекладу, в електронній комерції для локалізації зображень товарів, у міжнародному документообігу для перекладу сканованих документів та для перекладу іншомовних діаграм або схем.

Актуальність роботи полягає у зростаючій потребі ефективної обробки візуальної текстової інформації з високою точністю та збереженням візуальної автентичності. Існуючі підходи стикаються з проблемою каскадних помилок, коли неточності на одному етапі поширюються на наступні. Хмарні сервіси забезпечують найвищу точність, але вимагають стабільного інтернет-з’єднання та підіймають питання конфіденційності. Локальні рішення гарантують приватність, проте часто поступаються у точності та не забезпечують якісного відтворення візуального стилю. Існуючі сервіси можуть працювати добре на одному зображенні та працювати погано на інших, тому було б гарно додати налаштування до таких систем, щоб користувач міг сам налаштовувати як саме йому треба перекласти зображення.

Сучасний стан досліджень свідчить про ефективність модульних підходів, що поєднують спеціалізовані моделі для кожного етапу обробки: CRAFT та EAST для детекції, CRNN та TrOCR для розпізнавання, Transformer для машинного перекладу, генеративні моделі для візуальної реконструкції.

Наукова задача полягає у систематичному аналізі існуючих методів детекції, розпізнавання, перекладу та візуального відтворення тексту на зображеннях, а також у розробці системи автоматичного перекладу з можливістю свого налаштування, яка забезпечує оптимальний баланс між точністю, збереженням візуального стилю та швидкістю обробки для локальних застосунків.

1 АНАЛІЗ ІСНУЮЧИХ МЕТОДІВ ПЕРЕКЛАДУ ТЕКСТУ НА ЗОБРАЖЕННЯХ

1.1 Теоретичні основи задачі

Автоматичні системи перекладу тексту на зображеннях вирішують задачу локалізації, розпізнавання, перекладу та візуального відтворення текстової інформації зі збереженням оригінального стилю зображення. Застосування таких систем охоплює туризм, документообіг, електронну комерцію та медицину.

Переклад тексту на зображеннях відрізняється від традиційних систем OCR, оскільки вимагає не лише розпізнавання символів, а й заміни тексту перекладеним варіантом зі збереженням візуальної структури. У теоретичній основі задачі виділяють чотири взаємопов’язані етапи.

Етап 1. Детекція текстових областей визначає координати обмежувальних прямокутників для кожного фрагменту з викликами у вигляді варіативності масштабу, довільної орієнтації та перспективних спотворень.

Етап 2. Розпізнавання символів – перетворює локалізовані фрагменти у текст з урахуванням різноманітності шрифтів та систем письма.

Етап 3. Машинний переклад – забезпечує коректний переклад з урахуванням обмежень простору та культурної адаптації.

Етап 4. Візуальна реконструкція – включає відновлення фону та рендеринг перекладеного тексту зі збереженням стилю.

Існуючі підходи до вирішення задачі можна класифікувати за кількома критеріями. За рівнем інтеграції розрізняють:

– модульні системи (окремі моделі для кожного етапу);

– наскрізні системи (єдина нейронна мережа виконує весь процес);

– гібридні системи (комбінація класичних алгоритмів з нейромережами).

За платформою розрізняють:

– хмарні сервіси з залежністю від інтернет-з’єднання та закритими алгоритмами;

– локальні застосунки, які забезпечують швидкість та приватність даних;

– та системи граничних обчислень для обробки на мобільному пристрої в реальному часі.

Однією із особливостей цих етапів є каскадний характер помилок – неточності на одному етапі поширюються на наступні. Якщо область з текстом не виявлена на етапі детекції, система розпізнавання її не побачить. Неточне розпізнавання викривлює сенс всього речення для перекладача. Якщо перекладений текст довший за оригінал, він може не вміститися в доступну область. Складні текстури ускладнюють як детекцію, так і процес відновлення зображення [1–7].

1.2 Методи детекції тексту на зображеннях

Детекція тексту на зображеннях є важливим етапом, оскільки від точності локалізації залежить якість подальшого розпізнавання. Текст у природних сценах характеризується великою варіативністю візуальних характеристик: різні розміри шрифтів, кольорова палітра, довільна орієнтація, розміщення на неоднорідному фоні з текстурами та перспективними спотвореннями.

Класичні методи базувалися на ручному проектуванні ознак. Метод MSER виявляє області зображення, стабільні при послідовному пороговому перетворенні [8]. Алгоритм аналізує зображення на різних рівнях порогової бінаризації та ідентифікує регіони, які залишаються незмінними протягом широкого діапазону порогових значень, що характерно для текстових символів з високою контрастністю відносно фону. Алгоритм SWT базується на

припущенні, що текстові символи мають приблизно однакову товщину штриху вздовж усієї літери [9]. Метод обчислює товщину штриху для кожного пікселя шляхом аналізу градієнтів зображення та трасування променів між протилежними краями символу. Попри ефективність для стандартних шрифтів, обидва методи погано справляються з низьким контрастом, шумом та складними текстурами фону.

Морфологічний підхід використовує послідовність морфологічних операцій для виділення текстових областей [19]. Типовий алгоритм застосовує Top-hat трансформацію для виділення світлих об’єктів на темному фоні або Bottom-hat для темних об’єктів на світлому фоні, після чого використовує морфологічне закриття для з’єднання близько розташованих символів у текстові рядки. Аналіз зв’язних компонент дозволяє відфільтрувати нетекстові регіони на основі геометричних характеристик. Морфологічні методи ефективні для текстів з чітким контрастом та регулярним розташуванням, але погано справляються зі складними фонами та варіативними шрифтами. Попри обмеження, класичні методи (MSER, SWT, морфологічні операції) погано справляються з низьким контрастом, шумом та складними текстурами фону.

З появою глибокого навчання у середині 2010-х років розпочався новий етап. Архітектури на основі згорткових нейронних мереж (CNN) здатні автоматично навчатися виявляти візуальні ознаки безпосередньо з даних (рис. 1.1).

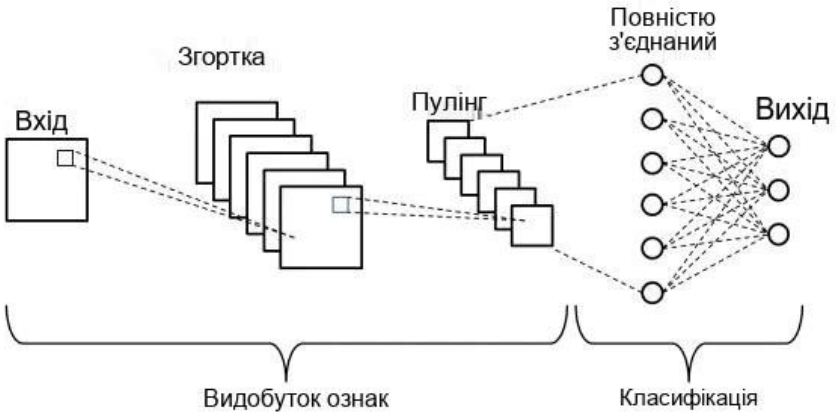

Рисунок 1.1 – Базова CNN архітектура

Метод EAST запропонував наскрізну архітектуру на базі повнозгорткової нейронної мережі [10]. Архітектура складається з мережі (PVANet або VGG-16) для екстракції багаторівневих візуальних ознак – гілки, яка послідовно об’єднує ознаки з різних рівнів для захоплення як дрібних деталей, так і глобального контексту, та вихідного шару, який паралельно генерує дві карти: ймовірність присутності тексту для кожного пікселя та параметри орієнтованих обмежувальних прямокутників. Перевага EAST полягає в прямому передбаченні текстових рядків довільної орієнтації без проміжних етапів генерації кандидатів або постобробки.

Метод CRAFT використовує інший підхід – посимвольну детекцію з аналізом зв’язків між символами [11]. Модель генерує дві карти, які передбачають центральні області окремих символів, та визначають простір зв’язку між сусідніми символами одного слова. Це дозволяє ефективно обробляти криволінійні тексти та слова з нестандартним розташуванням символів шляхом групування символів на основі їх просторової близькості. Модель використовує архітектуру на базі VGG-16 з U-Net подібною структурою для збереження просторової роздільної здатності.

Алгоритм DB вирішує проблему фіксованого порогу бінаризації шляхом введення навченого адаптивного порогу [12]. Замість підходу з ручним підбором порогового значення, DB навчає мережу одночасно передбачувати ймовірнісну карту тексту та карту адаптивних порогів, які можуть варіюватися для різних частин зображення залежно від локальних умов освітлення та контрасту.

Хмарні сервіси, такі як Google Cloud Vision API та Amazon Textract, використовують моделі, навчені на масштабних приватних датасетах [13, 14]. Ці сервіси демонструють вищу якість завдяки постійному оновленню моделей та доступу до значних обчислювальних ресурсів, підтримують понад 50 мов та здатні обробляти складні сценарії з низькою роздільною здатністю та перспективними спотвореннями. Основні обмеження включають залежність

від інтернет-з’єднання, латентність передачі даних, питання конфіденційності та комерційну модель оплати (табл. 1.1).

Сучасні дослідження зосереджені на застосуванні Vision Transformers, які використовують механізм самоуваги для моделювання глобальних залежностей між частинами зображення, та розробці легковагових архітектури для мобільних пристроїв на базі MobileNet та EfficientNet [15].

Таблиця 1.1 – Точність методів детекції тексту

Метод	Рік	Принцип роботи	Точність (ICDAR 2015)	Особливості	Обмеження
MSER	2002	Аналіз стабільних регіонів за контрастністю	64%	Простота реалізації, не потребує навчання	Чутливість до шуму, низький контраст
SWT	2010	Аналіз товщини штрихів символів	70%	Інваріантність до шрифтів та розмірів	Залежність від якості детекції країв
EAST	2017	Повнозгорткова CNN з прямим передбаченням	78%	End-to-end, довільна орієнтація	Проблеми з криволінійними текстами
CRAFT	2019	Посимвольна детекція з affinity score	87%	Криволінійні тексти, адаптація до форми	Більша обчислювальна складність
DB	2020	Навчальна адаптивна бінаризація	85%	Баланс швидкості та точності	Потребує налаштування гіперпараметрів
Google Vision	2024	Пропрієтарна хмарна модель	91%	Підтримка 50+ мов, гарна точність	Залежність від інтернету, платний

1.3 Методи розпізнавання тексту на зображеннях

Розпізнавання символів перетворює візуальну інформацію локалізованих текстових областей на машиночитний текст.

Одним із перших підходів до розпізнавання тексту на зображеннях стала архітектура CRNN. Ідея полягає в поєднанні переваг двох типів нейромереж. згорткові шари ефективно екстрагують просторові візуальні ознаки, тоді як рекурентні шари моделюють послідовний характер текстової інформації. Архітектура складається з трьох послідовних етапів обробки. На першому етапі згорткові шари перетворюють вхідне зображення на карту ознак.

$$H \times W \times C, \tag{1.1}$$

де H – висота;

W – ширина;

C – кількість каналів.

Модуль перетворення карти у послідовність інтерпретує кожен із W вертикальних стовпців цієї карти як окремий вектор ознак, що відповідає певному часовому кроку в послідовності. Двонаправлений LSTM обробляє отриману послідовність векторів, кодуючи контекстну інформацію у прямому та зворотному напрямках.

CRNN має механізм CTC, який вирішує проблему невідповідності довжини вихідної послідовності та цільового тексту. CTC вводить спеціальний пустий символ та дозволяє моделі генерувати послідовності довільної довжини, які потім автоматично згортаються до фінального тексту шляхом видалення повторів ідентичних символів та пустих символів. CTC обчислює ймовірність цільової послідовності як суму ймовірностей усіх можливих вирівнювань через алгоритм динамічного програмування. Це дозволяє навчати модель без необхідності точної по символьної сегментації зображення на етапі навчання. На датасеті IIIT5K CRNN досяг точності 78%, а на Street View Text – 81% [4]. Математично CTC обчислює ймовірність як суму всіх можливих вирівнювань.

$$P(y|x) = \sum_{\pi \epsilon B^{-1}(y)} \prod_{t=1}^{T} p(\pi_t|x)\,, \qquad (1.2)$$

де y – цільова послідовність;

π – можливе вирівнювання довжини T;

$B^{-1}(y)$ – множина вирівнювань що згортаються до y.

Проте CRNN має обмеження при роботі зі спотвореним текстом. Цю проблему вирішує архітектура ASTER (An Attentional Scene Text Recognizer with Flexible Rectification), яка інтегрує попередню ректифікацію зображення [16]. Перший компонент системи – Spatial Transformer Network (STN) з Thin-Plate Spline (TPS) трансформацією – автоматично навчається виправляти перспективні спотворення та викривлення. STN передбачає позиції контрольних точок на межах текстової області і обчислює TPS трансформацію для перетворення вхідного зображення у нормалізовану форму з горизонтально вирівняним текстом. Другий компонент ASTER використовує архітектуру послідовність-у-послідовність з механізмом уваги. Енкодер на базі ResNet екстрагує ознаки з ректифікованого зображення, а декодер генерує послідовність символів, використовуючи увагу для фокусування на релевантних частинах закодованого представлення. На кожному кроці декодування механізм обчислює вагові коефіцієнти, що визначають важливість різних просторових регіонів для генерації поточного символу. Завдяки ректифікації та механізму уваги ASTER досягає значно вищої точності – 93% на IIIT5K та 90% на SVT [16].

Але якісним покращенням в OCR відбулося з появою TrOCR – повністю трансформерної архітектури без використання згорткових мереж [17]. На відміну від попередніх підходів, TrOCR базується виключно на механізмах уваги. Енкодер використовує попередньо навчені Vision Transformer моделі, які розбивають вхідне зображення на патчі фіксованого розміру, лінійно проектують їх у вектори та обробляють через стек трансформів-шарів.

Декодер TrOCR використовує попередньо навчені текстові трансформери та генерує текст на рівні підслівних одиниць замість окремих символів. Це дозволяє ефективніше моделювати морфологічні закономірності та краще впоратися з рідкісними словами. TrOCR-Large демонструє найвищі результати серед розглянутих архітектур – 94% на IIIT5K та 96% на SVT [17] (табл. 1.2).

Таблиця 1.2 – Точність методів розпізнавання тексту

Метод	Рік	Архітектура	Точність (IIIT5K)	Точність (SVT)	Переваги	Недоліки
Tesseract	2018	LSTM-based	62%	53%	Підтримка 100+ мов, безкоштовний	Нижча точність на складних сценах
CRNN	2015	CNN + BiLSTM + CTC	78%	81%	End-to-end навчання, невелика модель	Проблеми з деформованими текстами
ASTER	2018	STN + TPS + Attention	93%	90%	Корекція, геометричних спотворень	Більше параметрів, повільніше
TrOCR-Large	2021	ViT + Text Transformer	94%	96%	Найвища точність, трансферне навчання	Великий розмір моделі (558M параметрів)

Перевага декодеру TrOCR полягає у використанні трансферного навчання через попереднє навчання компонентів на великих загальних датасетах з подальшим покращенням на специфічних OCR датасетах. Енкодер моделі Vision Transformer попередньо навчений на ImageNet для загального розуміння візуальних патернів, а текстовий декодер – на великих текстових корпусах для моделювання мовних закономірностей. Механізм уваги в обох компонентах дозволяє моделювати далекі залежності між елементами без обмежень рецептивного поля згорткових мереж.

У практичних застосуваннях широко використовуються готові рішення. Tesseract 5.x залишається однією з найпопулярніших OCR бібліотек завдяки підтримці понад 100 мов та безкоштовній ліцензії. Сучасні версії використовують LSTM-мережі для розпізнавання, проте показують меншу точність порівняно зі спеціалізованими моделями глибокого навчання – 62% на IIIT5K та 53% на SVT [18].

Модель PaddleOCR PP-OCRv4 від Baidu орієнтована на баланс точності та швидкості [12]. Система включає легковагові моделі для мобільних пристроїв та модульну архітектуру з окремими компонентами для детекції тексту, його розпізнавання та визначення орієнтації.

1.4 Методи перекладу та відтворення тексту

Переклад тексту на зображеннях поєднує машинний переклад текстової інформації та її візуальне відтворення. На відміну від класичного перекладу документів, тут необхідно враховувати просторові обмеження та візуальну цілісність. Сучасні системи машинного перекладу базуються на архітектурі Transformer з механізмом уваги, що дозволяє враховувати контекст усього речення. Google Translate та DeepL підтримують понад 100 мов, NLLB від Meta підтримує 200 мов включаючи низькоресурсні, які забезпечують багатомовний переклад без проміжної англійської.

Особливістю перекладу для зображень є робота з короткими фрагментами (1 – 5 слів) та адаптація до зміни довжини тексту. Англійська «Exit» (4 символи) – українська «Вихід» (5 символів), англійська «Open» – німецька «Geöffnet» (9 символів). Необхідна культурна та збереження формату з урахуванням доступного простору.

Для візуального відтворення існує кілька підходів. Прості графічні методи накладають переклад після замальовування фону, реалізовані у Google Translate AR, забезпечують швидкість на мобільних пристроях, але не

зберігають стиль. Методи відновлення зображення використовують генеративні моделі для відновлення фону [6]. Глибокі нейронні мережі для відновлення зображення навчаються відтворювати правдоподібні текстури. Після відновлення фону перекладений текст рендериться з використанням шрифтів, наближених до оригінальних.

Сучасні генеративні підходи використовують дифузійні моделі для автоматичної вставки перекладу у стилі оригіналу з відтворенням гарнітури, кольорової палітри, візуальних ефектів (тінь, обводка, градієнт) та орієнтації тексту. Процес включає аналіз стилю оригіналу, видалення тексту з відновленням фону та генерацію перекладу у стилі оригіналу. Ці методи забезпечують найвищу якість, але потребують потужних обчислювальних ресурсів та повільні [6].

Але існують проблеми, які включають зміну довжини тексту (переклад може не вміститися в область або залишити порожній простір), збереження стилю (відтворення шрифту, кольору, ефектів), складні фони (текст на текстурованих поверхнях – дерево, метал, тканина) та культурну адаптацію (врахування культурного контексту, зміна напрямку читання).

1.5 Датасети, метрики та виклики розвитку

Для навчання та оцінювання систем перекладу тексту на зображеннях використовуються спеціалізовані датасети (табл. 1.3). ICDAR 2015 містить понад 1500 зображень з природних сцен, з різними умовами освітлення, орієнтаціями та перспективними спотвореннями.

Датасет Total-Text включає 1555 зображень з криволінійними текстами, що вигинаються по контурах об’єктів, та використовується для оцінки геометричних деформацій.

Датасет COCO-Text – складається з 63686 зображень, що охоплює широкий спектр шрифтів, розмірів та орієнтацій, містить як читабельний, так і нечитабельний текст.

Датасет SynthText є синтетичним датасетом з 800000 зображень, створених накладанням синтетичного тексту на природні зображення, використовується для попереднього навчання завдяки великому обсягу та можливості контролювати параметри генерації.

Датасет IAM Handwriting Database містить понад 115 тисяч слів рукописного тексту від 657 авторів та використовується для оцінки розпізнавання рукописного тексту.

Для оцінювання використовують метрики для трьох кроків обробки.

Крок 1. Детекція та розпізнавання оцінюються через відношення правильно знайдених областей до всіх знайдених.

Крок 2. Recall – відношення правильно знайдених до всіх існуючих.

Крок 3. F-measure – гармонічне середнє між ними.

Якість перекладу оцінюється наступними метриками:

– метрика BLEU визначає схожість з еталонним перекладом через n-грами;

– метрика COMET є нейромережевою метрикою, що краще корелює з людською оцінкою.

Для коротких фраз BLEU показує обмежену кореляцію, тому перевага віддається COMET. Візуальна якість вимірюється через SSIM – структурна подібність, вище 0.90 означає добре збереження структури, PSNR – співвідношення сигнал/шум, вище 30 dB для добрих результатів, LPIPS – подібність на основі глибоких ознак.

Багатомовність вимагає підтримки мов з різними скриптами – більшість систем демонструють високу якість для англійської, але для рідкісних мов якість низька через відсутність навчальних датасетів. Балансування якості та швидкості важливе для мобільних застосунків – генеративні методи потребують секунди на обробку, що є не дуже добрим для роботи в реальному

часі. Збереження стилю при відтворенні складних шрифтів, градієнтних заливок та візуальних ефектів також залишається великою проблемою. Каскадні помилки на ранніх етапах поширюються на наступні, призводячи до некоректного результату.

Таблиця 1.3 – Характеристики датасетів для розпізнавання тексту

Датасет	Кількість зображень	Особливості	Призначення
ICDAR 2015	1500	Сценовий текст, різні умови	Benchmark детекції/OCR
Total-Text	1555	Криволінійні тексти	Тестування робастності
COCO-Text	63686	Різноманітні шрифти	Навчання на великих даних
SynthText	800000	Синтетичні дані	Попереднє навчання
IAM Database	115000 слів	Рукописний текст	Оцінка HTR

Перспективи розвитку пов’язані з інтеграцією мультимодальних моделей, які одночасно аналізують візуальний і текстовий контекст та враховують його при перекладі [19].

1.6 Постановка задачі дослідження

Проведений аналіз показав, що незважаючи на досягнення в окремих етапах обробки, залишається ряд проблем, які обмежують практичне застосування систем перекладу тексту на зображеннях. Більшість існуючих

систем використовують модульний підхід з незалежними компонентами, що призводить до накопичення каскадних помилок [7]. Хмарні сервіси забезпечують найвищу точність, але вимагають стабільного інтернет-з'єднання та підіймають питання конфіденційності даних [13]. Локальні рішення забезпечують приватність, але часто поступаються у точності.

Об'єктом дослідження є процес автоматичного перекладу текстової інформації на зображеннях із збереженням оригінального стилю.

Метою дослідження є розробка системи автоматичного перекладу тексту на зображеннях, яка забезпечує оптимальний баланс між якістю та швидкістю.

Для досягнення мети необхідно вирішити наступні задачі:

– провести порівняльний аналіз доступних моделей детекції та розпізнавання для визначення оптимального поєднання точності та швидкості;

– розробити архітектуру системи з мінімізацією каскадних помилок між етапам;

– реалізувати модуль відновлення фону та візуального відтворення тексту зі збереженням базових стилістичних характеристик без ресурсномістких генеративних моделей;

– провести експериментальне дослідження з оцінкою якості за метриками точності розпізнавання, адекватності перекладу та візуальної цілісності;

– оцінити продуктивність системи та визначити можливості оптимізації для пристроїв з обмеженими ресурсами.

Очікуваним результатом є функціонуюча система перекладу тексту на зображеннях, яка працює локально без залежності від хмарних сервісів.

2 ВИБІР ТА ТЕОРЕТИЧНЕ ОБҐРУНТУВАННЯ МЕТОДІВ ДЕТЕКЦІЇ, РОЗПІЗНАВАННЯ, ПЕРЕКЛАДУ ТА ВИДАЛЕННЯ ТЕКСТУ

2.1 Загальна архітектура запропонованого підходу

Автоматичний переклад тексту на зображеннях з відтворенням візуального стилю представляє комплексну задачу що потребує інтеграції методів комп’ютерного зору, обробки природної мови та генерації графічного контенту. Запропонована система реалізує модульний підхід, де кожен етап виконує специфічну функцію у загальному процесі трансформації зображення.

2.1.1 Опис загальної структури системи

Загальна архітектура системи обробки тексту на зображеннях складається з п’яти основних етапів (рис. 2.1).

Етап 1. Відбувається детекція текстових областей. Тут визначаються координати регіонів із текстом та формуються рамки. Це важливий крок, адже точність подальшого розпізнавання безпосередньо залежить від того, наскільки добре був знайдений текст. Етап детекції використовує згорткові нейронні мережі що генерують карти ймовірностей присутності тексту та геометричні параметри боксів з підтримкою довільної орієнтації [21]. Розпізнавання базується на архітектурах CRNN або Transformer що моделюють послідовність символів з контекстними залежностями. Машинний переклад виконується через великі мовні моделі з урахуванням контексту всіх фрагментів одночасно [22]. Візуальне відтворення поєднує методи відтворення фону для видалення тексту та адаптивний рендеринг з підбором параметрів шрифту [23].

Розпізнавання символів перетворює візуальну інформацію на машиночитний текст, враховуючи різноманітність шрифтів, розмірів і

можливих спотворень. Для цього використовуються сучасні архітектури такі як CRNN або Transformer, які здатні враховувати контекст символів у послідовності.

Етап 2. На цьому этапі алгоритм програми намагається знайти дуже близькі знайдені текстові бокси, за потрібністю об’єднати їх, дивлячись на обраний поріг.

Етап 3. Виконується машинний переклад. Система трансформує текст на цільову мову, зберігаючи сенс кожного фрагмента та ураховуючи контекст усіх речень.

Етап 4. Текст видаляється за допомогою методів реконструкції фону.

Етап 5. Відбувається інтеграція перекладеного тексту у зображення, генерується перекладений текст зі збереженням візуального стилю оригіналу.

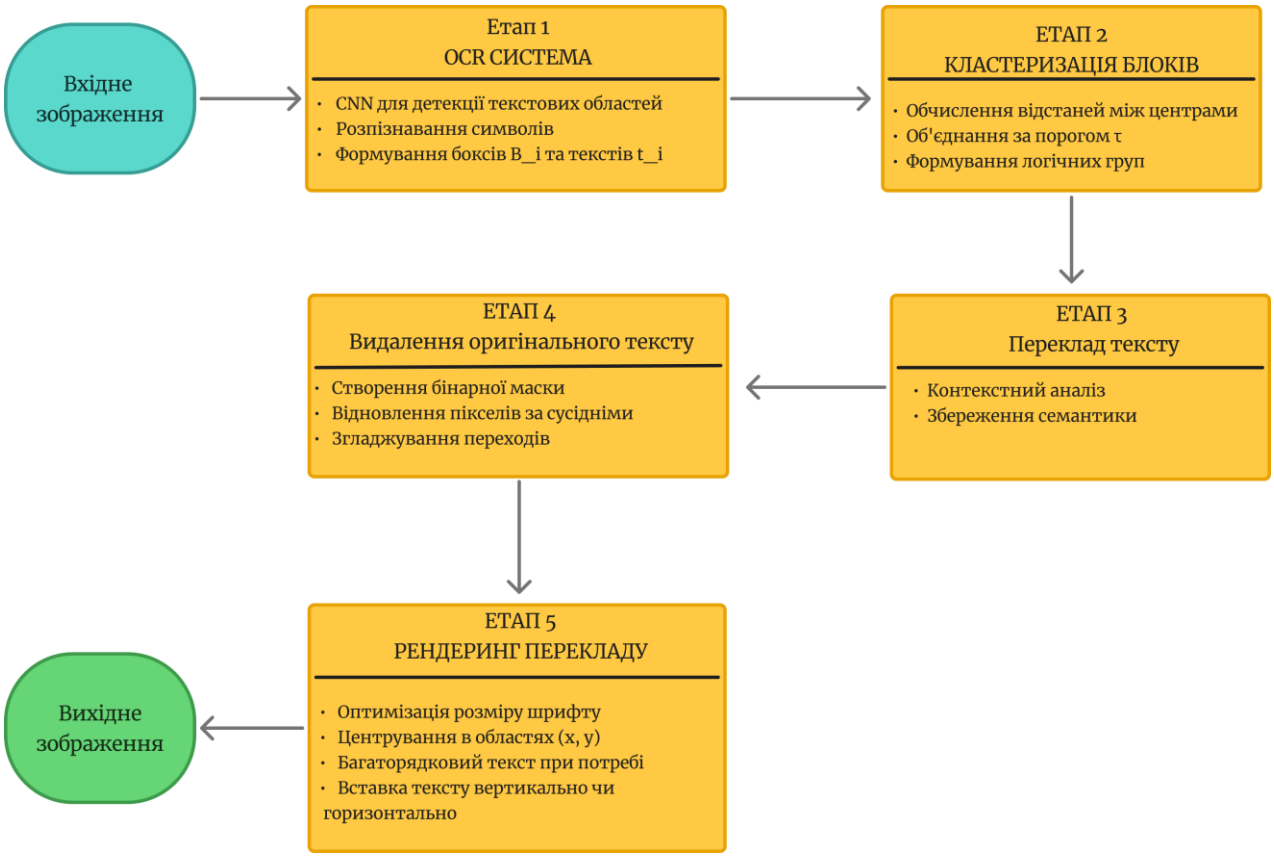

Рисунок 2.1 – Блок-схема загальної архітектури системи

Розробка системи вимагає балансування між точністю, швидкістю та апаратними обмеженнями. Функціональні вимоги включають підтримку понад п’ятнадцяти мов детекцію текстів з довільною орієнтацією включаючи горизонтальний, вертикальний, збереження візуального стилю після перекладу, контекстний переклад з урахуванням просторового розташування блоків [24, 25].

2.1.2 Взаємодія між компонентами

Взаємодія між компонентами забезпечується стандартизованими форматами даних та чітко визначеними інтерфейсами. Кожен етап отримує структуровані дані від попереднього, виконує трансформації та передає результати наступному, зберігаючи відповідність між текстовими блоками та координатами.

Попередня обробка передає покращене зображення модулю детекції у форматі масиву пікселів з нормалізованими значеннями. Модуль детекції повертає список координат обмежувальних рамок де кожен бокс представлений чотирма точками у форматі $\{(x_1, y_1), (x_2, y_2), (x_3, y_3), (x_4, y_4)\}$. Модуль розпізнавання формує множину трійок $\{(B_i, t_i, c_i)\}$ де B_i – координати, t_i – розпізнаний текст, c_i – впевненість. Модуль перекладу обробляє текстові фрагменти з урахуванням їх просторового розташування для забезпечення контекстної узгодженості. Модуль візуального відтворення отримує оригінальне зображення, координати, оригінальні тексти та переклади для формування фінального результату.

Після детекції виконується злиття близько розташованих текстових областей у логічні блоки оскільки OCR-системи часто виявляють окремі слова як незалежні блоки тоді як для коректного перекладу потрібен контекст цілого речення. Операція злиття визначається як:

$$merge(B_i, B_j) = \begin{cases} true, \ if \ d_h(B_i, B_j) \leq \theta_x \wedge \ d_v(B_i, B_j) \leq \ \theta_y \\ false, \qquad\qquad\qquad\qquad\qquad otherwise \end{cases}, \quad (2.1)$$

де d_h та d_v – горизонтальна та вертикальна відстані між боксами;

θ_x та θ_y– порогові значення що визначаються автоматично на основі середнього розміру символів.

2.2. Методи детекції та розпізнавання тексту на зображеннях

OCR-системи базуються на методах глибокого навчання та складаються з кількох взаємопов'язаних компонентів. Класична архітектура включає три етапи.

Етап 1. Детекція локалізує позиції текстових фрагментів та формує обмежувальні рамки.

Етап 2. Розпізнавання перетворює візуальні шпблони у послідовності символів.

Етап 3. Постобробка виконує лінгвістичну корекцію через словники та мовні моделі. Такі системи часто використовують згорткові нейронні мережі для виділення ознак та рекурентні архітектури для послідовного розпізнавання символів. Завдяки цьому OCR-рішення досягають високої точності навіть на зображеннях зі складним фоном, різними шрифтами чи нахиленим текстом.

2.2.1 Загальна архітектура OCR-систем

Текст характеризується варіативністю, а саме різні розміри шрифтів, кольорова палітра, довільна орієнтація, розміщення на неоднорідному фоні з текстурами та перспективними спотвореннями. Сучасні методи

використовують згорткові мережі що автоматично навчаються виявляти візуальні ознаки без ручного проектування [26].

Математично OCR описується як композиція функцій. Детекція визначається як $f_{det} : I \to B$ що відображає зображення у множину обмежувальних рамок. Розпізнавання – $f_{rec} : (I, B_i) \to t_i$ що для кожної області генерує текстовий рядок.

Загальний процес повертає множину пар координат-текст.

$$OCR(I) = \{(B_i, f_{rec}(I, B_i))\}_{i=1}^{n}, \tag{2.2}$$

де f_{rec} – функція розпізнавання;

B_i – обмежувальна рамка (конкретна область на зображенні);

(I, B_i) – зображення та конкретна рамка B_i.

2.2.2 Методи детекції тексту

Сучасні підходи детекції класифікуються на методи прямого передбачення рамок, посимвольної сегментації та семантичної сегментації.

Модель CRAFT базується на посимвольній детекції з аналізом просторових зв’язків між символами. На відміну від підходів що передбачають текстові рядки цілком, модель CRAFT спочатку виявляє окремі символи а потім групує їх у слова. Архітектура використовує повнозгорткову мережу на базі VGG-16 з U-Net структурою для збереження просторової роздільної здатності (рис. 2.2).

Мережа генерує дві карти ймовірностей. Перша карта оцінка області – передбачає центральні області окремих символів де високі значення вказують на присутність символу. Друга карта оцінка зв’язку – визначає простір зв’язку між сусідніми символами одного слова. Текстові області формуються через

порогову бінаризацію та аналіз зв'язних компонент. Ключова перевага – ефективна обробка криволінійних текстів та слів з нестандартним розташуванням символів. Недолік – вища обчислювальна складність через необхідність обробки кожного символу окремо.

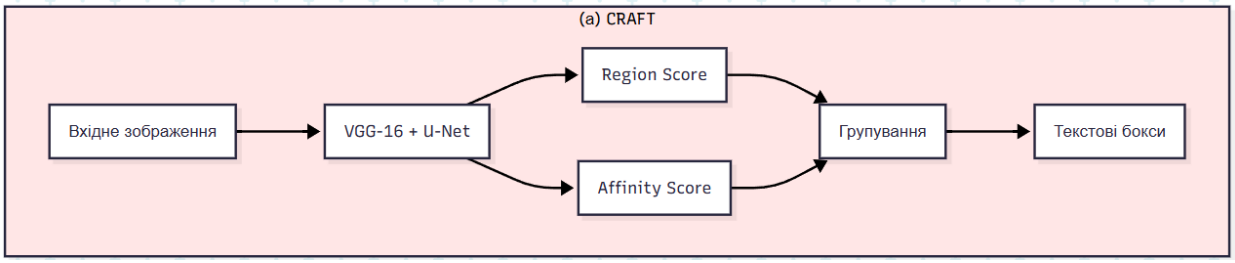

Рисунок 2.2 – CRAFT – посимвольне виявлення з картами регіонів та коефіцієнтів зв'язності

Модель DBNet вирішує проблему фіксованого порогу бінаризації через навчені адаптивні пороги [27]. Традиційні методи використовують фіксоване порогове значення що погано працює для зображень з нерівномірним освітленням. Модель DBNet навчає мережу одночасно передбачувати ймовірнісну карту тексту та карту адаптивних порогів які варіюються для різних частин зображення. Архітектура включає нейромережові архітектури ResNet або MobileNetV3 як основну мережу для екстракції ознак, Feature Pyramid Network для об'єднання багаторівневих ознак та дві паралельні голови для передбачення карт. Процес бінаризації визначається як:

$$B(x,y) = \frac{1}{1 + e^{-k(P(x,y)-T(x,y))}}, \tag{2.3}$$

де $P(x,y)$ – ймовірність присутності тексту;

$T(x,y)$ – адаптивний поріг;

K – параметр крутизни сигмоїди.

Диференційованість операції дозволяє наскрізне навчання всієї мережі. Це забезпечує оптимізацію всіх етапів детекції та адаптацію порогів до локальних умов (рис. 2.3).

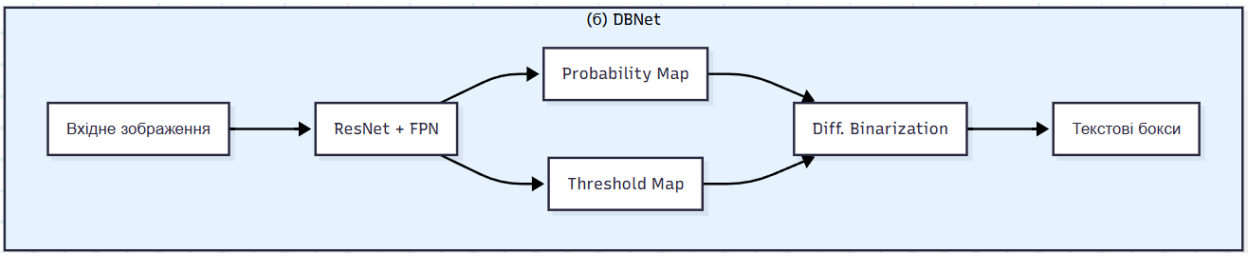

Рисунок 2.3 – DBNet – адаптивна бінаризація з картами ймовірності та порогів

Модель SegFormer використовує трансформерну архітектуру для семантичної сегментації [28]. Трансформери моделюють глобальні залежності між частинами зображення що дозволяє краще захоплювати контекст довгих текстових рядків. Модель SegFormer розбиває зображення на патчі фіксованого розміру які лінійно проектуються у вектори. Вектори обробляються через стек трансформерних блоків. На виході формується багаторівневе представлення ознак, яке поєднує як локальну, так і глобальну інформацію про структуру зображення. Декодер SegFormer є легким і ефективним – він об’єднує багатомасштабні ознаки без використання складних операцій, що забезпечує швидкість роботи та точність сегментації.

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

де Q, K, V – проекції query, key, value;

d_k – розмірність key векторів.

На відміну від згорток з обмеженим рецептивним полем, механізм уваги дозволяє кожному патчу взаємодіяти з усіма іншими незалежно від відстані (рис. 2.4).

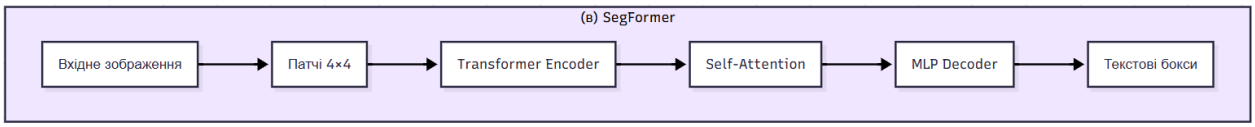

Рисунок 2.4 – SegFormer – трансформерна сегментація з патчами

Порівняння методів показує компроміси. Модель CRAFT забезпечує найкращу точність для криволінійних текстів але потребує більше ресурсів. DBNet демонструє оптимальний баланс між швидкістю та точністю. Vision Transformer методи показують найкращу здатність моделювати довгі залежності але потребують більше пам’яті.

2.2.4 Порівняльний аналіз OCR-систем

Вибір OCR-системи вимагає врахування архітектурних особливостей, підтримки мов, обчислювальних вимог та можливостей налаштування.

У межах дослідження було використано чотири системи оптичного розпізнавання тексту, а саме EasyOCR, PaddleOCR, Tesseract та Surya OCR. Вибір цих моделей обумовлений їхньою популярністю в академічних та прикладних задачах, підтримкою розпізнавання тексту у природних сценах, у різних діаграмах та схемах, або у зображеннях документного вмісту та відмінностями в архітектурних підходах [29 – 31]. Усі рішення тестувались для англомовних зображень у режимі CPU-обчислень.

EasyOCR – це багатомовна OCR-система на основі глибоких нейромереж, що використовує комбінацію детектора текстових областей та розпізнавальної моделі на основі CRNN-архітектури з модулем CTC-декодування. Модель підтримує понад 80 мов, проте в межах експерименту застосовувалася лише англійська. EasyOCR автоматично виконує попередню нормалізацію текстових фрагментів і прогнозує оцінку впевненості кожного розпізнаного блоку. У проведених тестах використовувалась стандартна

конфігурація без ручного втручання в попередню обробку, а модель запускалася в CPU-режимі з вимкненим GPU-прискоренням для збереження єдиних умов тестування. До переваг можна віднести просту установку, гарну швидкість на GPU, мінімальні налаштування, реалістична оцінка впевненості. Але недоліки також серйозні, а саме середня точність (CER 12 – 30%), проблеми з викривленим текстом та погіршення на багатотекстових сценах.

Tesseract – OCR-система з відкритим кодом, що активно розвивається та підтримує понад 100 мов. Використовує LSTM-архітектуру для розпізнавання тексту і простий CTC-декодер. Перевагами Tesseract є безкоштовність, велика спільнота, підтримка багатомовності, можливість донавчання на власних даних. Але через серйозні недоліки вона відстає від трьох інших систем. Нижча точність на складних сценах і багатоколонних документах, обмежені можливості попередньої обробки, відносно низька швидкість CPU роблять її найгіршим варіантом.

PaddleOCR є модульною системою розпізнавання тексту, розробленою на базі фреймворку PaddlePaddle, і включає окремі модулі для детекції, корекції, розпізнавання та постобробки. У дослідженні використовувалася конфігурація з такими параметрами, зокрема увімкнена класифікація орієнтації документа, вимкнене випрямлення сторінок та активоване построчне вирівнювання. Важливими параметрами PaddleOCR є також тип алфавіту поріг впевненості для сегментації тексту, тип розпізнавальної мережі (за замовчуванням CRNN), а також параметри постпроцесингу, які контролюють фільтрацію дубльованих та низьковпевнених текстових сегментів. PaddleOCR має вбудований механізм обчислення впевненості розпізнавання для кожного рядка, що дозволило уніфікувати формат виводу з іншими моделями. З переваг можна виділити баланс точності та швидкості, модульність, гнучкість налаштування. Щодо недоліків, то це є консервативна детекція на складних фонах, помилки класифікації орієнтації на нестандартних сценах.

Surya OCR – сучасна система розпізнавання тексту на основі трансформер архітектури, орієнтована на якісну обробку як сценічного тексту, так і щільно структурованих документів. Її ключовою перевагою є можливість застосування двох режимів роботи, серед яких із використанням детектора текстових блоків та без нього, коли на вхід розпізнавання передається весь кадр як єдина обмежувальна рамка. У дослідженні детектор активувався лише для датасету SROIE, де текст має складну багатоблокову структуру, тоді як для IIIT5K та ICDAR 2013 використовувався повний кадр без попередньої локалізації текстових областей. Surya OCR демонструє особливо високу стійкість до нерівномірного вирівнювання, нахилу символів та нерегулярної щільності тексту. Модель генерує не тільки розпізнаний рядок, але й оцінку впевненості, усереднену по всіх текстових сегментах зображення. Щодо переваг – це найвища точність (CER 0.5 – 18.4%), найкраща багатомовність, хороша робота зі щільним текстом та таблицями. Недоліками є низька швидкість на CPU (до 158 с/зображення), високе споживання пам’яті (табл. 2.1).

Таблиця 2.1 – Порівняльна характеристика OCR-систем.

Характеристика	EasyOCR	PaddleOCR	Surya OCR	Tesseract v5
Рік розробки	2020	2020–2024	2023–2024	2023
Фреймворк	PyTorch	PaddlePaddle	PyTorch	C++ / LSTM
Архітектура детекції	CRAFT (CNN)	DB (CNN+FPN)	SegFormer (Transformer)	LSTM-based segmentation
Детекції	VGG-like	ResNet/MobileNetV3	Transformer	LSTM
Архітектура розпізнавання	CRNN	CRNN/SVTR	Transformer seq2seq	LSTM
Декодер	CTC	CTC	Attention	CTC
Підтримка мов	80+	80+	90+	100+
Модульність	Низька	Висока	Середня	Низька
Філософія	Простота	Швидкість, модульність	Точність, багатомовність	Багатомовність

2.3 Моделі перекладу тексту

Для перекладу текстів з OCR важливі специфічні виклики, серед яких короткі фрагменти з обмеженим контекстом, помилки розпізнавання, семантичні зв’язки через просторове розташування, обмеження довжини перекладу для вміщення у оригінальну область [32]. Тому системи машинного перекладу часто поєднують нейронні моделі з додатковими механізмами контекстної корекції та просторової прив’язки. Використання мультимодальних підходів, що враховують текстову і візуальну інформацію, дозволяють підвищити якість перекладу та зберегти структуру зображення.

2.3.1 Загальна схема машинного перекладу

Базова архітектура складається з енкодера що обробляє вхідну послідовність та перетворює її у векторне представлення семантичного змісту, та декодера що послідовно генерує символи цільовою мовою. Процес є авторегресивним де кожен символ залежить від попередніх [32].

$$P(y|x) = \prod_{t=1}^{n} P(y_t \mid y < t, x)\,, \tag{2.5}$$

де x – вхідна послідовність;

y – переклад;

$y < t$ – попередньо згенеровані символи.

2.3.2 Використання великих мовних моделей

Великі мовні моделі типу GPT використовують декодерну архітектуру з маскованою самоувагою (masked self-attention) для авторегресивної генерації.

Через це, мовні моделі здатні враховувати ширший контекст та семантичні зв’язки між множинними фрагментами [33].

Для перекладу з зображень можливість контекстного перекладу є, напевно, найважливішою характеристикою. Фрагменти часто семантично пов’язані через просторове розташування наприклад заголовок та підзаголовок. Передача всіх фрагментів одночасно дозволяє моделі розуміти загальний контекст та забезпечувати узгоджений переклад термінології та стилю [34].

Інша перевага – робастність до помилок OCR. Великі мовні моделі навчені на різноманітних текстах включаючи зашумлені дані, що дозволяє краще справлятися з некоректними входами. Модель може ідентифікувати та виправляти помилки розпізнавання базуючись на контексті. Але у цього варіанту перекладу є також недоліки, вища латентність через складність моделі, залежність від API що вимагає інтернету, потенційно менша стабільність результатів через генеративну природу.

2.3.3 Використані моделі перекладу

У розробленій системі оцінювання якості перекладу базувалося на порівнянні трьох концептуально різних підходів:

– масові хмарні NMT-перекладачі, представлені Google Translate;

– оптимізовані комерційні NMT-рушії, зокрема DeepL, що спеціалізуються на якості синтаксичного формулювання;

– великі мовні моделі, які здатні виконувати не лише переклад, а й семантичну реконструкцію тексту в умовах спотвореного тексту, що характерно для результатів OCR [35 – 38].

Google Translate було інтегровано через обгортку «deep_translator», що дозволяє безпосередньо отримувати переклад без розгортання локальної моделі. Основною перевагою сервісу є висока швидкість, багатомовність і

стабільність при роботі з короткими фразами загального призначення. Разом з тим, сервіс не забезпечує повноцінного пакетного CI-сумісного режиму. Кожен OCR-фрагмент відправляється окремим запитом, що при великій кількості текстових сегментів призводить до суттєвого зростання затримки. Важливою обмежувальною властивістю Google Translate є відсутність міжрядкового контексту – кожен фрагмент обробляється ізольовано, без можливості корекції змісту на основі сусідніх слів чи фраз, навіть якщо вони є частиною одного інформаційного простору на зображенні [35].

DeepL використовує трансформерні моделі, оптимізовані для лінгвістичної плавності та граматичної коректності. Порівняно з Google Translate, DeepL краще зберігає природність мовлення, менш схильний до буквальних перекладів і підтримує пакетну передачу масивів рядків, що суттєво підвищує ефективність обробки OCR-тексту на зображеннях із десятками фрагментів. Водночас, попри якіснішу побудову фраз, він також не застосовує смислову реконструкцію у випадках, коли OCR-текст має порушений порядок слів, відсутні частини слів або неповноту текстового контексту. DeepL перекладає поданий текст, а не відновлює його семантичну цілісність [36].

На відміну від класичних NMT-методів, великим мовним моделям властивий принципово інший характер обробки тексту. LLM виконують переклад не як пряме узгодження пар фраз, а як генерацію найімовірнішої цільової послідовності токенів з урахуванням глобального контексту [37 – 38].

Така архітектура дозволяє моделі наступні переваги:

– відновлювати логіку речень, навіть якщо частина слів втрачена або переплутана;

– коригувати порушений синтаксис;

– узгоджувати переклад між кількома фразами одного зображення;

– адаптувати стиль під тип контенту;

– повертати результат у строгому форматі.

Ці властивості є доволі важливими для перекладу OCR-тексту, де типовими є уривчастість, помилки розпізнавання, розриви синтаксичних конструкцій та відсутність чіткої межі речень[39 – 41].

У роботі перевірялися кілька LLM-сервісів API-класу, які реалізують подібний принцип контекстно-керованого перекладу, з яких GPT-3.5-Turbo, GPT-4o-mini, Claude Haiku, Gemini Flash. Попри різні внутрішні архітектури, усі моделі проявили спільні переваги LLM-підходу над класичними NMT-перекладачами, а саме кращу роботу з фрагментами, смислове відновлення пошкодженого тексту, семантичну узгодженість перекладу та можливість точно контролювати формат відповіді через інструкції [40 – 44].

Ще одна перевага для задачі виявилась здатність LLM повертати структурований JSON-масив у тій самій послідовності, що й вхідні OCR-сегменти, що гарантує правильне подальше накладання перекладу на вихідні текстові бокси зображення. Жоден із класичних NMT-перекладачів не дає можливості нав’язати такий формат виводу на рівні інструкцій, тоді як для LLM це стандартний сценарій [42 – 43].

Для фінальної моделі було обрано GPT-4o-mini, оскільки за експериментами вона має наступні переваги:

– найвищу стабільність структурованого JSON-виводу без додаткових пояснень;

– найменшу середню затримку серед моделей, що зберігають контекст на рівні зображення;

– стійкість до поширених OCR-дефектів;

– кращий компроміс між вартістю API-виклику і якістю реконструкції сенсу порівняно з повною GPT-4 [41].

Усі запити до DeepL та LLM виконувалися у пакетному режимі, щоб зменшити кількість зовнішніх запитів, прискорити обробку та забезпечити крос-фразовий контекст у межах одного зображення. Для забезпечення порівняння, вхідні фрази попередньо проходили однаковий фільтраційний етап.

Лістинг 2.1 Формат інструкції, який застосовувався до LLM:

You need to first read all of the words, understand the topic and context. Translate the following list of words/phrases into language with code {target_lang}. If a sentence lacks meaning or word order is broken, reconstruct it using context. Remove unnecessary symbols Return only a valid JSON array of translated strings in the same order, no comments.

Цей підхід дає змогу моделі інтерпретувати набір OCR-фраз як єдиний пов'язаний контекст, а не як незалежні одиниці, що кардинально підвищує адекватність перекладу порівняно з класичним NMT (табл. 2.2).

Таблиця 2.2 – Порівняння систем перекладу.

Характеристика	LLM	Google Translate	DeepL
Архітектура	Decoder-only Transformer з masked self-attention	Encoder-Decoder Transformer	Encoder-Decoder Transformer з ensemble
Підтримка мов	50 – 100 залежно від моделі	130+ мов	30 мов (переважно європейські)
Контекстне розуміння	Високе (обробка множинних фрагментів одночасно)	Середнє (незалежна обробка речень)	Середнє-високе (обмежений контекст)
Швидкість обробки	Повільна (2-5 секунд на запит)	Швидка (<1 секунди)	Середня (1-2 секунди)
Якість для коротких текстів	Висока (особливо з контекстом)	Середня-висока (стабільна)	Дуже висока (найкраща для EU мов)
Робастність до помилок OCR	Висока (може виправляти помилки)	Низька-середня	Середня
Вимоги до інтернету	Так (API)	Так (API)	Так (API)
Можливість локального розгортання	Обмежена (великі моделі)	Ні	Ні
Узгодженість термінології	Висока (між фрагментами)	Низька	Середня
Вартість API	Висока (за токени)	Помірна	Вища за Google
Основне призначення	Контекстний переклад з OCR	Швидкий переклад великих обсягів	Якісний переклад

2.4 Відтворення перекладеного тексту на зображенні

Візуальне відтворення вимагає видалення оригінального тексту без артефактів, відновлення фонової текстури, адаптації параметрів шрифту до зміни довжини тексту, відтворення візуальних характеристик оригіналу включаючи колір, розмір, орієнтацію, тінь, а також збереження загальної естетики зображення [44].

2.4.1 Завдання редагування зображень

Основні проблеми включають складні фонові текстури де текст розміщений на дереві, металі, тканині або природних ландшафтах що ускладнює якісне відновлення фону. Зміна довжини тексту між мовами створює проблему вміщення перекладу у доступну область оскільки коефіцієнти експансії можуть досягати п’ятдесяти відсотків для деяких мовних пар таких як англійська-німецька. Візуальні ефекти такі як тіні, обводки, градієнти додають складності оскільки їх параметри важко визначити автоматично з оригінального зображення [45].

2.4.2 Метод відновлення зображення для видалення тексту

Відновлення зображення представляє клас методів для відновлення відсутніх областей на основі інформації з оточення. У контексті перекладу метод відновлення зображення використовується для видалення текстових областей та відновлення фону.

Класичні методи базуються на диференціальних рівняннях. Метод Telea використовує алгоритм швидкого просування для ітеративного заповнення від країв до центру [35].

$$I(p) = \frac{\sum_{q \in N(p)} w(p,q) \cdot I(q)}{\sum_{q \in N(p)} w(p,q)}, \quad (2.6)$$

де $I(p)$ – значення пікселя що відновлюється;

$N(p)$ – множина сусідніх відновлених пікселів;

$w(p,q)$ – вага від відстані та напрямку градієнта.

Метод ефективний для простих фонів та працює швидко на CPU що робить його придатним для застосувань в реальному часі.

Основне обмеження класичних методів – складність відновлення структурованих текстур де проста дифузія не може адекватно відтворити періодичні патерни. Генеративні підходи використовують глибокі нейронні мережі для навчання складних відображень [45]. GAN та дифузійні моделі демонструють значно вищу якість для складних текстур завдяки здатності розпізнавати та відтворювати високорівневі візуальні патерни. Однак вимагають потужних GPU та мають вищу латентність що робить їх непридатними для обробки в реальному часі на CPU.

Для практичних застосунків класичні методи оптимальні для відтворення в реальному часі на CPU та забезпечують прийнятну якість для більшості випадків.

2.4.3 Реалізація вставки перекладеного тексту

Після видалення тексту наступним етапом є генерація та позиціонування перекладу з максимальним збереженням візуального стилю. Процес включає підбір шрифту, визначення розміру, адаптацію до довжини, позиціонування та відтворення візуальних характеристик.

Підбір розміру шрифту є важливим для читабельності перекладеного зображення. Основна задача є знаходження максимального розміру при якому переклад повністю вміщується за допомогою наступної формули:

$$S = \max\{s : w_{text}(s) \leq w_{box} \wedge h_{text}(s) \leq h_{box}\}, \quad (2.7)$$

де s – розмір шрифту;

$w_{text}(s)$ та $h_{text}(s)$ – ширина та висота тексту;

w_{box} та h_{box} – доступна ширина та висота області.

Алгоритм підбору використовує бінарний пошук у діапазоні від мінімального до максимального розміру. На кожній ітерації обчислюється середній розмір, рендериться текст та перевіряється чи вміщується він у бокс.

Для випадків коли переклад не вміщується в одну строку застосовується розбиття на кілька строк. Алгоритм намагається створити строки приблизно однакової довжини через аналіз можливих позицій розбиття між словами для збалансованого вигляду. Позиціонування тексту виконується через центрування по горизонталі та вертикалі. Визначення кольору виконується через аналіз оригінального зображення у текстовій області з обчисленням домінуючого кольору текстових пікселів [45].

Обробка орієнтації вимагає визначення чи текст був горизонтальним, або вертикальним. Для вертикального тексту він спочатку рендериться горизонтально потім повертається на дев’яносто градусів. Вибір шрифту для різних мов критичний для коректного відображення. Для європейських мов використовуються стандартні шрифти, для азіатських мов необхідні спеціалізовані шрифти з тисячами ієрогліфів.

2.5 Метрики оцінювання якості

Для комплексної оцінки продуктивності системи використовується набір метрик що відображають різні аспекти роботи компонентів.

Метрика Character Error Rate (CER) вимірює частку помилок на рівні символів відносно загальної кількості символів.

$$CER = \frac{S + D + I}{N}, \tag{2.8}$$

де S – кількість замін символів;

D – видалення;

I – вставки;

N – загальна кількість символів у правильному тексті.

Значення коливається від 0 (ідеальне розпізнавання) до 1 і вище де менше значення краще. Ця метрика дозволяє оцінити точність на рівні окремих символів що важливо для розуміння характеру помилок розпізнавання.

Метрика Word Error Rate (WER) аналогічна метрика на рівні слів.

$$WER = \frac{s_w + D_w + I_w}{N_w}, \tag{2.9}$$

де w – позначає операції на рівні слів.

Метрика WER є більш суворою метрикою оскільки навіть одна неправильна літера робить слово хибним. Однак вона краще відображає практичну корисність результатів оскільки для людини важливіше розуміти правильні слова ніж окремі символи [47].

Метрика Frames Per Second показує скільки зображень система здатна обробити за секунду.

$$\mathrm{F}PS = \frac{1}{Average\ Time}. \tag{2.10}$$

що визначає придатність для роботи в реальному часі. Для інтерактивних застосувань бажано досягнення значень вище одиниці що дозволяє обробляти зображення швидше ніж за секунду. Average Time вимірює середній час обробки одного зображення у секундах.

Метрика Average Confidence відображає впевненість системи у власних передбаченнях від 0 до 1.

$$\frac{1}{n}\sum_{i=1}^{n} P(t_i|B_i), \tag{2.11}$$

де $P(t_i \mid B_i)$ – ймовірність розпізнаного текстуt_i для області B_i обчислена моделлю.

Зазвичай корелює з точністю результатів – вищі значення свідчать про більш надійне розпізнавання. Ця метрика дозволяє фільтрувати ненадійні результати встановлюючи порогове значення впевненості нижче якого результати відкидаються або позначаються для ручної перевірки.

Метрика Accuracy вимірює частку повністю правильно розпізнаних зображень від загальної кількості.

$$Accuracy = \frac{E}{T} \cdot 100\%\,, \tag{2.12}$$

де E – кількість зображень, де розпізнаний текст повністю співпадає з еталонним;

T – власна кількість зображень.

Ця метрика є більш суворою ніж CER та WER, оскільки навіть одна помилка робить весь результат хибним. Однак вона найкраще відображає практичну корисність системи для задач, де потрібна абсолютна точність, наприклад розпізнавання номерних знаків, серійних номерів або кодів документів. Значення 100% означає ідеальне розпізнавання всіх зображень, тоді як 0% вказує на відсутність жодного повністю правильного результату.

3 РЕАЛІЗАЦІЯ ПРОГРАМИ ТА ТЕСТУВАННЯ

3.1 Обґрунтування вибору мови програмування для програмної реалізації

Для роботи зі зображеннями, LLM та іншими технологіями ШІ зазвичай обирають не складні мови програмування з широким спектром бібліотек, великою кількість навчальних матеріалів та документацій. Однією та найпопулярнішою мовою для роботи з ШІ та алгоритмами є мова, оскільки вона має широкий набір інструментів для комп'ютерного зору, обробки природної мови та побудови інтерфейсів користувача [46]. Python забезпечує відкриту екосистему бібліотек і добре документовані API, що спрощує створення комплексних систем, які поєднують OCR, машинний переклад і відновлення зображень.

3.1.1 Вибір технологій та бібліотек

Під час вибору бібліотек враховувалися наступні критерії, а саме стабільність і популярність, наявність документації, можливість інтеграції у конвеєр обробки та підтримка необхідних функцій. На основі цього було використано настпупні бібліотеки:

– бібліотека Gradio – бібліотека для створення веб-інтерфейсів машинного навчання, яка дозволяє завантажувати зображення, обирати модель OCR і налаштовувати параметри перекладу [47];

– бібліотека NumPy використовується для роботи з масивами пікселів, об'єднання результатів детекції тексту та підготовки даних для аналізу [48];

– бібліотека Matplotlib застосовується для побудови графіків точності моделей і візуалізації процесу перекладу [49];

– бібліотека Pandas забезпечує роботу з табличними даними, зберігання результатів експериментів і статистичний аналіз [50];

– бібліотека JiWER використовується для розрахунку метрик якості розпізнавання, таких як WER і CER;

– бібліотека Editdistance – для швидкого розрахунку відстані Левенштейна при аналізі помилок розпізнавання;

– бібліотека DeepL – офіційний клієнт для інтеграції з сервісом машинного перекладу DeepL, який показує високу точність перекладу коротких текстів.

Усі бібліотеки встановлювалися у віртуальному середовищі Python з фіксацією версій у файлі requirements.txt. Це забезпечує відтворюваність експериментів і відсутність конфліктів залежностей. Сукупне використання Gradio, NumPy, Pandas та JiWER формує ефективний інструментарій для реалізації, тестування й оцінки якості розробленої системи.

3.1.2 Вибір середовища розробки та інструментів

Середовище розробки на мові Python повинно забезпечувати стабільну роботу з бібліотеками машинного навчання, зручне налагодження коду та швидке тестування. Для цього було обрано інтегроване середовище PyCharm і хмарну платформу Google Colab, які ефективно доповнювали одна одну.

Основна розробка здійснювалася у PyCharm – повнофункціональному середовищі для Python від JetBrains, що підтримує форматування, підсвічування синтаксису, інтеграцію з Git і зручну навігацію у великих проектах [51]. PyCharm забезпечив стабільну роботу з бібліотеками PaddleOCR та EasyOCR, які іноді спричиняли проблеми у хмарних середовищах. PyCharm гарно інтегрує з venv і pip, вміє працювати з великими файлами зображень, через нього дуже зручно налагодження через breakpoints для пошуку багів або слабких місць, крім цього PyCharm має доволі зручну файлову систему та

великий спектр різних налаштувань для комфортної роботи. Недоліком є виконання всіх обчислень лише на локальному обладнанні.

Для тестування моделей і експериментів із параметрами OCR використовувалося середовище Google Colab – хмарна платформа для запуску Python-коду з доступом до GPU [52–53]. Ця платформа доволі популярна через її безкоштовні обчислювальні ресурси, зручну інтеграцію з Google Drive і можливість швидкого обміну результатами. Colab застосовувався для оцінки точності моделей OCR, побудови графіків та візуалізації результатів. Серед обмежень – обмежений час сесії, проблеми сумісності з окремими бібліотеками, а саме PaddleOCR та SuryaOCR і можливе примусове завершення процесів. Через проблему с цими бібліотеками і було рішення перейти до PyCharm для написання основного коду. Тому Colab використовувався переважно на етапі досліджень, а фінальна система розроблялася в PyCharm. Для ізоляції залежностей використовувалося віртуальне середовище venv з фіксацією версій у requirements.txt. Керування вихідним кодом здійснювалося через GitHub, що забезпечувало контроль змін і збереження резервних копій. Для обробки зображень застосовувалися OpenCV та Pillow – високопродуктивні бібліотеки для масштабування, повороту, накладання тексту та візуалізації [54 – 55].

Структура середовища розробки включала локальне середовище PyCharm із venv та бібліотеками Python, зовнішні API (DeepL, Google Translate), вебінтерфейс на Gradio та репозиторій GitHub [56].

3.2 Архітектура та програмна реалізація системи

Алгоритм роботи розробленої системи передбачає послідовну обробку вхідного зображення з метою розпізнавання, перекладу та відновлення текстової інформації іншою мовою.

3.2.1 Особливості інтеграції з інтерфейсом користувача

Інтерфейс користувача системи реалізовано з використанням бібліотеки Gradio, яка забезпечує просту інтеграцію алгоритмів машинного навчання з веб-інтерфейсом. Для створення графічної частини застосовано модель Blocks, що дозволяє компонувати елементи та організовувати взаємодію користувача з модулями програми. Центральним елементом є функція process_image translation, яка координує основні етапи пайплайну, а саме розпізнавання тексту, переклад, видалення тексту з оригінального зображення, накладання перекладу та відображення результату (рис. А.1). Таким чином, Gradio виступає посередником між користувачем і програмною логікою.

Інтерфейс містить елементи керування для налаштування параметрів обробки, завантаження зображення, вибір OCR-моделі та системи перекладу. (рис. А.2).

Для виявлення текстових ділянок використовується одна з моделей OCR, яку користувач обирає самостійно. Система підтримує три основні інструменти:

– PaddleOCR – високоточна модель, що виконується на CPU;

– EasyOCR – універсальний інструмент із підтримкою понад 80 мов;

– SuryaOCR – сучасний багатомовний підхід на базі глибоких нейронних мереж.

Окрім вибору моделі OCR і сервісу перекладу, користувач також може налаштувати параметри сегментації тексту. Зокрема, система надає такі режими об’єднання текстових блоків, де px – кількість пікселів (по x, по y, або разом):

– Close (5 px) – для щільно розташованого тексту;

– Medium (10 px) – універсальний режим;

– Far (100 px) – для віддалених написів або окремих фрагментів.

Додатково можна вибрати режим багаторядкової обробки. Цей параметр був доданий для більш кращої вставки тексту. Часто виходить, що

перекладений текст більший за оригінал, і через це, параметр потрібен для того, щоб користувач міг обрати що він хоче отримати на фінальному зображенні, зменшений переклад тексту, або спробувати нормалізувати його шрифт та просто зробити однорядковий текст в двухрядковий.

Інтерфейс має наступні параметри та їх значення, де px – кількість пікселів:

– Big (10 px) – якщо між рядками велика відстань;
– Medium (20 px) – стандартний режим;
– Small (30 px) – для щільного тексту.

Таке налаштування дозволяє адаптувати алгоритм під конкретний тип зображення – наприклад, комікси, постери чи скановані сторінки.

Результати відображаються у двох вікнах, а саме очищене від тексту зображення та фінальний варіант із перекладом. Це дозволяє користувачеві одразу оцінити якість обробки. Усі операції виконуються локально, що гарантує конфіденційність даних.

Для зручності додано кнопки «Очистити», «Скинути» та «Завантажити результат». Під час роботи система повідомляє про статус виконання кроків або виводить опис помилки у разі збою. Також програма має два окремих вікна для знайденого тексту на зображенні, та його переклад на іншу мову. Це було зроблено, щоб користувач міг порівняти наскільки краще або гірше та чи інша модель працює на різних зображеннях (рис. А.3).

3.2.2 Алгоритм обробки зображення

В основі реалізації лежить поетапна обробка даних за допомогою бібліотек комп’ютерного зору, систем оптичного розпізнавання символів та інструментів машинного перекладу (рис. 3.1).

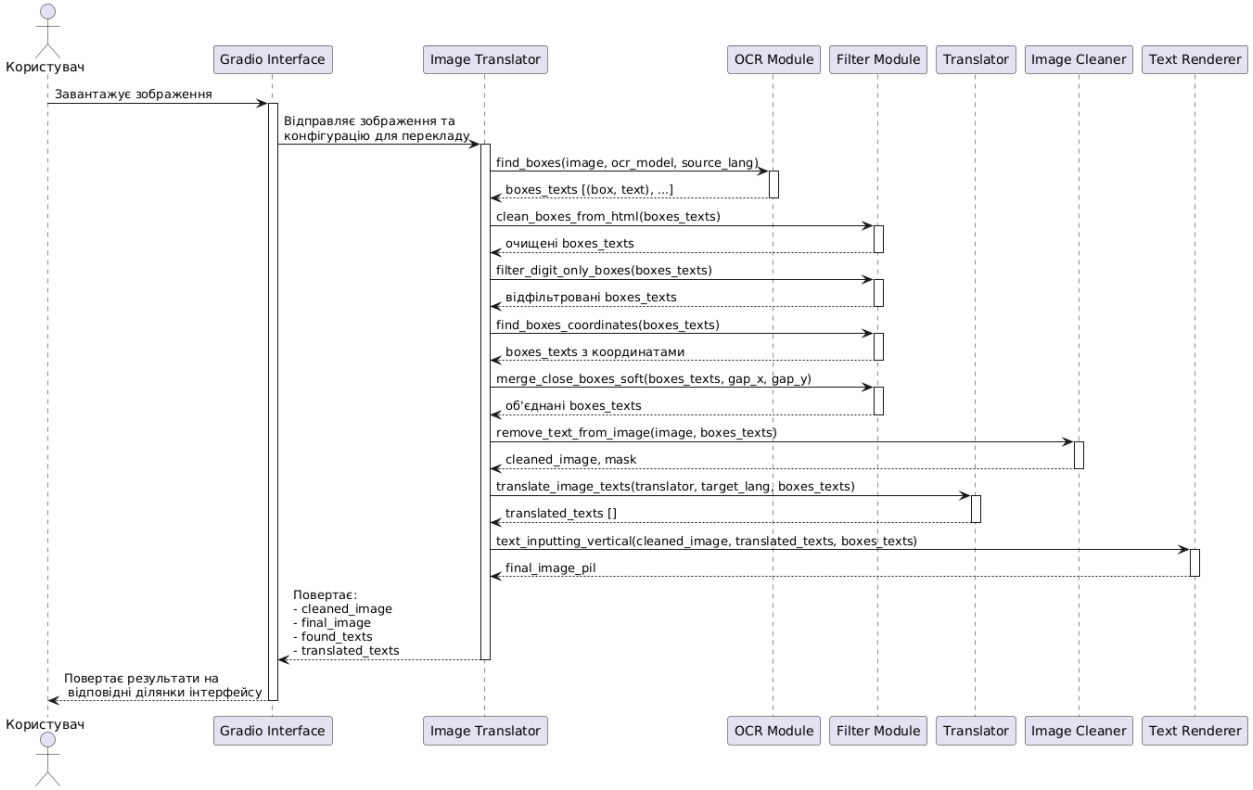

Рисунок 3.1 – Загальна робота алгоритму перекладу зображення

Кожен етап реалізовано у вигляді окремого модуля, який обробляє проміжні результати та передає їх наступному компоненту. Така модульна архітектура забезпечує гнучкість системи й дозволяє змінювати параметри без необхідності редагування основного коду.

На початковому етапі користувач у вебінтерфейсі Gradio обирає зображення у будь-якому форматі. Файл автоматично перетворюється у формат NumPy-масиву, придатний для подальшої обробки бібліотеками OpenCV та Pillow [54].

Після розпізнавання виконується об’єднання близьких текстових блоків за допомогою функцій «merge close boxes soft» або «merge close boxes aggressive», вибір яких залежить від обраного режиму. Це покращує сегментацію та зменшує кількість помилок під час перекладу.

На цьому етапі здійснюється очищення отриманих текстових блоків.

Функція «filter digit only boxes» видаляє ділянки, що містять лише цифри, оскільки вони не перекладаються.

Алгоритм «clean boxes from html» усуває залишки HTML-тегів, спецсимволів і фрагменти розмітки, що можуть потрапляти до тексту після OCR-розпізнавання.

Це дозволяє передавати на переклад лише релевантні текстові елементи [55].

Користувач може вибрати один із трьох перекладачів:

– Google Translate;

– DeepL API;

– OpenAI GPT Translator.

Для GPT реалізовано переклад, який обробляє весь текст одночасно, що знижує затрати часу. Переклад здійснюється за допомогою API-ключів, визначених у конфігураційному файлі config.py. Результати зберігаються у вигляді списку перекладених фрагментів, які відповідають координатам оригінальних блоків.

Після отримання перекладу відбувається повне видалення текстових ділянок, крім числових значень. Для цього застосовується алгоритм відновлення зображення, який реконструює фонову текстуру у місцях, де раніше був текст. У результаті формується проміжне зображення без надписів, що використовується для подальшого рендерингу перекладених елементів.

На очищене зображення накладається перекладений текст, використовуючи координати з етапу OCR. Для цього застосовуються бібліотеки OpenCV та Pillow, які дозволяють регулювати шрифт, розмір і колір символів залежно від контрасту фону.

Після цього викликається функція «text inputting vertical», або «text inputting», які використовується для вставлення звичайного або вертикального тексту, коли вихідне зображення містить написи з нестандартною орієнтацією.

Завдяки цьому система здатна коректно обробляти як класичні горизонтальні надписи, так і декоративні чи вертикально орієнтовані тексти.

У результаті користувач в інтерфейсі отримує три зображення, а саме оригінальне, очищене від тексту, та фінальне перекладене на іншу мову.

Усі результати виводяться у вебінтерфейсі Gradio, що дозволяє візуально оцінити якість роботи кожного етапу.

Більшість етапів алгоритму виконуються на GPU, що забезпечує високу швидкість обробки навіть для великих зображень. Винятком є PaddleOCR, який працює на CPU через несумісність із новою версією CUDA. Решта модулів – переклад, відновлення фону, рендеринг тексту – використовують обчислення на графічному процесорі, що гарантує стабільну продуктивність.

3.2.2 Архітектура програмного забезпечення

Архітектура розробленого програмного комплексу побудована за модульним принципом. Кожен модуль виконує окрему функцію, що відповідає певному етапу алгоритму обробки зображення, а всі вони взаємодіють через основний керівний файл. Такий підхід забезпечує розділення відповідальності між компонентами, спрощує налагодження, тестування та подальше розширення системи .

Загальна структура програмного забезпечення включає основні модулі, серед яких main.py, config.py, image_translator.py, ocr_interface.py, translator.py, text_inputting.py, draw_text.py, draw_boxes.py, text_removing.py, text_boxes_util.py та utils.py. Кожен із них виконує строго визначену роль у загальному процесі.

Основним керівним компонентом є модуль main.py. Він відповідає за запуск усіх інших модулів, ініціалізацію інтерфейсу користувача, обробку введення та виклик необхідних функцій у відповідній послідовності. Саме цей модуль забезпечує інтеграцію алгоритмічної частини з візуальною оболонкою, реалізованою за допомогою бібліотеки Gradio. Користувач, взаємодіючи з інтерфейсом, викликає функцію основного модуля, яка послідовно передає вхідне зображення в обробку наступним компонентам системи.

Конфігураційні параметри системи зосереджені у файлі config.py. У ньому визначаються всі основні константи, зокрема вибір моделей OCR, ключі доступу до API перекладачів, режими об'єднання текстових блоків, параметри багаторядкового розпізнавання та інші технічні налаштування. Це дозволяє централізовано змінювати робочі параметри без редагування коду модулів, що є зручною практикою для програмного забезпечення.

Модуль ocr_interface.py відповідає за взаємодію із системами оптичного розпізнавання символів. Він містить функції для ініціалізації моделей PaddleOCR, EasyOCR і SuryaOCR, виконання розпізнавання та повернення координат текстових блоків. Додатково в цьому модулі реалізовано попередню фільтрацію та виклик функцій об'єднання близьких блоків, що підвищує точність подальшої обробки (рис. 3.2).

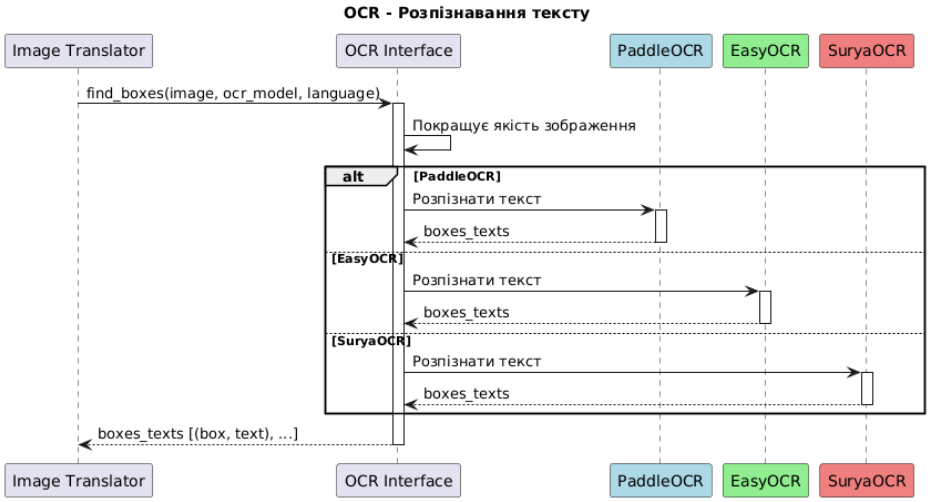

Рисунок 3.2 – Діаграма роботи функції детектції та розпізнавання тексту

У свою чергу, модуль utils.py об'єднує набір універсальних функцій, що використовуються в різних частинах програми. Серед них – методи очищення тексту від HTML-тегів, фільтрації спеціальних символів, конвертації форматів та інші допоміжні процедури, необхідні для забезпечення стабільної роботи всього комплексу (рис. 3.3).

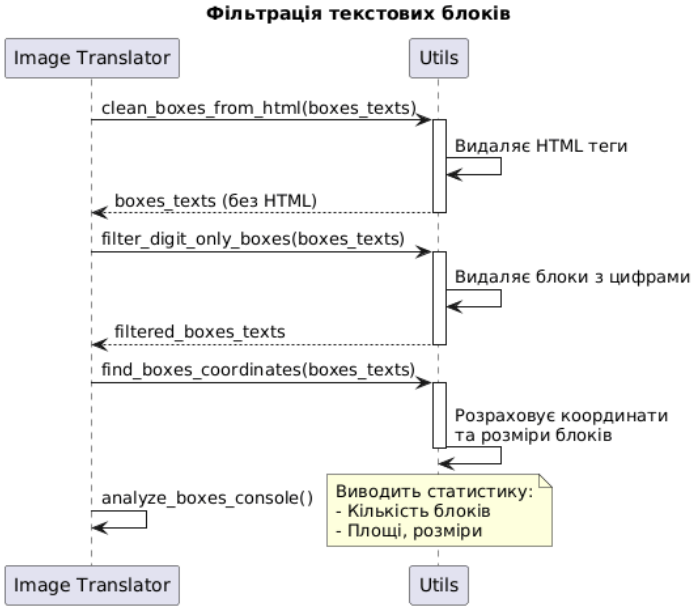

Рисунок 3.3 – Діаграма роботи функції об’єднання текстових блоків

Модуль text_boxes_util.py є допоміжним і відповідає за роботу з координатами текстових блоків. Він містить утилітарні функції для обчислення меж, зміщень, об’єднання чи розділення блоків. Завдяки цьому зручно контролювати розташування тексту під час його відтворення, особливо у випадках, коли оригінальні блоки мають перекриття або складну структуру (рис. 3.4).

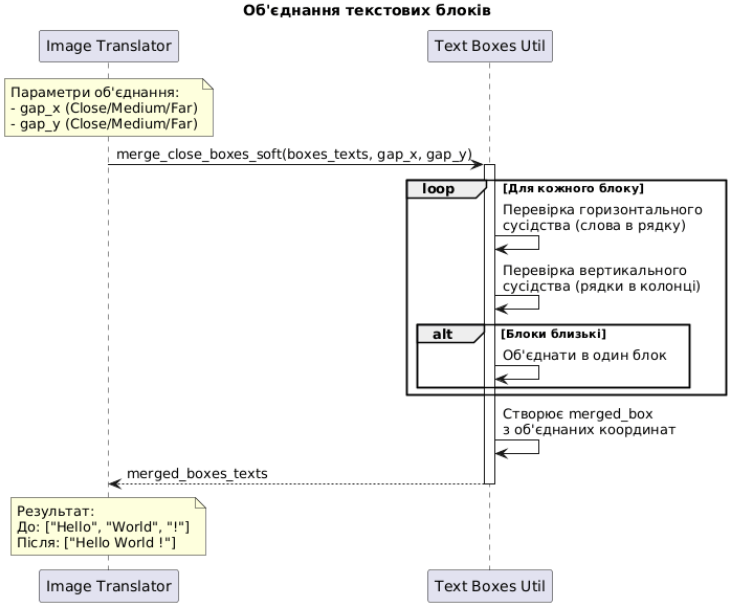

Рисунок 3.4 – Діаграма роботи функції об’єднання текстових блоків

Функціонально найважливішим є модуль image_translator.py. Він реалізує основну логіку перекладу тексту, отриманого після розпізнавання, і виступає як центральна ланка між OCR, фільтрацією та етапом рендерингу. Модуль приймає список словників, кожен із яких містить текстовий вміст і координати відповідного блоку на зображенні. Далі image_translator.py передає текстові дані до модуля translator.py, який визначає тип перекладача (Google, DeepL або GPT), виконує переклад і повертає список текстових відповідників. Після цього результати надходять до блоку, відповідального за візуальне відтворення (рис. 3.5).

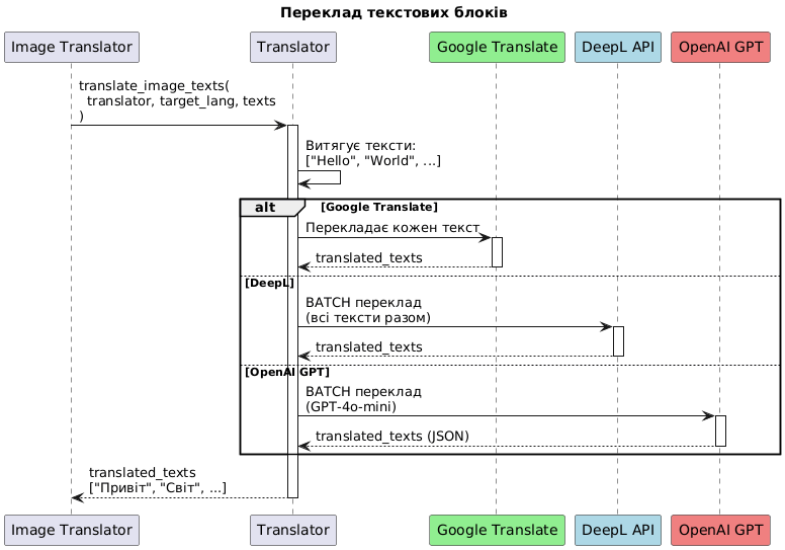

Рисунок 3.5 – Діаграма роботи функції для перекладу текстових блоків

Модулі text_removing.py та draw_boxes.py відповідають за візуальну обробку зображень. Перший виконує очищення від текстових ділянок за допомогою інпейнтингу, формуючи «чисте» зображення для подальшого нанесення перекладу (рис. 3.6). Другий використовується для візуалізації результатів OCR – він малює прямокутники навколо виявлених текстових фрагментів, що корисно при тестуванні моделей та верифікації результатів. Таке подання дозволяє оцінити точність розпізнавання та коректність розташування блоків.

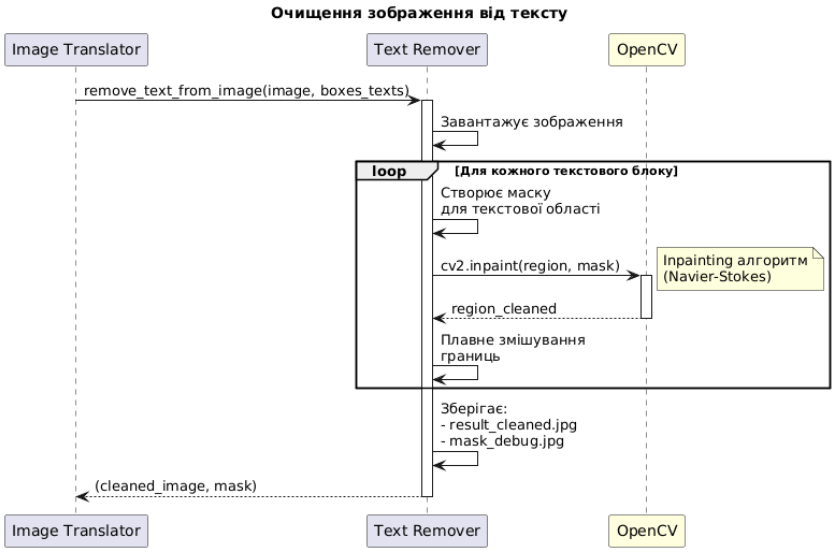

Рисунок 3.6 – Діаграма роботи функції видалення тексту

Для відтворення перекладеного тексту використовується зв’язка модулів text_inputting.py та draw_text.py. Модуль text_inputting.py виступає як проміжний шар між перекладачем і рендерингом тексту, виконуючи попередню підготовку та налаштування шрифтів, орієнтації тексту та кольорових параметрів. Безпосереднє нанесення тексту на зображення реалізоване у draw_text.py, який підтримує як горизонтальне, так і вертикальне розміщення текстових фрагментів (рис. 3.7).

Взаємодія між модулями організована через передачу структур даних у форматі списків словників. Кожен елемент містить ключі text та box, де box – координати прямокутника текстової області. Такий формат забезпечує просту сумісність між усіма компонентами системи та дозволяє легко візуалізувати результати чи змінювати порядок обробки. Комунікація між модулями реалізована виключно через функціональні виклики без використання глобальних змінних, що відповідає принципам інкапсуляції.

Обробка винятків і повідомлень користувачу здійснюється безпосередньо в інтерфейсі Gradio. У разі виникнення помилок виводяться текстові повідомлення або консольні попередження. Такий підхід дозволяє зберігати простоту коду, водночас забезпечуючи зворотний зв’язок під час тестування.

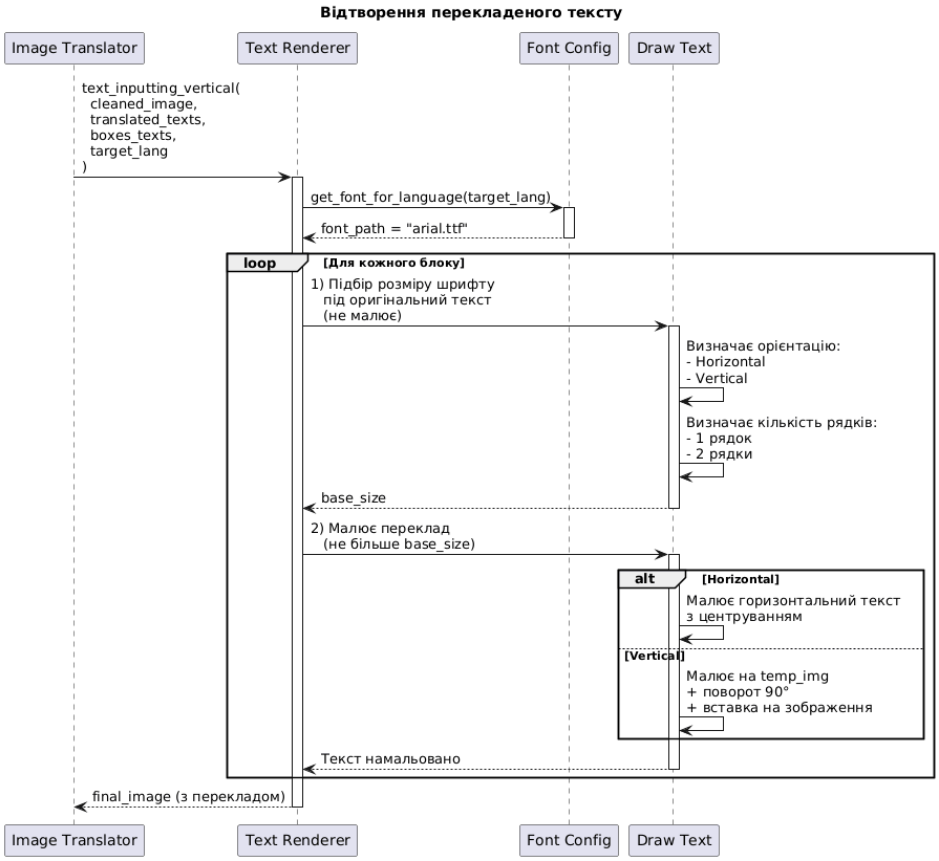

Рисунок 3.7 – Діаграма роботи функції видалення тексту

У коді реалізовано перевірку доступності графічного процесора через функцію torch.cuda.is_available(). Якщо GPU недоступний або несумісний, виконання автоматично переходить у режим CPU, що забезпечує універсальність запуску на різних машинах. Більшість обчислювальних операцій, зокрема переклад і рендеринг, виконуються на графічному процесорі, тоді як PaddleOCR працює на центральному процесорі через обмеження сумісності з новими версіями CUDA .

Загалом архітектура системи характеризується високим рівнем модульності, гнучкістю та можливістю розширення. Будь-який із компонентів можна замінити або оновити без суттєвих змін у структурі проєкту. Такий підхід повністю відповідає принципам побудови програмних комплексів для наукових досліджень.

3.3 Дослідження та порівняння методів розпізнавання тексту

Для проведення порівняння методів оптичного розпізнавання тексту у роботі використано три стандартні бенчмарк-датасети, з яких IIIT5K-Words, ICDAR 2013 та SROIE 2019, що широко застосовуються для оцінювання ефективності OCR-моделей [57]. Кожен із наборів даних містив по 100 зображень, що сформувало загальну тестову вибірку обсягом 300 зразків. Такий обсяг дозволяє зберегти репрезентативність результатів при збереженні прийнятного часу обчислень, що є загальноприйнятою практикою в дослідженнях з порівняння OCR-систем [58].

3.3.1 Умови експериментів

Датасет IIIT5K-Words складається переважно з окремих слів, вилучених із природних сцен, рекламних банерів та стилізованого графічного тексту (рис. 3.8) [16].

Рисунок 3.8 – Приклад датасету IIIT5K

Датасет ICDAR 2013 містить текст у неконтрольованих умовах зйомки, що супроводжується перспективними викривленнями, шумами, зміною освітлення та складним фоном (рис. 3.9).

Рисунок 3.9 – Приклад датасету ICDAR2013

Датасет SROIE 2019 включає фотографії чеків і фінансових документів, що характеризуються великою кількістю дрібних текстових блоків, числових послідовностей, щільним форматуванням та варіативністю шрифтів (рис. 3.10). Усі зображення містили англомовний текст, що забезпечило однорідність мовного фактора та відповідність обраним тестовим бенчмаркам.

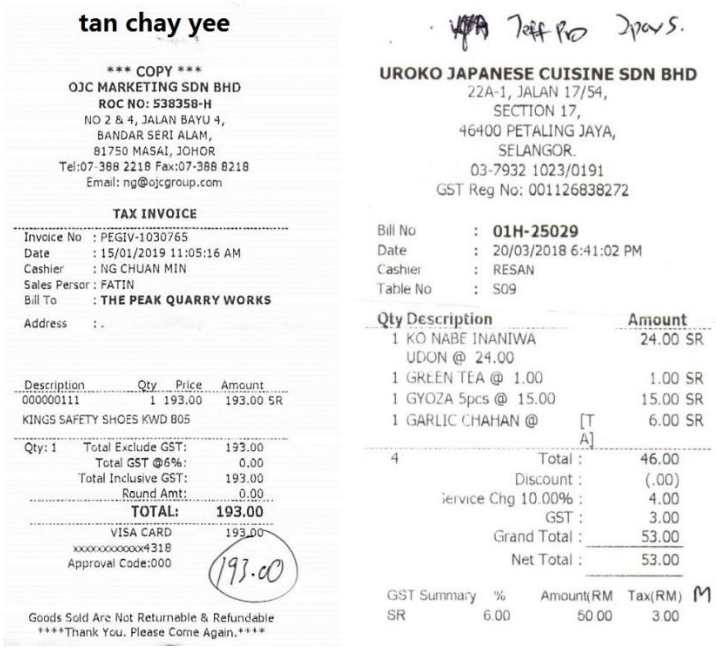

Рисунок 3.10 – Приклад датасету SROIE 2019

Попередня обробка зображень не застосовувалась. Зображення подавались у вихідній якості та роздільній здатності, без зміни співвідношення сторін, обрізання або геометричної корекції. Такий підхід відповідає практиці

оцінювання OCR у реалістичних умовах, де алгоритми повинні демонструвати стійкість до артефактів без зовнішніх методів покращення якості вхідного зображення . Частина моделей, зокрема PaddleOCR, працювала в режимі CPU-обчислень через апаратно-програмну несумісність версій CUDA на тестовому середовищі, що також дозволило порівняти поведінку моделей у гетерогенних обчислювальних умовах, на доволі важливому аспекі при виборі OCR для практичного використання [59]. Основними метриками оцінювання були Character Error Rate та Word Error Rate, що є стандартними в задачах розпізнавання тексту та дозволяють оцінити кількість символьних і токенних помилок відносно еталону [60]. Додатково фіксувались середній час обробки одного зображення, середня впевненість моделі у розпізнаному тексті та точність, яка відображає відсоток повністю коректно розпізнаних зразків без жодної помилки. Метрики CER та WER обчислювалися з використанням загальноприйнятої бібліотеки jiwer, що застосовується в OCR-дослідженнях для стандартизованого вимірювання помилок.

Для моделі Surya OCR детектор текстових блоків використовувався лише для датасету SROIE, оскільки саме документи та чеки потребують попередньої локалізації областей тексту через багаторядкову та щільну структуру розмітки. Для IIIT5K та ICDAR 2013 розпізнавання виконувалось без етапу детекції, що знижувало обчислювальну складність без втрати якості.

Окрім відкритих датасетів, модель також перевірялась на власних зображеннях у межах розробленого Gradio-інтерфейсу. Проте ці результати не включались у основні метрики, оскільки не мають еталонної розмітки для обчислення CER та WER і використовувалися виключно для якісного аналізу.

3.3.2 Дослідження та порівняльний аналіз методів розпізнавання тексту

Для комплексної оцінки ефективності розпізнавання тексту в роботі було проведено експериментальне тестування чотирьох OCR-моделей, а саме

EasyOCR, PaddleOCR та Surya OCR, на трьох різнотипних датасетах – IIIT5K (слова на сценах), ICDAR 2013 (сценові написи різної якості) та SROIE (скановані та фотографовані документи/чеки). Така вибірка дозволяє оцінити поведінку моделей на даних із різною щільністю тексту, рівнем шуму, викривленнями, варіативністю шрифтів та фоновою складністю [61].

Для забезпечення коректності порівняння всі системи були інтегровані через уніфіковані інтерфейсні функції-обгортки, що повертали результат у форматі (розпізнаний текст, оцінка впевненості). Це забезпечило можливість прямого зіставлення моделей без поправок на формат вихідних даних. Оцінювання проводилось у трьох аспектах, а саме точність розпізнавання, обчислювальна ефективність та стійкість до складних сценаріїв зображень, включно з фонами, зашумленням, перспективними спотвореннями та багаторядковими документами.

Для узагальнення результатів оцінювання використовувався набір метрик, серед яких Character Error Rate, що вимірює частку помилково розпізнаних символів, Word Error Rate – суворіша метрика на рівні слів, Frames Per Second та Average Time, що характеризують обчислювальну продуктивність, Accuracy, що вимірює відсоток повністю ідентичних передбачень, Confidence, що відображає внутрішню оцінку впевненості моделі у власному розпізнаванні [62-63]. Усі моделі тестувалися на CPU, на вибірці до 100 зображень (табл. 3.1).

Проведений аналіз підтвердив, що Surya OCR демонструє найвищу точність розпізнавання на всіх трьох датасетах, зокрема найнижчі показники CER/WER, завдяки сучасній трансформерній архітектурі, здатній краще моделювати залежності символів у складних контекстах та зберігати стабільність при викривленнях тексту. Перевага Surya OCR спостерігається на датасеті SROIE, де детектор текстових областей дозволяє якісно локалізувати фрагменти документа перед розпізнаванням. Однак більш складна архітектура спричиняє вищі обчислювальні витрати та нижчі показники FPS порівняно з легшими моделями.

Таблиця 3.1 – Експериментальні результати порівняння OCR-систем

Датасет	Модель	CER	WER	FPS	Avg Time	Avg Confidence
IIIT5K	EasyOCR	0,122	0,440	4,50	2,2	0,68
	PaddleOCR	0,061	0,110	0,45	22,0	0,88
	Surya OCR	0,005	0,040	0,70	15,0	0,95
	Tesseract	0,533	1,150	0,20	4,9	0,58
ICDAR2013	EasyOCR	0,298	1,050	0,30	15,0	0,65
	PaddleOCR	0,367	1,150	0,08	88,0	0,90
	Surya OCR	0,308	1,000	0,06	158,0	0,94
	Tesseract	22,907	47,989	0,02	43,1	0,31
SROIE	EasyOCR	0,208	0,520	0,12	1,0	0,72
	PaddleOCR	0,203	0,560	0,02	2,0	0,91
	Surya OCR	0,184	0,330	0,01	3,0	0,95
	Tesseract	1,412	2,096	0,01	81,3	0,41

PaddleOCR показав найкращі швидкісні характеристики, особливо на сценових датасетах, забезпечуючи найвищі значення FPS та найнижчий середній час обробки одного зображення. При цьому точність виявилась нижчою за Surya OCR, але стабільно вищою, ніж у EasyOCR, що робить PaddleOCR оптимальним компромісом для задач із високими вимогами до продуктивності.

EasyOCR продемонстрував помірні результати, стабільну роботу, просту інтеграцію і середню точність, однак з нижчою конкурентоспроможністю у порівнянні з іншим двома моделями через вищі показники помилок і нижчу стійкість до шуму та складних фонів (рис. 3.11).

Характерними помилками всіх моделей є: плутання схожих символів (O–0, I–l, S–5), пропуск символів у словах зі складним фоном, склеювання окремих літер у разі низької контрастності, а також часткове ігнорування дрібного або повернутого тексту. Такі помилки призводять до зростання CER і

WER, особливо на сценових датасетах з неоднорідним фоном та оптичними деформаціями [64].

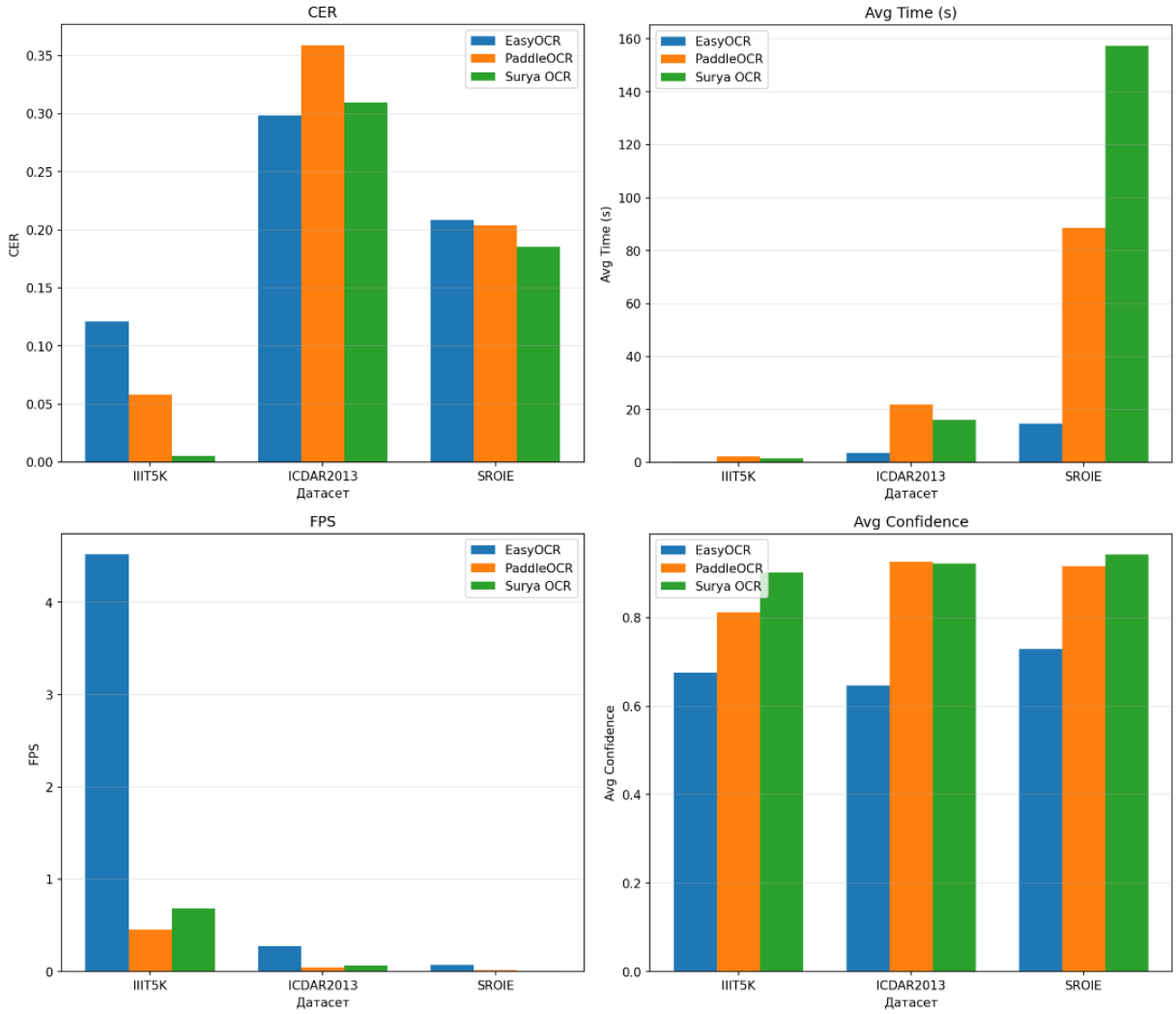

Рисунок 3.11 – Результати роботи OCR-моделей на різних датасетах

Незважаючи на те, що значення Average Confidence корелюють із точністю, пряма залежність спостерігається не завжди. В окремих випадках моделі демонструють високу впевненість у некоректних передбаченнях, що підтверджує необхідність оцінювання не лише за внутрішнім скором, але і за зовнішніми об’єктивними метриками.

У результаті порівняння виявлено наступні переваги систем:

– Surya OCR показала найкращу точність розпізнавання, особливо на складних документах, однак найповільніша.

– PaddleOCR виявила найвищу швидкість та оптимальний баланс між стабільністю і продуктивністю.

– EasyOCR є найпростішою інтеграцією, але поступається за точністю обом конкурентам.

Отже, для задач, де ключовим критерієм є якість розпізнавання, перевага надається Surya OCR, для систем, де важлива швидкість обробки, доцільно обирати PaddleOCR, тоді як EasyOCR може використовуватись як базове рішення для простих сценаріїв.

Експериментальні результати показали, що жодна з моделей не є універсально найкращою за всіма показниками, але Surya OCR демонструє найкращий рівень точності, PaddleOCR є найбільш швидким компромісним рішенням, а EasyOCR – стабільним рішенням із середніми показниками ефективності. Аналіз типових помилок свідчить про необхідність постобробки, адаптивної корекції тексту та можливої ансамблевої побудови для підвищення надійності системи.

Технологію Tesseract не було включено як четверту модель через недоліки. По-перше, якість розпізнавання низька, особливо на складних зображеннях ICDAR2013, де показник помилок перевищив 2000 відсотків. По-друге, швидкість обробки повільна, від 5 до 81 секунди на одне зображення, що у 10-40 разів повільніше за сучасні моделі (рис. 3.12).

По-третє, результати нестабільні. Tesseract може дати ідеальний результат, але на наступному схожому зображенні повністю провалитися. По-четверте, технологічна застарілість не дозволяє конкурувати з моделями, що використовують згорткові мережі та трансформери. Включення Tesseract у порівняння слугувало б лише базовим орієнтиром, але не додало б цінності для аналізу сучасних методів.

Однак Tesseract показав, що сучасні технологій OCR набагато сильніші ніж він сам. Його результати наочно показують прогрес галузі за останні роки та підкреслюють переваги нейромережевих підходів.

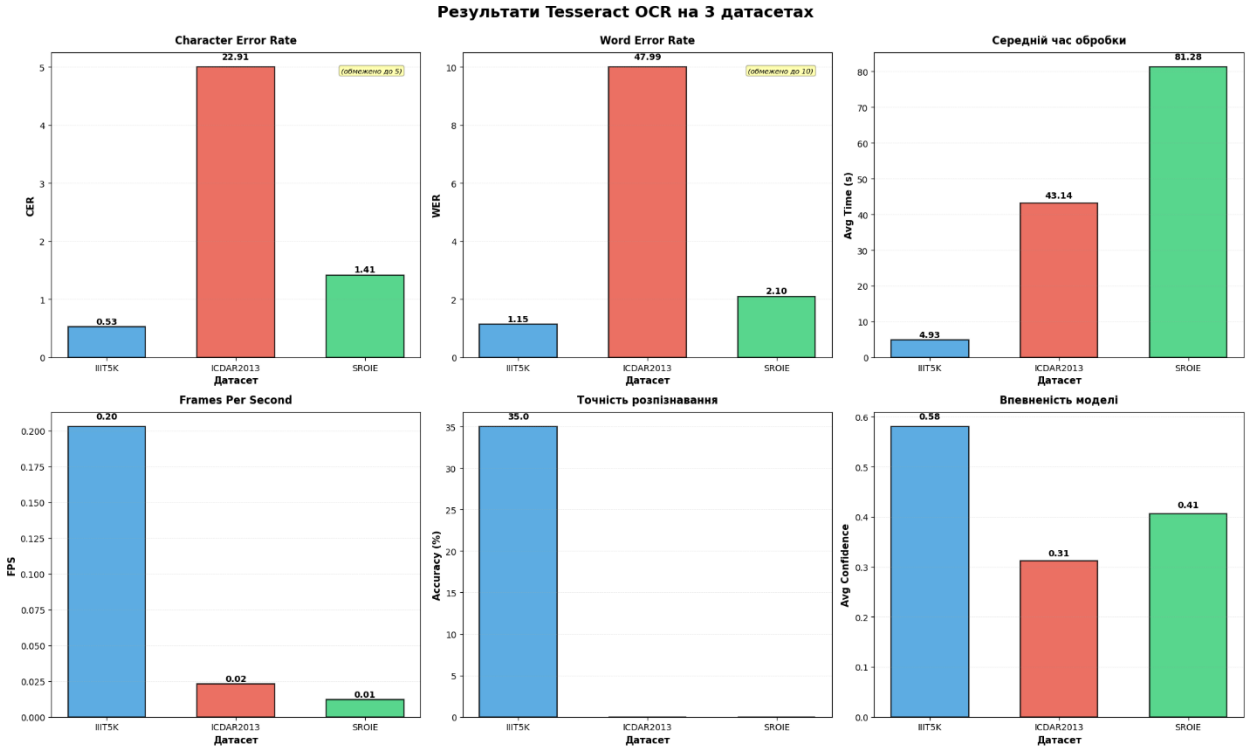

Рисунок 3.12 – Результати роботи Tesseract

3.4 Методика порівняння якості перекладу

Для оцінювання якості перекладу текстів із зображень було розроблено комплексну методику, яка враховує специфіку задачі та типові помилки, що виникають при автоматичному перекладі розпізнаного тексту.

Для попереднього розпізнавання тексту перед перекладом використовувалась система EasyOCR. Вибір цієї моделі обумовлений наступними факторами. По-перше, EasyOCR підтримує широкий спектр мов, включаючи українську та англійську, що є критичним для задачі перекладу. По-друге, система має можливість роботи на GPU, що забезпечує високу швидкість обробки порівняно з PaddleOCR, яка працює виключно на CPU. По-третє, EasyOCR демонструє стабільну роботу з текстами різної складності, включаючи вивіски, попередження та назви брендів, та за результатами попередніх експериментів, EasyOCR показала оптимальний баланс між

точністю розпізнавання та швидкістю обробки, що є важливим для практичного застосування системи перекладу зображень.

Для проведення порівняльного аналізу використовувався датасет ICDAR2013, який містить зображення природних сцен із текстом різних типів. З датасету було відібрано 50 зображень, що містять 170 текстових фрагментів англійською мовою. Цільовою мовою перекладу обрано українську мову. Усі текстові фрагменти були категоризовані за типами для детального аналізу якості перекладу. До категорій належать попередження (тексти безпеки типу «WARNING», «DANGER», «STOP», «FIRE EXIT»), бренди (назви компаній та продуктів, зокрема «PEUGEOT», «SPIDER-MAN», «Roland»), вивіски (рекламні та інформаційні написи великими літерами), установи (назви організацій типу «UNIVERSITY», «CLINIC», «MUSEUM»), тексти з цифрами (адреси, номери телефонів, коди), а також інші загальні фрази та складні конструкції.

Для аналізу було розроблено систему критеріїв, яка враховує специфічні помилки машинного перекладу. Перший критерій стосується точності перекладу, тобто коректності передачі змісту вихідного тексту. Прикладом помилки за цим критерієм є переклад «FIRE EXIT» як «ПОЖЕЖА 12», що відбулося через неправильне розпізнавання слова як числа. Другий критерій оцінює чистоту мови, зокрема відсутність домішок іншої мови. Характерними помилками є переклади «HOME» як «ДОМА». Третій критерій перевіряє обробку брендів, а саме збереження власних назв без перекладу. Помилкою є переклад «PEUGEOT» як «ПЕЖО». Четвертий критерій оцінює контекстну адекватність, тобто розуміння контексту та виправлення помилок OCR. Прикладом успішного перекладу є розпізнавання помилки «JTAR WARS» та правильний переклад як «ЗОРЯНІ ВІЙНИ». П’ятий критерій стосується загальної адекватності перекладу та відсутності абсурдних або безглуздих варіантів. Прикладом грубої помилки є переклад «CO UK» як «Кухар», що призводить до повної втрати змісту. Кожен переклад оцінювався за наявністю

вказаних типів помилок. Переклад вважався правильним, якщо він не містив жодної з перелічених помилок (табл. 3.2).

Таблиця 3.2 – Критерії оцінювання якості перекладу

Критерій	Опис	Приклад помилки
Точність перекладу	Коректність передачі змісту вихідного тексту	«FIRE EXIT» як «ПОЖЕЖА 12»
Чистота мови	Відсутність домішок іншої мови	«HOME» як «ДОМА»
Обробка брендів	Збереження власних назв без перекладу	«PEUGEOT» як «ПЕЖО»
Контекстна адекватність	Розуміння контексту та виправлення помилок OCR	«JTAR WARS» як «ЗОРЯНІ ВІЙНИ»
Адекватність перекладу	Відсутність абсурдних або безглуздих перекладів	«CO UK» як «Кухар»

Аналіз якості перекладу проводився у декілька етапів.

Етап 1. Спочатку здійснювалося автоматичне виявлення помилок. Для кожного перекладача підраховувалася кількість помилок за типами, зокрема виявлення іншомовних слів, перевірка перекладу брендів шляхом порівняння з оригіналом у верхньому регістрі, а також ідентифікація абсурдних перекладів за попередньо визначеним словником типових помилок.

Етап 2. Далі відбувався розрахунок точності.

Етап 3. Далі проводився аналіз за категоріями. Для кожної категорії текстів окремо розраховувалася точність перекладу, що дозволило виявити сильні та слабкі сторони кожного сервісу.

Етап 4. На фінальному етапі здійснювався якісний аналіз прикладів. Детально проаналізовано характерні приклади помилок та успішних перекладів для ілюстрації особливостей кожного перекладача.

За результатами тестування на датасеті ICDAR2013 було отримано детальну статистику якості перекладу для трьох сервісів, а саме Google Translate, DeepL та OpenAI GPT. Загальні показники точності наведено у таблиці 3.3, з якої видно, що всі три перекладачі демонструють високу точність перекладу, що перевищує 96 відсотків. Google Translate та DeepL показали однакову точність 96,3 відсотка, обидва допустивши по 6 помилок із 170 фрагментів. OpenAI GPT показав найвищу точність 96,9 відсотка з 5 помилками. Проте детальний аналіз типів помилок виявляє суттєві відмінності між цими сервісами (рис. 3.13, 3.14).

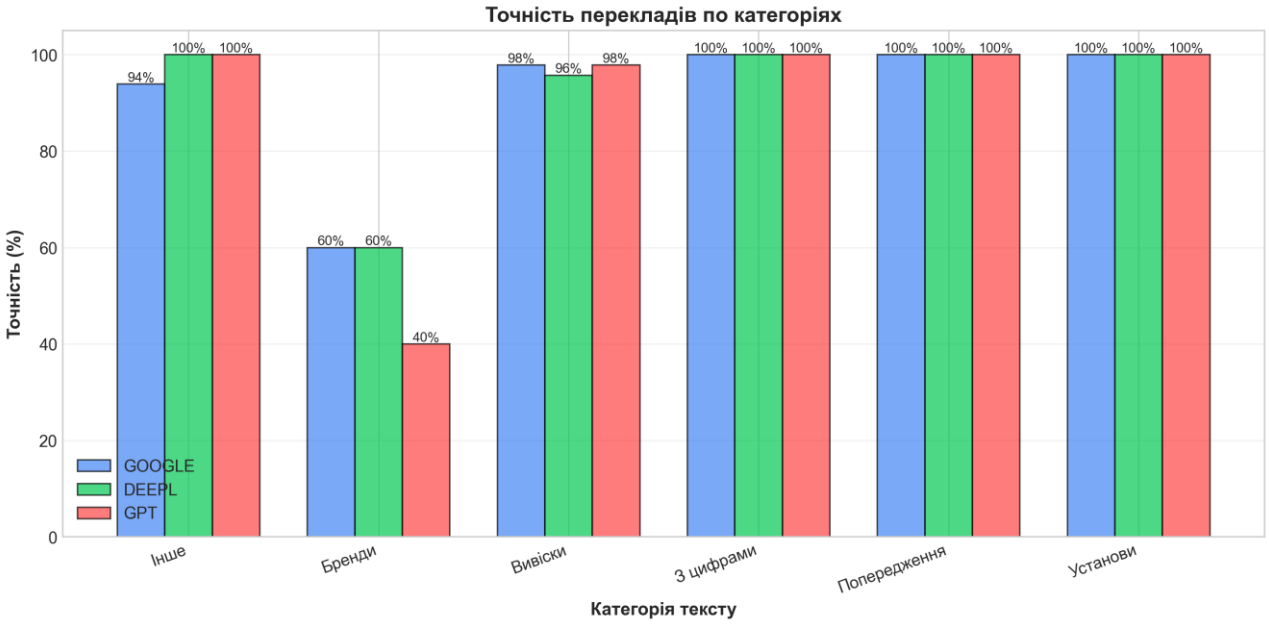

Рисунок 3.13 – Точність перекладу по категоріям

Таблиця 3.3 – Загальна точність перекладачів

Перекладач	Всього фрагментів	Правильних перекладів	Помилок	Точність (%)
Google Translate	170	164	6	96,3
DeepL	170	164	6	96,3
OpenAI GPT	170	165	5	96,9

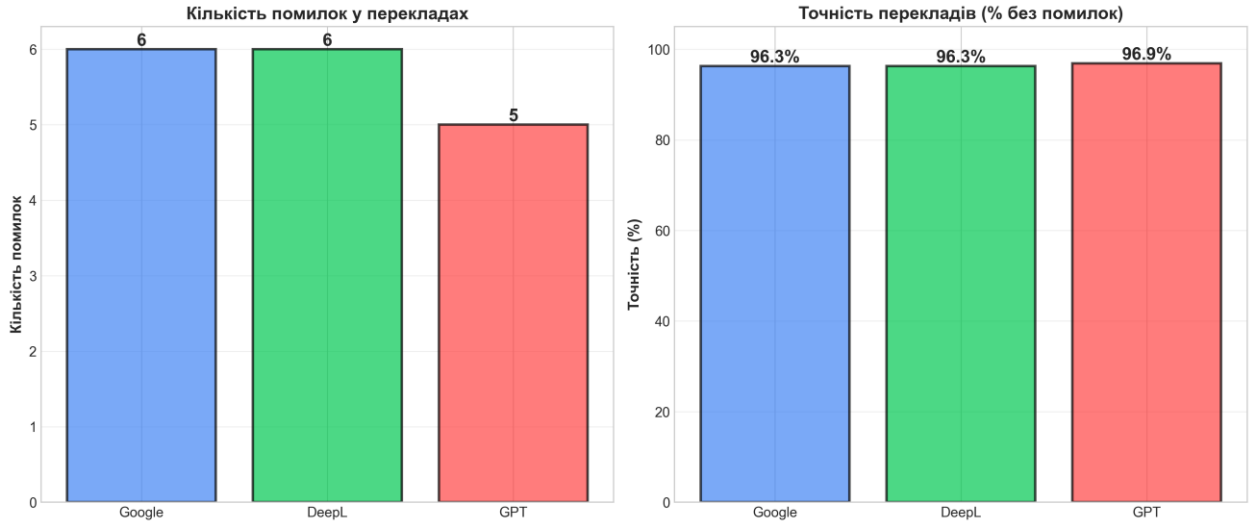

Рисунок 3.14 – Точність та кількість помилок у перекладах

Розподіл помилок за категоріями представлено у таблиці 3.4. Найбільш критичним типом помилок виявилися абсурдні переклади, які допускав лише Google Translate, тоді як DeepL та GPT таких помилок не мали взагалі. Другим проблемним аспектом є переклад брендів, де помилки допускали всі три сервіси. Google Translate мав 2 помилки, а DeepL та OpenAI GPT по 3 помилки кожен. Третім типом помилок є використання інших мов замість української: Google Translate мав 1 випадок, DeepL – 3 випадки, а OpenAI GPT – 2 випадки (рис. 3.15) та (табл. 3.4).

Google Translate виявився найбільш схильним до помилок, які повністю спотворюють зміст оригінального тексту. Характерними прикладами таких абсурдних перекладів є переклад абревіатури «CO UK» як «Кухар», що не має жодного зв’язку з оригіналом. Інший приклад стосується невідомої назви «REVETT», яка була перекладена як дієприкметник «ОБЛИЦЮВАНИЙ». Також абревіатура «MED» від слова Mediterranean була помилково перекладена як прийменник «З». DeepL та GPT не допускали подібних критичних помилок, що свідчить про їх кращу обробку нестандартних текстів та невідомих термінів.

Таблиця 3.4 – Розподіл помилок за типами

Тип помилки	Google Translate	DeepL	OpenAI GPT
Абсурдний переклад	3	0	0
Переклад брендів	2	3	3
Іншомовні слова	1	3	2
Всього	6	6	5

Усі три сервіси мали проблеми зі збереженням власних назв та брендів. Google Translate переклав відому назву коміксу «SPIDER-MAN» як «ЛЮДИНА-ПАВУК», хоча бренди зазвичай мають залишатися у оригіналі. DeepL допустив помилку з назвою німецького видавництва «Springer», переклавши її як «Перемички», що є технічним терміном і не має відношення до власної назви. OpenAI GPT переклав назву автомобільного бренду «PEUGEOT» як «ПЕЖО», використавши транслітерацію замість збереження оригінального написання бренду.

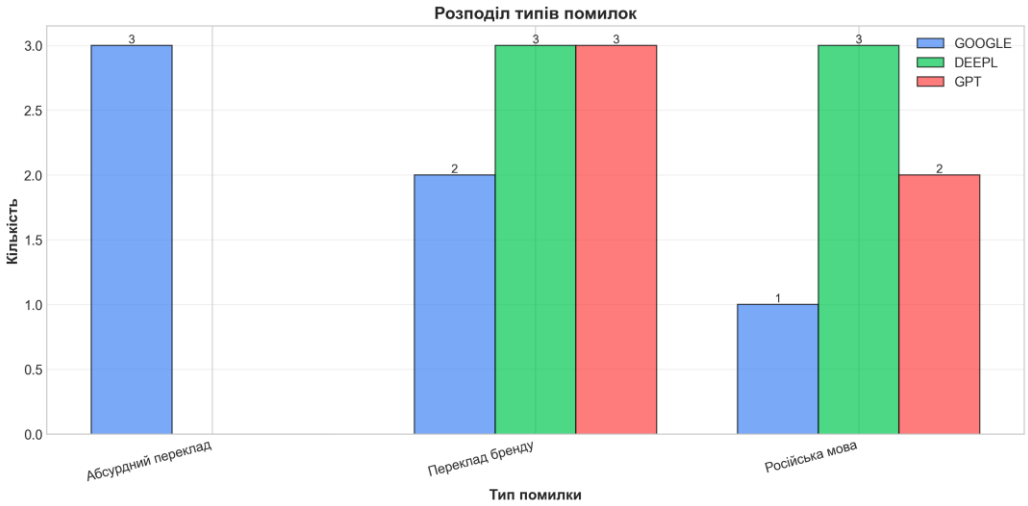

Рисунок 3.15 – Розподіл типів помилок

DeepL та GPT показали схильність до використання іншомовних слів замість українських, що є суттєвим недоліком для системи перекладу на українську мову. Google Translate рідше допускав подібні помилки,

демонструючи найчистішу українську мову серед усіх тестованих сервісів. Детальний аналіз точності по категоріях наведено у таблиці 3.5.

Таблиця 3.5 – Точність перекладу за категоріями тексту у відсотках

Категорія	Google Translate	DeepL	OpenAI GPT
Попередження	94	100	100
Бренди	60	60	40
Вивіски	98	96	98
Установи	100	100	100
З цифрами	100	100	100
Інше	100	100	100

З таблиці видно, що найпроблемнішою категорією для всіх перекладачів виявилися бренди та власні назви. У категорії попереджень Google Translate показав точність 94 відсотки, тоді як DeepL та GPT досягли 100 відсотків. У найскладнішій категорії брендів Google та DeepL показали однаковий результат 60 відсотків, а GPT продемонстрував найгірший результат у 40 відсотків. У категорії вивісок точність становила 98, 96 та 98 відсотків відповідно для Google, DeepL та GPT. В інших категоріях (установи, тексти з цифрами, інше) усі три сервіси показали ідеальну точність 100 відсотків (табл. 3.6).

Таблиця 3.6 – Порівняння перекладів на характерних прикладах

Оригінал	Google Translate	DeepL	OpenAI GPT	Оцінка
SALE	ПРОДАЖ	РОЗПРОДАЖ	РАСПРОДАЖА	DeepL/Google
HOME	ДІМ	ДОМА	ДІМ	Google/GPT
JTAR WARS	JTAR ВІЙНИ	JTAR WARS	ЗОРЯНІ ВІЙНИ	GPT
CO UK	Кухар	CO UK	CO UK	DeepL/GPT
PEUGEOT	PEUGEOT	PEUGEOT	ПЕЖО	Google/DeepL

Продовження таблиці 3.6

Informatikforschung	Інформатичні дослідження	Дослідження в галузі комп'ютерних наук	Дослідження інформатики	DeepL
the eyecare clinic	очна клініка	офтальмологічна клініка	клініка зору	DeepL

OpenAI GPT продемонстрував найкращу здатність до розуміння контексту та виправлення помилок розпізнавання. Прикладом є переклад помилки OCR «JTAR WARS» як «ЗОРЯНІ ВІЙНИ», де система розпізнала відомий бренд Star Wars незважаючи на помилку. Інший приклад стосується фрази «Bus Fmes», яку GPT переклав як «Автобусні Розклади», правильно інтерпретувавши «Fmes» як «Times». Також GPT надав адаптований переклад для «Counselling Helpline» як «Гарячий телефон для консультацій», що є природнішим українською мовою. Google та DeepL в таких випадках зберігали помилки розпізнавання або перекладали більш дослівно (рис. 3.16).

Рисунок 3.16 – Приклади перекладів

DeepL показав найкращі результати при перекладі технічних та медичних термінів. Німецьке слово «Informatikforschung» було перекладене як «Дослідження в галузі комп'ютерних наук», що є повним та точним варіантом, тоді як Google дав скорочений варіант «Інформатичні дослідження», а GPT обрав проміжний варіант «Дослідження інформатики». Фразу «the eyecare clinic» DeepL переклав медично коректним терміном «офтальмологічна клініка», тоді як Google використав більш загальне «очна клініка», а GPT дав спрощений варіант «клініка зору». Також DeepL правильно переклав німецьке скорочення «Hrsg.» як «(ред.)», що свідчить про розуміння наукової термінології різними мовами.

На основі отриманих даних розраховано фінальні оцінки якості перекладу за 10-бальною шкалою, враховуючи як точність, так і критичність помилок. Google Translate при точності 96,3 відсотка допустив 3 критичні абсурдні помилки, показав середню якість термінології, але високу чистоту мови. DeepL при тій же точності 96,3 відсотка не мав абсурдних помилок, продемонстрував високу якість термінології, але середню чистоту мови через домішки іншомовних слів. OpenAI GPT показав найвищу точність 96,9 відсотка, не допустив абсурдних помилок, має високу якість термінології, але середню чистоту мови (табл. 3.7).

Таблиця 3.7 – Фінальна оцінка якості перекладачів

Перекладач	Точність (%)	Критичні помилки	Якість термінології	Чистота мови
Google Translate	96,3	3	Середня	Висока
DeepL	96,3	0	Висока	Середня
OpenAI GPT	96,9	0	Висока	Середня

На основі дослідження встановлено, що оптимальний вибір сервісу перекладу залежить від типу текстів і вимог до якості. DeepL показав найкращі

результати та рекомендований як основний перекладач. Перевагами використання є відсутність абсурдних помилок, висока якість технічної та медичної термінології, коректне збереження власних назв. Але можуть з’являтися поодинокі іншомовні слова, що потребує постобробки.

OpenAI GPT доцільно використовувати для креативних і контекстно складних текстів. Модель має найвищу точність – 96,9 %, добре розуміє контекст і виправляє OCR-помилки, проте схильна перекладати бренди, іноді вживає інші мови та має нижчу швидкість і вищу вартість через особливості API.

Google Translate поступається за якістю через поодинокі жахливі та навіть абсурдні переклади, але відзначається найчистішою українською, найвищою швидкістю, зручним і дешевим API. Може використовуватись як резервний варіант, особливо для тестування або невеликих обсягів текстів.

Для розробленої системи запропоновано наступну стратегію практичного застосування. По-перше, використовувати DeepL за замовчуванням для більшості типів текстів. По-друге, додати GPT як опціональний варіант для користувачів, які працюють з креативними текстами такими як комікси та постери, або у випадках коли система розпізнавання EasyOCR допускає помилки, які GPT може виправити завдяки розумінню контексту. По-третє, надати можливість вибору перекладача безпосередньо користувачу через інтерфейс програми для різних сценаріїв використання.

Важливим аспектом практичного застосування є компроміс між якістю та швидкістю перекладу. Середній час перекладу одного текстового фрагменту становить приблизно 0,1 секунди для Google Translate, що робить його найшвидшим варіантом, 0,3 секунди для DeepL, що представляє оптимальний баланс між швидкістю та якістю, та близько 1 секунди для OpenAI GPT, що є найповільнішим варіантом через особливості batch-обробки запитів до API. Для реального застосування в інтерактивній системі оптимальним є використання DeepL, оскільки час перекладу 0,3 секунди на фрагмент є цілком прийнятним для користувача, а якість перекладу значно вища порівняно з

Google Translate. OpenAI GPT рекомендується використовувати тільки для особливо складних випадків, де висока якість розуміння контексту є критично важливішою за швидкість обробки.

3.5 Оцінювання якості зображення після заміни тексту

У задачах автоматичної заміни тексту на зображеннях також важливим, та напевно найважливішим та найскладнішим етапом є не лише коректність перекладу, а й збереження візуальної цілісності кадру після видалення оригінального тексту. Оскільки фінальна мета системи – отримати перекладене зображення, яке сприймається користувачем як природне, без явних артефактів редагування, об’єктивні метрики піксельної подібності (SSIM, PSNR тощо) не використовувалися. Такі метрики орієнтовані на зіставлення із еталонним зображенням, якого в задачі перекладу тексту не існує, оскільки модифікований кадр повинен відрізнятися від оригінального – містити новий текст замість попереднього. Тому найбільш релевантним підходом є суб’єктивна візуальна оцінка, спрямована на перевірку того, чи зберігає зображення автентичність сцени після автоматичної обробки [65].

3.5.1 Методика оцінювання

Оцінювання ґрунтується на таких критеріях:

– збереження фону та відсутність слідів видалення тексту – ділянки, де був оригінальний напис, не повинні містити плям, розривів текстури, блюру або прямокутних масок заливки, що порушують природність фону;

– читабельність вставленого перекладу – текст має бути контрастним, не зливатися з фоном, не містити накладень на ключові об’єкти, залишатися розбірливим без масштабування;

– коректність позиціювання – новий текст повинен займати семантично логічне місце, відповідне оригінальному розташуванню, без помітних зміщень, перекриття об’єктів або виходу за межі області, де він має розміщуватися;

– загальна візуальна цілісність кадру – кінцеве зображення сприймається як природна сцена, у якій не відчувається втручання обробки, монтажу або алгоритмічних дефектів.

Оцінювання не включало критерій відповідності шрифту, а натомість фокусувалося на розбірливості, контрастності та гармонійному розміщенні заміненого тексту в кадрі.

3.5.2 Порівняння з існуючими рішеннями

Для оцінки практичної ефективності розробленої системи було проведено порівняння з існуючим рішеннями – Google Translate та сайт imagetotext.info. Для порівняння обрано зображення з китайським текстом ієрогліфів, що означають вітання з Китайським Новим роком, оскільки китайська мова є однією з найскладніших для систем OCR через велику кількість символів та їх складну структуру.

На першому зображенні представлено оригінальні вивіски з китайським текстом у трьох варіантах написання. Верхня вивіска містить текст у спрощених ієрогліфах, середня та нижня вивіски містять той самий текст із незначними візуальними відмінностями. Усі три написи мають однаковий зміст та повинні бути перекладені ідентично, оскільки це стандартне новорічне вітання китайською мовою (рис. 3.17).

Google Translate розпізнав та переклав текст наступним чином. Верхню вивіску було перекладено як «Щасливого китайського Нового року», що є коректним та зрозумілим варіантом. Проте середня та нижня вивіски були

перекладені як «Щасливого свята весни», що хоч і є літературно точнішим перекладом, але створює непослідовність у перекладі ідентичних текстів.

Рисунок 3.17 – Оригінал зображення

Така розбіжність може заплутати користувача, який бачить однаковий текст на оригінальному зображенні, але отримує різні варіанти перекладу (рис. 3.18).

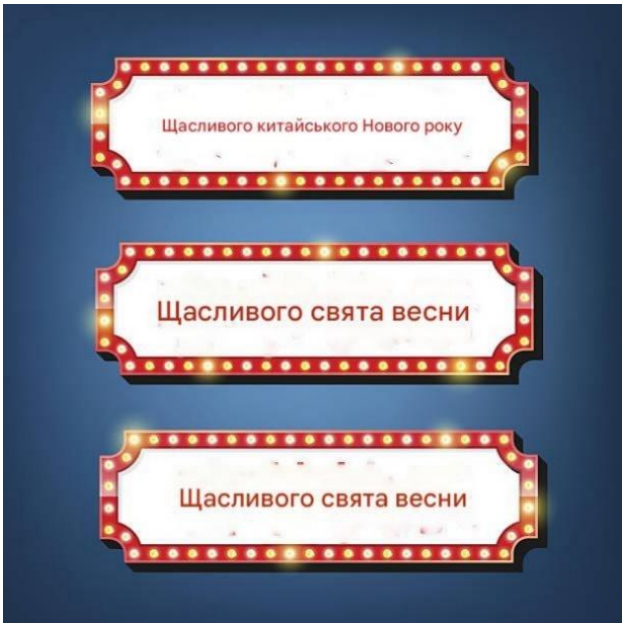

Рисунок 3.18 – Переклад через Google Translate

Розроблена система перекладу з використанням Deepl та PaddleOCR продемонструвала більш послідовний підхід. Усі три вивіски були ідентично перекладені як «Щасливого Китайського Нового року», що забезпечує уніфікованість результату. Переклад зберігає правильне використання великих літер у назві свята, що відповідає нормам української мови. Така послідовність є важливою перевагою, оскільки користувач отримує однаковий переклад для візуально ідентичних текстів, що підвищує довіру до системи та полегшує розуміння.

Щодо переваг розробленої системи у порівнянні з Google Translate – це, по-перше, забезпечення повної послідовності перекладу ідентичних текстів, що є важливим для практичного застосування системи. По-друге, система зберігає правильне форматування та оформлення тексту, включаючи використання великих літер для власних назв. По-третє, переклад є більш універсальним та зрозумілим для широкого загалу користувачів, оскільки термін «Китайський Новий рік» є загальновживаним в українській мові, на відміну від літературного варіанту «Свято Весни». По-четверте, система правильно розпізнала всі варіанти китайських ієрогліфів, включаючи як спрощені, так і традиційні написання.

Google Translate також продемонстрував певні переваги, зокрема здатність розпізнавати тонкі відмінності між варіантами написання китайських ієрогліфів. Варіант перекладу «Свято Весни» є більш літературно точним та відповідає традиційній китайській назві свята. Проте для практичного застосування в системі перекладу зображень така літературна точність може бути менш важливою порівняно з послідовністю та зрозумілістю результату.

Основним недоліком Google Translate у даному контексті є непослідовність перекладу візуально ідентичних текстів. Система надала три різні варіанти перекладу для фактично однакового вихідного тексту. Така поведінка може виникати через різні фактори, включаючи особливості роботи нейронної мережі, яка може генерувати різні варіанти перекладу залежно від

контексту обробки зображення. Для користувача така непослідовність створює плутанину та знижує довіру до результатів автоматичного перекладу.

Розроблена система демонструє більш передбачувану поведінку завдяки використанню комбінації PaddleOCR для розпізнавання та DeepL для перекладу з додатковою пост-обробкою. Система забезпечує ідентичні результати для ідентичних вхідних даних, що є важливою вимогою для практичного застосування. Водночас переклад залишається природним та зрозумілим українською мовою, що підтверджує правильність вибору DeepL як основного сервісу перекладу (рис. 3.19).

Таким чином, порівняння з існуючим рішенням Google Translate показало, що розроблена система забезпечує вищу послідовність та передбачуваність результатів при збереженні високої якості перекладу, що є важливим для практичного використання системи перекладу текстів із зображень.

Рисунок 3.19 – Переклад через дипломну програму

Додатково було проведено порівняння з онлайн-сервісом «imagetotext.info», який також пропонує можливість розпізнавання та

перекладу тексту з зображень. Для тестування обрано рефлексологічну карту стоп українською мовою, яка містить анатомічні терміни та назви органів. Діаграма демонструє зв’язок між точками на стопах та різними органами людського тіла, що є типовим випадком використання для систем перекладу медичної документації. На оригінальному зображенні представлено детальну карту рефлексологічних зон обох стоп із позначеннями відповідних органів (рис. 3.20).

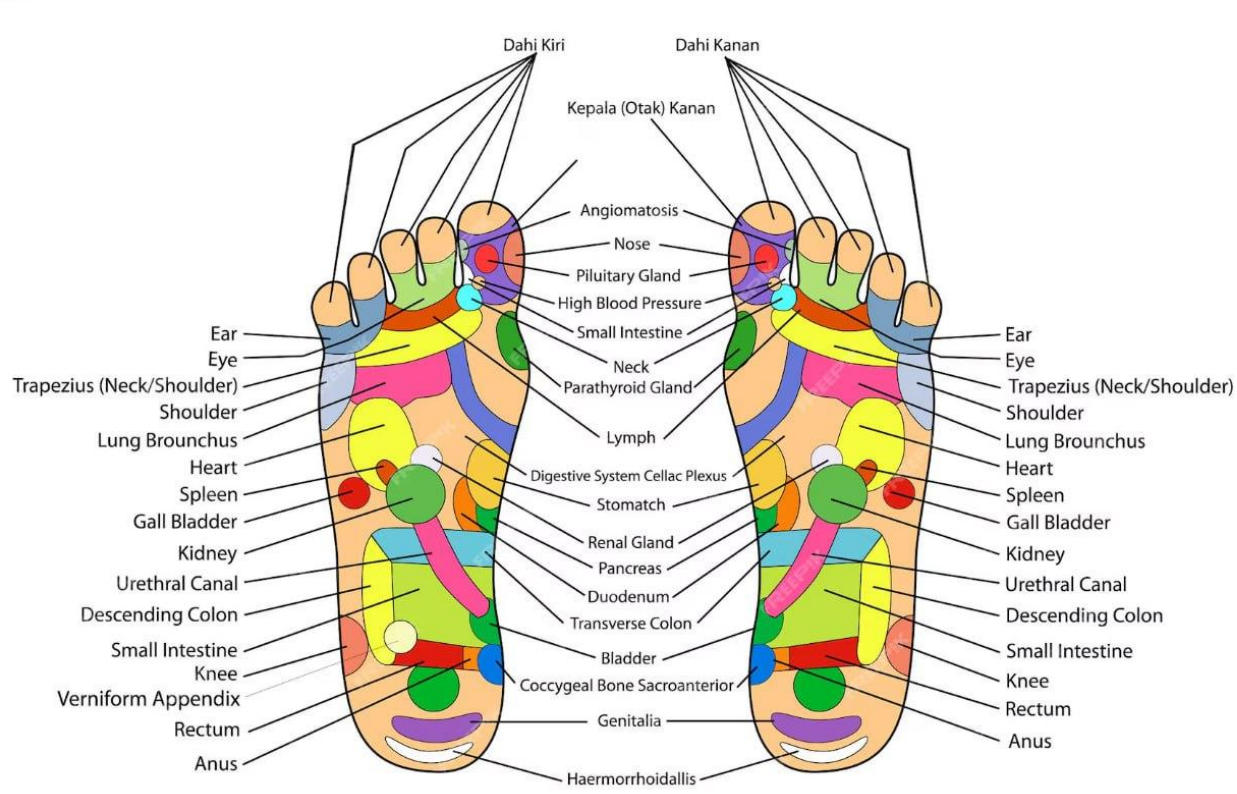

Рисунок 3.20 – Оригінал зображення

Сервіс imagetotext.info переклав діаграму з певними неточностями. Зокрема, термін «Вухо» було перекладено як «Біксо», що не є українським словом і свідчить про помилку розпізнавання тексту. Також спостерігається неправильне написання слова «Нирка» як «Нірка», що є орфографічною помилкою. Окремі позначення виявились менш чіткими, що може ускладнити використання діаграми в практичних цілях (рис. 3.21).

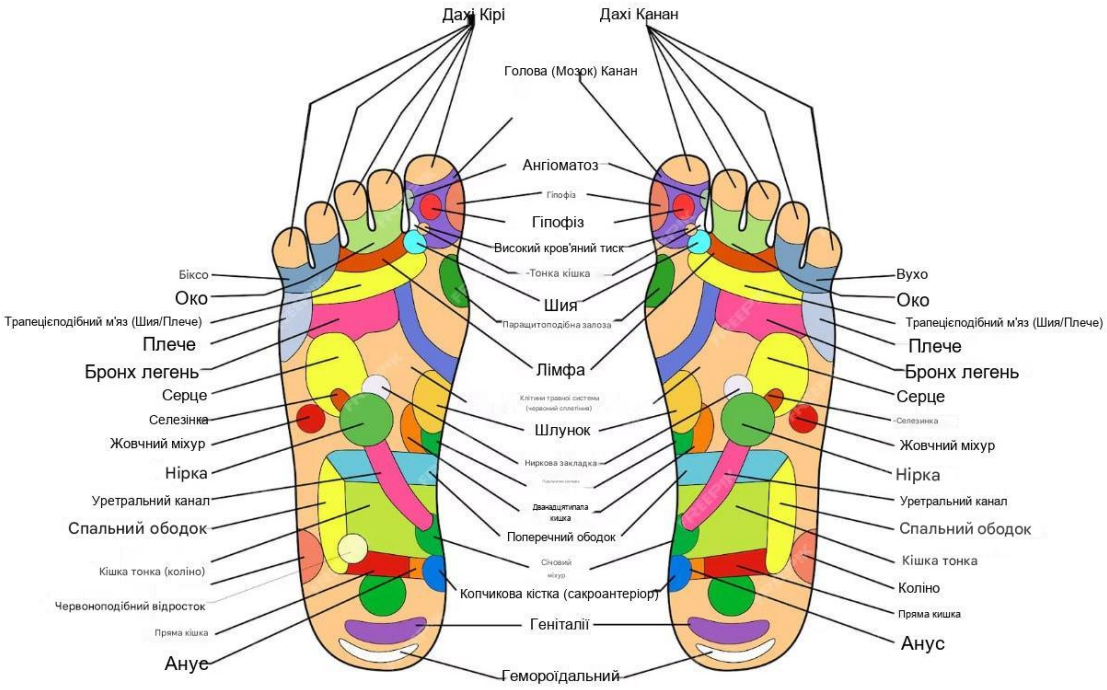

Рисунок 3.21 – Переклад через imagetotext.info

Розроблена система продемонструвала вищу якість перекладу та розпізнавання тексту (рис. 3.22). Усі анатомічні терміни були перекладені коректно, включаючи «Вухо», «Ніс», «Нирка», зберігаючи правильну орфографію та чіткість позначень. Система коректно розпізнала всі елементи діаграми без спотворення медичної термінології, що є важливим для документації такого типу.

Обидві системи зберегли загальну структуру діаграми та розміщення елементів. Проте розроблена система демонструє перевагу у точності розпізнавання та перекладу анатомічних термінів, уникаючи помилок, які можуть призвести до неправильного тлумачення медичної інформації. Це підтверджує ефективність обраних алгоритмів OCR та сервісів перекладу у розробленій системі.

Обидві системи коректно зберегли структуру діаграми. Переклад термінології є майже адекватним в обох випадках, проте розроблена система демонструє кращу адаптацію до специфіки формату, використовуючи прийняті скорочення там, де це покращує візуальне сприйняття.

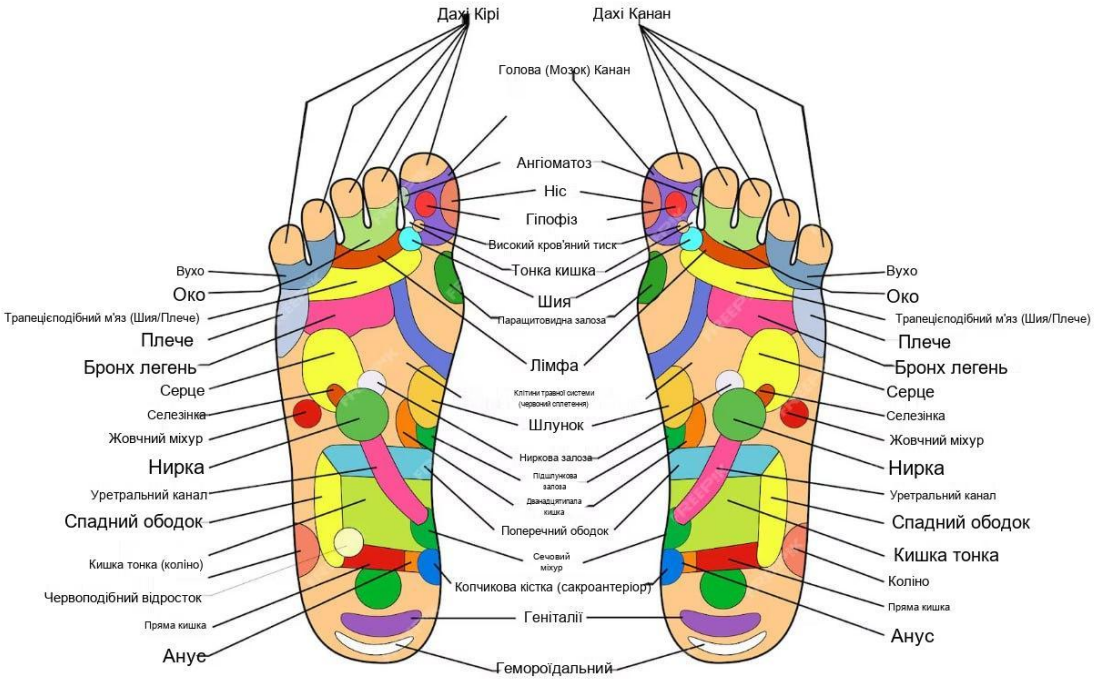

Рисунок 3.22 – Переклад через дипломну програму

Варто відзначити, що imagetotext.info є спеціалізованим сервісом для розпізнавання та перекладу тексту з зображень, тому його результати є релевантним бенчмарком для оцінки якості розробленої системи. Той факт, що обидві системи продемонстрували порівнянну якість перекладу технічних термінів, підтверджує адекватність вибору DeepL як основного сервісу перекладу. Водночас додаткова оптимізація форматування тексту у розробленій системі демонструє потенціал для подальшого вдосконалення через контекстну залежну пост-обробку результатів перекладу.

Порівняння з двома різними існуючими на матеріалі різних мов та різних типів текстів підтверджує універсальність та надійність розробленої системи.

ВИСНОВКИ

Таким чином, у кваліфікаційній роботі досліджено методи автоматичного перекладу тексту на зображеннях із збереженням візуального стилю та вирішено такі завдання:

– проведено аналіз сучасних підходів до детекції тексту (MSER, SWT, EAST, CRAFT, DB), оптичного розпізнавання (CRNN, ASTER, TrOCR), машинного перекладу та візуального рендерингу, що дало можливість визначити їх архітектурні особливості, переваги та обмеження;

– проведено систематичне порівняння чотирьох OCR-систем (EasyOCR, PaddleOCR, Surya OCR та Teseract) та трьох сервісів машинного перекладу (Google Translate, DeepL, GPT-4o-mini) на стандартних датасетах (IIIT5K, ICDAR 2013, SROIE) і тестових фрагментах української мови, що дозволило оцінити точність, швидкість та стійкість до різних типів зображень;

– сформовано покроковий алгоритм обробки зображення — від детекції та розпізнавання тексту до перекладу та візуальної реконструкції, що дало можливість реалізувати повний цикл автоматичного перекладу;

– запропоновано адаптивні параметри злиття текстових блоків та багаторядкового розпізнавання, а також реалізовано вертикальний рендеринг, що дозволило розширити систему на східноазійські мови та декоративні написи;

– розроблено модульну архітектуру системи з можливістю вибору OCR- та translation-компонентів через веб-інтерфейс, що забезпечило гнучкість налаштувань і можливість локального виконання без обов’язкової залежності від хмарних сервісів.

У рамках кваліфікаційної роботи було створено програмну систему автоматичного перекладу тексту на зображеннях, що включає модулі детекції, OCR, машинного перекладу та візуального відтворення. Побудовано комплексний алгоритм для кожного етапу та реалізовано його у вигляді

працюючого застосунку. Проведено експериментальні дослідження на 300 тестових зображеннях і 170 текстових фрагментах, що дало можливість зіставити ефективність різних OCR- і методів перекладу.

Проведені експерименти показали, що точність розпізнавання, залежно від вибраної моделі, знаходиться в межах 81,6–99,5%, а середній час обробки одного зображення становить 2,2–15 секунд. Порівняння якості перекладу продемонструвало, що DeepL досягає точності 96,3%, GPT-4o-mini — 96,9% із найкращою контекстною адаптацією, тоді як Google Translate забезпечує найбільш природну українську мову, але має поодинокі некоректні переклади.

Наукова новизна роботи полягає у створенні відкритої та гнучкої системи, придатної для локалізації візуального контенту, автоматичної обробки зображень, в якій користувач сам може обирати модель знаходження тексту, модель перекладу та параметри вставки тексту. Отримані результати дозволяють сформувати рекомендації щодо вибору оптимальних компонентів залежно від вимог до точності, швидкості та типу зображень.

Результати роботи апробовано у вигляді 2 тез доповідей під час IV Міжнародної науково-практичної конференцію «Technologies, theories and developments: modern scientific teaching», 23-26 вересня 2025 р., Валенсія, Іспанія [66], та II Міжнародної науково-практичної конференції «INNOVATIONS OF MODERN SCIENCE AND EDUCATION», 29-31.10.2025, Ванкувер, Канада [67].

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Daradkeh, Y. I., Gorokhovatskyi, V., Tvoroshenko, I., & Zeghid, M. (2022). Tools for fast metric data search in structural methods for image classification. IEEE Access, 10, pp. 124738–124746.2.

2. Long, S., He, X., & Yao, C. (2021). Scene text detection and recognition: The deep learning era. International Journal of Computer Vision, 129, pp. 161–184.

3. Gorokhovatskyi, V., Stiahlyk, N., & Tsarevska, V. (2021). Combination method of accelerated metric data search in image classification problems. Advanced Information Systems, 5(3), pp. 5–12.

4. Shi, B., Bai, X., & Yao, C. (2017). An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence, 39(11), pp. 2298–2304.

5. Daradkeh, Y. I., Gorokhovatskyi, V., Tvoroshenko, I., Gadetska, S., & Al-Dhaifallah, M. (2021). Methods of classification of images on the basis of the values of statistical distributions. IEEE Access, 9, pp. 92964–92973.

6. Daradkeh, Y. I., Gorokhovatskyi, V., Tvoroshenko, I., & Zeghid, M. (2024). Improving the effectiveness of image classification structural methods. Computers, Materials & Continua, 80(2), pp. 3085–3106.

7. Niu, L., Meng, F., & Zhou, J. (2024). UMTIT: Unifying recognition, translation, and generation for multimodal text image translation. In Proceedings of LREC-COLING 2024 pp. 16953–16972.

8. Gorokhovatskyi, O., Gorokhovatskyi, V., & Peredrii, O. (2018). Analysis of application of cluster descriptions in space of characteristic image features. Data, 3(4), pp. 52.

9. Gorokhovatskyi, O., Peredrii, O., Gorokhovatskyi, V., & Vlasenko, N. (2023). Explanation of CNN image classifiers with hiding parts. In Explainable Deep Learning Artificial Intelligence pp. 125–146. Academic Press.

10. Gorokhovatskyi, V., Peredrii, O., Tvoroshenko, I., & Markov, T. (2023). Distance matrix for a set of structural description components. Advanced Information Systems, 7(1), pp. 5–13.

11. Ковтуненко, А., Яковлева, О., & Любченко, В. (2020). Дослідження сумісного використання математичної морфології та згорткових нейронних мереж для розпізнавання цінників. Вісник НТУ «ХПІ», 1(3), pp. 24–31.

12. Du, Y., Li, C., & Guo, R. (2022). PP-OCRv3: More attempts for the improvement of ultra lightweight OCR system. arXiv. URL: https://arxiv.org/abs/2206.03001 (Дата звернення 04.11.2025).

13. Google Cloud. (n.d.). Cloud Vision API. URL: https://cloud.google.com/vision (Дата звернення 04.11.2025).

14. Amazon Web Services. (n.d.). Amazon Textract. URL: https://aws.amazon.com/textract (Дата звернення 04.11.2025).

15. Gadetska, S. V., & Gorokhovatsky, V. O. (2018). Statistical measures for computation of the image relevance of visual objects. Telecommunications and Radio Engineering, 77(12), pp. 1041–1053.

16. Kobylin, O., Gorokhovatskyi, V., Tvoroshenko, I., & Peredrii, O. (2020). The application of non-parametric statistics methods in image classifiers. Telecommunications and Radio Engineering, 79(10), pp. 855–863.

17. Li, M., Lv, T., & Cui, L. (2023). TrOCR: Transformer-based OCR with pre-trained models. In AAAI Conference on Artificial Intelligence Vol. 37, No. 11, pp.13094–13102.

18. Smith, R. (2007). An overview of the Tesseract OCR engine. In Proceedings of ICDAR, pp. 629–633.

19. Gorokhovatskyi, V. A. (2014). Structural analysis and intellectual data processing in computer vision. Kharkiv.

20. Gorokhovatskyi, V. O., & Gadetska, S. V. (2020). Statistical processing and data mining in structural image classification methods. Kharkiv: FLP Panov A. N.

21. Gorokhovatskyi, V., Tvoroshenko, I., & Chmutov, Y. (2022). Application of systems of orthogonal functions for formation of sign space. Advanced Information Systems, 6(3), pp. 5–12.

22. Daradkeh, Y. I., Tvoroshenko, I., Gorokhovatskyi, V., Latiff, L. A., & Ahmad, N. (2021). Development of effective methods for structural image recognition using fuzzy logic. IEEE Access, 9, pp. 13417–13428.

23. Gorokhovatskyi, V. O., & Gadetska, S. V. (2019). Determination of relevance of visual object images by statistical analysis. Telecommunications and Radio Engineering, 78(3), pp. 211–220.

24. Yakovleva, O., Nebeský, L., & Liakhov, P. (2023). Research methods of texture image analysis to solve the texture search problem. In IV International Scientific and Practical Conference pp. 252–261.

25. Кобилін, О. А., & Творошенко, І. С. (2021). Методи цифрової обробки зображень. Харків: ХНУРЕ.

26. Daradkeh, Y. I., Gorokhovatskyi, V., Tvoroshenko, I., Gadetska, S., & Al-Dhaifallah, M. (2023). Statistical data analysis models for determining the relevance of structural image descriptions. IEEE Access, 11, pp. 126938–126949.

27. Gorokhovatskyi, V., Gadetska, S., & Ponomarenko, R. (2020). Recognition of visual objects based on statistical distributions for blocks of structural description of image. In Lecture Notes in Computational Intelligence and Decision Making pp. 501–512.

28. Pomazan, V., Tvoroshenko, I., & Gorokhovatskyi, V. (2023). Handwritten character recognition models based on CNNs. International Journal of Academic Engineering Research, 7(9), pp. 64–72.

29. JaidedAI. (2020). EasyOCR: Ready-to-use OCR. URL: https://github.com/JaidedAI/EasyOCR (Дата звернення 04.11.2025).

30. Paruchuri, V. (2024). Surya: Multilingual document OCR toolkit. URL: https://github.com/VikParuchuri/surya (Дата звернення 04.11.2025).

31. Afli, H., Barrault, L., & Schwenk, H. (2016). OCR error correction using statistical machine translation. IJCNLP, pp. 105–113.

32. Agrawal, S., Carpuat, M., & Durme, B. V. (2022). Context-aware neural machine translation. TACL, pp. 948–963.

33. O'Brien, S. (2019). Neural machine translation and post-editing efficiency. MT Summit.

34. Raffel, C., et al. (2020). Exploring the limits of transfer learning with a unified text-to-text transformer. JMLR.

35. Bawden, R., et al. (2022). DeepL translation quality for multilingual adaptation. EAMT.

36. Wu, Y., et al. (2016). Google's neural machine translation system. arXiv:1609.08144.

37. Yakovleva, O., Matúšová, S., & Táncošová, J. (2024). Investigation of LLMs… In XVI International Scientific and Practical Conference pp. 289–295.

38. Yakovleva, O., Nebeský, L., & Kirichenko, A. (2023). Using GPT models… In V International Scientific and Practical Conference pp. 172–178.

39. Suprun, A., Tvoroshenko, I., Gorokhovatskyi, V., & Yakovleva, O. (2025). Development and research of a method for combined LLM use. International Journal of Academic and Applied Research, 9(10), pp. 249–263.

40. OpenAI. (2023). GPT-4 Technical Report. URL: https://arxiv.org/abs/2303.08774 (Дата звернення 04.11.2025).

41. OpenAI. (2024). GPT-4o System Card. URL : https://openai.com/index/gpt-4o-system-card/ (Дата звернення 04.11.2025).

42. Творошенко, І. С. (2021). Технології прийняття рішень в інформаційних системах. Харків: ХНУРЕ.

43. Yang, S., et al. (2017). Awesome typography: Statistics-based text effects transfer. CVPR, pp. 7464–7473.

44. Huang, J.-B., et al. (2014). Image completion using planar structure guidance. ACM TOG, 33(4), pp. 1–10.

45. Criminisi, A., Pérez, P., & Toyama, K. (2004). Region filling and object removal by exemplar-based inpainting. IEEE TIP, 13(9), pp. 1200–1212.

46. McKinney, W. (2010). Data structures for statistical computing in Python. In Proceedings of the 9th Python in Science Conference pp. 51–56.

47. Yakovleva, O., Matúšová, S., & Talakh, V. (2025). Gradio and Hugging capabilities… ΛΌΓΟΣ (Boston), pp. 202–205.

48. Harris, C. R., Millman, K. J., van der Walt, S. J., et al. (2020). Array programming with NumPy. Nature, 585, pp. 357–362.

49. Hunter, J. D. (2007). Matplotlib: A 2D graphics environment. Computing in Science & Engineering, 9(3), pp. 90–95.

50. Reback, J., et al. (2021). Pandas. Zenodo.

51. Watanabe, S., & Radev, D. (2020). JiWER: A Python library for ASR error rate computation. URL: https://github.com/jitsi/jiwer (Дата звернення 04.11.2025).

52. JetBrains. (2025). PyCharm IDE – Official Documentation. URL: https://www.jetbrains.com/pycharm (Дата звернення 04.11.2025).

53. Bisong, E. (2019). Building ML & DL models on Google Colab. In Building Machine Learning and Deep Learning Models on Google Cloud Platform. Apress.

54. Bradski, G. (2000). The OpenCV Library. Dr. Dobb's Journal of Software Tools.

55. Pillow Developers. (2025). Pillow: Python Imaging Library. URL: https://python-pillow.org (Дата звернення 04.11.2025).

56. Chacon, S., & Straub, B. (2014). Pro Git. Apress.

57. Sommerville, I. (2016). Software Engineering (10th ed.). Pearson.

58. Karatzas, D., et al. (2013). ICDAR 2013 robust reading competition. In ICDAR.

59. Huang, Y., et al. (2019). ICDAR 2019 competition on scanned receipt OCR and information extraction (SROIE). In ICDAR.

60. Du, Y., et al. (2020). On-device vs cloud-based OCR: Efficiency and accuracy trade-offs. Pattern Recognition Letters.

61. Morris, A., et al. (2004). From WER and RIL to MER and WIL. INTERSPEECH.

62. Ye, F., et al. (2023). Efficient text detection in dense documents using lightweight transformer detectors. Document Analysis Systems.

63. Du, Y., et al. (2020). PaddleOCR: An industrial OCR system. arXiv.

64. Romanov, A., et al. (2023). Surya OCR: Transformer-based text recognition. arXiv.

65. Rice, S., Jenkins, F., & Nartker, T. (1995). The fourth annual test of OCR accuracy. ISRI Technical Report.

66. Харченко, В. В., & Любченко, В. А. (2025). Аналіз сучасних підходів детекції та розпізнавання тексту на зображеннях. *In Technologies, theories and developments: Modern scientific teaching. Proceedings of the IV International Scientific and Practical Conference* pp. 54–59.

67. Харченко, В. В., & Любченко, В. А. (2025). Порівняльний аналіз продуктивності сучасних моделей детекції та розпізнавання тексту на зображеннях. *In Innovations of modern science and education. Proceedings of the 2nd International Scientific and Practical Conference* pp. 376–383.