Вінницький національний технічний університет
Факультет інформаційних технологій та комп'ютерної інженерії
Кафедра обчислювальної техніки

МАГІСТЕРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

на тему:

«ЗАСОБИ ОПТИЧНОГО РОЗПІЗНАВАННЯ СПОТВОРЕНИХ ДРУКОВАНИХ СИМВОЛІВ ТЕКСТОВИХ ДОКУМЕНТІВ»

Виконав: студент 2-го курсу, групи 2КІ-24м
спеціальності 123 – Комп'ютерна інженерія
Перестюк О. В.

Керівник: д.т.н, проф. каф. ОТ
Мартинюк Т. Б.
«12» 12 2025 р.

Допущено до захисту
Завідувач кафедри ОТ
д.т.н., проф. Азаров О. Д.
«16» 12 2025 р.

Опонент: к.т.н, доцент каф. ПЗ
Черноволик Г.О.
«12» 12 2025 р.

Вінницький національний технічний університет
Факультет інформаційних технологій та комп'ютерної інженерії
Кафедра обчислювальної техніки
Галузь знань — Інформаційні технології
Освітній рівень — магістр
Спеціальність — 123 Комп'ютерна інженерія
Освітня програма — Комп'ютерна інженерія

ЗАТВЕРДЖУЮ
Завідувач кафедри
обчислювальної техніки
д.т.н., проф. О. Д. Азаров
«25» вересня 2025 року

ЗАВДАННЯ
НА МАГІСТЕРСЬКУ КВАЛІФІКАЦІЙНУ РОБОТУ

студенту Перестюку Олександру Вікторовичу

1 Тема роботи: «Засоби оптичного розпізнавання спотворених друкованих символів текстових документів».

Керівник роботи — д.т.н., проф. каф. ОТ Мартинюк Тетяна Борисівна, затверджено наказом ВНТУ від «24» вересня 2025 р. № 313.

2 Строк подання студентом роботи 10 грудня 2025 р

3 Вихідні дані до роботи: зображення текстових документів із роздільною здатністю не менше 300 dpi, представлені у форматі RGB. Робота виконується для систем оптичного розпізнавання текстів, що містять спотворення (шум, нахил, нерівномірність яскравості, дефекти друку).

4 Зміст розрахунково-пояснювальної записки: вступ, аналіз методів та засобів оптичного розпізнавання спотворених друкованих символів текстових документів, розробка технології та вибір засобів розпізнавання спотворених друкованих символів, розробка програмних засобів оптичного розпізнавання, тестування та оцінка ефективності функціонування створеної програми, розрахунок економічної доцільності створення програмного забезпечення для розпізнавання спотворених друкованих символів.

5 Перелік графічного матеріалу: структурна схема методів розпізнавання тексту, структурна схема класифікації нейронних мереж, блок-схема процесу розпізнавання тексту, структурна схема програми розпізнавання тексту.

6 Консультанти розділів роботи наведені в таблиці 1.

Таблиця 1 — Консультанти розділів

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийня
1-4	д. т. н., проф. каф. ОТ Мартинюк Тетяна Борисівна		
5	к .т. н., доц. каф. ЕПВМ Адлер Оксана Олександрівна		
нормо-контроль	асистент каф. ОТ Швець Сергій Ілліч		

7 Дата видачі завдання 25.09.2025

8 Календарний план виконання етапів магістерської кваліфікаційної роботи наведено в таблиці 2.

Таблиця 2 — Календарний план

№ з/п	Назва етапів проекту (роботи)	Строк виконання етапів роботи	Примітка
1	Пошук та огляд інформаційних джерел	26.09.2025р.	вик.
2	Аналіз існуючих методів і засобів оптичного розпізнавання спотворених друкованих символів	10.10.2025р.	вик.
3	Дослідження алгоритмів попередньої обробки та сегментації зображень	25.10.2025р.	вик.
4	Проектування нейромережевої технології	19.11.2025р.	вик.
5	Розрахунок економічної доцільності	02.11.2025р.	вик.
6	Оформлення пояснювальної записки і презентації	06.11.2025р.	вик.
7	Попередній захист	10.11.2025р.	вик.
8	Перевірка «антиплагіат»	01.12.2025р.	вик.
9	Аналіз виконання роботи, висновки, додатки	05.12.2025р.	вик.
10	Оформлення пояснювальної записки та ілюстративного матеріалу	07.12.2025р.	вик.
11	Перевірка якості виконання магістерської роботи та усунення недоліків	10.12.2025р.	вик.

Студент Олександр ПЕРЕСТЮК

Керівник д.т.н., професор Тетяна МАРТИНЮК

АНОТАЦІЯ

УДК 004.93

Перестюк О. В. Засоби оптичного розпізнавання спотворених друкованих символів текстових документів. Магістерська кваліфікаційна робота зі спеціальності 123 – «Комп'ютерна інженерія», освітня програма «Комп'ютерна інженерія». Вінниця: ВНТУ, 2025. — 123 с.

На укр. мові. Бібліогр.: 36 назв; рис. 13; табл. 6.

У роботі розглянуто методи і засоби оптичного розпізнавання спотворених друкованих символів , визначено їхні обмеження при роботі з неякісними або пошкодженими зображеннями. Запропоновано вдосконалену технологію обробки зображень, яка включає етапи попередньої фільтрації, сегментації та класифікації символів із використанням нейронних мереж.

Розроблено програмний засіб для розпізнавання спотворених друкованих символів, що забезпечує адаптацію до різних типів спотворень і підвищує точність відтворення текстової інформації. Проведено навчання нейронної мережі на експериментальному наборі даних, виконано тестування програми та оцінено її ефективність у порівнянні з традиційними підходами. Виконано економічне обґрунтування доцільності створення та впровадження розробленого програмного продукту.

Ключові слова: оптичне розпізнавання тексту, спотворені символи, сегментація, згорткова нейронна мережа, машинне навчання, обробка зображень.

ABSTRACT

Perestiuk O. V. Optical Recognition Tools for Distorted Printed Characters in Text Documents. Master's Qualification Thesis in specialty 123 – "Computer Engineering", educational program "Computer Engineering". Vinnytsia: VNTU, 2025. — 123 p.

In Ukrainian. Bibliography: 36 sources; figures: 13; tables: 6.

The thesis examines methods and tools for optical recognition of distorted printed characters and identifies their limitations when processing low-quality or damaged images. An improved image processing technology is proposed, which includes stages of preliminary filtering, segmentation, and character classification using neural networks.

A software tool for recognizing distorted printed characters has been developed, providing adaptation to various types of distortions and increasing the accuracy of text information reconstruction. A neural network was trained on an experimental dataset; the software tool was tested, and its effectiveness was evaluated in comparison with traditional OCR approaches. An economic justification for the development and implementation of the proposed software product has also been performed.

Keywords: optical character recognition, distorted characters, segmentation, convolutional neural network, machine learning, image processing.

ЗМІСТ

ВСТУП

В умовах стрімкого зростання обсягів цифрової інформації актуальним завданням сучасних інформаційних технологій є розроблення систем аналізу, розпізнавання та класифікації візуальних даних, зокрема засобів оптичного розпізнавання тексту, що забезпечують автоматичне перетворення зображень друкованих документів у редагований текстовий формат. Такі системи широко застосовуються в задачах архівування та цифровізації документів, автоматизованого заповнення форм і пошуку текстових даних, однак процес розпізнавання символів у реальних умовах ускладнюється наявністю спотворень, спричинених низькою якістю друку, нерівномірним освітленням, шумами або перекосами сторінок, що негативно впливає на точність традиційних OCR-систем.

Використання алгоритмів комп’ютерного зору у поєднанні з методами штучного інтелекту дозволяє автоматизувати процес аналізу зображень і підвищити надійність розпізнавання текстової інформації [1]. Застосування таких технологій відкриває можливість не лише відновлення чітких текстів, але й успішного розпізнавання документів, що містять пошкоджені, частково стерті або спотворені символи. Саме тому розроблення ефективних засобів оптичного розпізнавання спотворених друкованих символів є актуальним науково-практичним завданням сучасного етапу розвитку комп’ютерної інженерії.

У сучасних умовах зростання обсягів електронних даних виникає нагальна потреба в автоматизованій обробці документів, які зберігаються у паперовому вигляді або у вигляді графічних зображень. Від ефективності роботи систем оптичного розпізнавання безпосередньо залежить швидкість переходу від аналогових носіїв до цифрових баз даних. Проте якість отриманих зображень далеко не завжди є задовільною: на сканах часто присутні тіні, дефекти друку, викривлення та шуми, які ускладнюють

розпізнавання символів. Це особливо характерно для архівних документів, технічних інструкцій або копій, створених із великим ступенем зношеності.

Класичні алгоритми, що базуються на порівнянні зразків або геометричному аналізі контурів [2], не здатні повною мірою враховувати варіативність спотворених символів. У таких випадках система або помилково класифікує символи, або взагалі не розпізнає їх, що призводить до втрати частини інформації. Виникає потреба у вдосконаленні підходів до попередньої обробки зображень, сегментації та аналізу ознак символів, а також у впровадженні методів, здатних адаптуватися до зміни умов зображення.

Використання технологій машинного навчання, зокрема нейронних мереж, створює нові можливості для вирішення цієї проблеми. Такі системи здатні самостійно формувати внутрішні моделі представлення символів, узагальнювати закономірності у даних та розпізнавати навіть ті зразки, які не входили до навчальної вибірки [3]. Це відкриває перспективи побудови інтелектуальних OCR-рішень, що можуть стабільно працювати зі спотвореними або частково пошкодженими текстами.

Отже, незважаючи на досягнутий прогрес у розвитку технологій оптичного розпізнавання, проблема підвищення точності і надійності розпізнавання спотворених друкованих символів залишається **актуальною**. Тому завдання створення вдосконалених методів і засобів OCR, здатних адаптуватися до складних умов обробки зображень текстових документів, потребує подальших досліджень та практичної реалізації.

Магістерська робота виконана по кафедральній науково-дослідній тематиці студентського наукового гуртка кафедри ОТ «Розробка комп’ютерної системи пошуку і розпізнавання об’єктів», керівником якого є старший викладач кафедри Очкуров М. А.

Метою дослідження є вдосконалення методів оптичного розпізнавання текстової інформації з друкованих документів, що містять спотворені або низькоякісні символи, шляхом застосування сучасних алгоритмів обробки зображень та нейронних мереж для підвищення точності.

Задачі дослідження:

— виконати аналіз існуючих методів і алгоритмів оптичного розпізнавання текстової інформації, зокрема друкованих символів та розглянути основні причини виникнення спотворень у цифрових зображеннях текстових документів та їхній вплив на точність розпізнавання;

— обґрунтувати підхід до підвищення стійкості систем OCR до спотворень шляхом удосконалення методів попередньої обробки та сегментації зображень;

— створити алгоритм розпізнавання друкованих символів на основі нейронних мереж, здатний адаптуватися до варіативності шрифтів та якості друку, обрати інструментарій і розробити програмний засіб для розпізнавання спотворених друкованих символів;

— виконати тестування програмного продукту та здійснити оцінку точності розпізнавання спотворених символів, визначити переваги створеної програми порівняно з традиційними методами оптичного розпізнавання;

— виконати обґрунтування доцільності виконання нового наукового рішення, розрахувати економічні витрати для створення програмних засобів розпізнавання друкованих символів тестових документів та визначити переваги від впровадження нового програмного продукту.

Об’єкт дослідження — процес оптичного розпізнавання друкованих текстових документів, що містять спотворення або дефекти зображення символів.

Предмет дослідження — методи та програмні засоби підвищення точності оптичного розпізнавання спотворених друкованих символів за допомогою нейронних мереж і сучасних технологій обробки зображень.

Методи дослідження: використовувалися методи цифрової обробки зображень для покращення якості текстових документів перед розпізнаванням, методи дискретної математики для роботи з матричними представленнями, а також елементи теорії ймовірностей і математичної статистики для оцінювання точності класифікації. Для моделювання процесу

розпізнавання застосовано алгоритми машинного навчання, зокрема згорткові нейронні мережі, адаптивні до спотворень і шумів. Реалізацію програмної частини виконано з використанням принципів об’єктно-орієнтованого програмування, що забезпечило гнучку структуру системи.

Наукова новизна отриманих результатів магістерської роботи полягає у тому, що удосконалено технологію обробки цифрового зображення текстового документу для виділення й розпізнавання спотворених друкованих символів тексту, яка відрізняється від відомих підходів послідовним виконанням етапів сегментації зображення, використанням нейронних мереж і лінгвістичного доопрацювання, що дозволяє підвищити точність розпізнавання друкованих символів у отриманому зображенні тексту, у тому числі і символів із дефектами.

Практичне значення одержаних результатів:

— створено послідовність оптичного розпізнавання друкованих символів із використанням нейронних мереж, що забезпечує стійкість до різних видів спотворень;

— розроблено програмний засіб для обробки та розпізнавання текстових документів, який може застосовуватися для цифровізації архівів, документів технічного призначення або текстів, отриманих за допомогою побутових сканерів і мобільних пристроїв.

Апробація результатів магістерської роботи зроблено доповідь на LV Всеукраїнській науково-технічній конференції підрозділів ВНТУ (ВНТКП ВНТУ, Вінниця, 2025 р.).

За результати магістерської роботи **опубліковано** 1 тези доповіді: Перестюк О. В., Мартинюк Т. Б, Очкуров М. А. Засоби оптичного розпізнавання спотворених друкованих символів текстових документів. // Міжнародна науково-практична інтернет-конференція «Молодь в науці: дослідження, проблеми, перспективи» (НТКП ВНТУ, Вінниця, 2025 р.). [Електронний ресурс]. — Режим доступу: https://conferences.vntu.edu.ua/index.php/mn/mn2026/paper/view/26769.

1 АНАЛІЗ МЕТОДІВ ТА ЗАСОБІВ ОПТИЧНОГО РОЗПІЗНАВАННЯ ТЕКСТУ

Оптичне розпізнавання тексту ґрунтується на аналізі зображень друкованих документів і перетворенні візуальної інформації у машинно-читану форму. Основна ідея полягає у виділенні символів та їх подальшій ідентифікації за набором характерних ознак, що дозволяє відтворювати текстові дані у цифровому вигляді. При цьому на точність результатів істотно впливають якість зображення, чіткість контурів і наявність спотворень, які можуть виникати внаслідок дефектів друку, перекосу сторінки або нерівномірного освітлення під час сканування.

1.1 Загальні принципи та етапи оптичного розпізнавання текстової інформації

Сучасні інформаційні технології активно розвиваються у напрямі автоматизації обробки документів, що містять текстову інформацію у графічному вигляді. Одним із найважливіших напрямів цього процесу є оптичне розпізнавання тексту (Optical Character Recognition, OCR), яке забезпечує перетворення зображення тексту, отриманого за допомогою сканера чи цифрової камери, у редагований машинний формат. Така технологія дає змогу значно скоротити витрати часу на введення даних і підвищити ефективність роботи з великими обсягами текстової інформації.

Принцип роботи систем оптичного розпізнавання ґрунтується на комплексній обробці цифрового зображення, метою якої є виявлення, аналіз і відтворення символів, що містяться на ньому. На початковому етапі формується електронне зображення документа, яке зазвичай має різний рівень якості залежно від умов сканування чи фотографування. Після цього виконується попередня обробка, під час якої здійснюється фільтрація шумів, вирівнювання контрасту, усунення фону та інших дефектів, що можуть перешкоджати точному аналізу тексту [4].

Подальший етап передбачає виділення окремих елементів тексту, зокрема рядків, слів і символів. Такий процес називається сегментацією і має важливе значення для подальшої ідентифікації символів. Для кожного виділеного символу визначаються числові характеристики, що описують його форму, пропорції та структуру. Ці характеристики утворюють набір ознак, який використовується для класифікації.

Безпосереднє розпізнавання здійснюється шляхом порівняння отриманих ознак із відомими зразками або за допомогою спеціальних алгоритмів машинного навчання. Якщо раніше більшість систем використовувала класичні шаблонні або статистичні методи, то сьогодні дедалі ширше застосовуються нейронні мережі, здатні адаптуватися до різних типів символів та умов їхнього зображення. Завдяки цьому процес розпізнавання стає більш гнучким і менш залежним від якості початкового зображення.

На завершальному етапі результати обробки проходять перевірку та корекцію, після чого текст відтворюється у зручному для користувача форматі. У цілому процес оптичного розпізнавання тексту являє собою багатоступеневу послідовність дій, у якій поєднуються методи цифрової обробки зображень, статистичного аналізу та штучного інтелекту. Від якості виконання кожного з цих етапів залежить точність і стійкість системи до спотворень, що визначає актуальність подальших досліджень у цій галузі [5].

Протягом свого розвитку технології оптичного розпізнавання тексту зазнали значних змін. Перші системи з’явилися ще у середині XX століття та ґрунтувалися на простих шаблонних методах, коли кожному символу відповідала певна форма, закладена у базі зразків. Такі системи могли розпізнавати лише обмежений набір символів, зазвичай друкованих великими літерами одного шрифту. Зі зростанням обчислювальних потужностей і появою нових алгоритмів ці підходи поступово вдосконалювалися, однак залишалися вразливими до найменших спотворень — наприклад, нахилу символу або зміни товщини лінії.

Наступним етапом розвитку стали структурні та ознакові методи, у яких розпізнавання виконувалося не за формою цілого символу, а за його окремими елементами. Такий підхід дозволив створювати більш гнучкі системи, здатні аналізувати складні шрифти та часткові дефекти зображення. Алгоритми цього типу базувалися на аналізі геометричних характеристик, напрямків контурів, кількості перетинів і пропорцій символів. Хоча точність таких систем перевищувала можливості шаблонних, вони залишалися чутливими до шумів і вимагали ретельного налаштування параметрів для кожного конкретного випадку.

Подальший розвиток OCR-технологій пов’язаний із появою статистичних методів і алгоритмів машинного навчання. Ці підходи дозволили автоматизувати процес формування ознак та оптимізувати класифікацію на основі великих обсягів навчальних даних. Системи почали не лише порівнювати символи з еталонами, а й навчатися розпізнавати їх за ймовірнісними характеристиками. Такий підхід став ключовим кроком до створення інтелектуальних систем оптичного розпізнавання.

Найбільш відчутний прорив у розвитку технологій відбувся з поширенням нейронних мереж, які здатні самостійно виділяти та узагальнювати ознаки, необхідні для розпізнавання. Завдяки цьому сучасні системи OCR можуть працювати зі спотвореними, неякісними або частково пошкодженими зображеннями тексту, досягаючи високої точності навіть у складних умовах. Використання згорткових нейронних мереж дало змогу перейти від ручного вибору ознак до повністю автоматизованого аналізу зображення, що суттєво розширило можливості розпізнавання як друкованих, так і рукописних символів.

Таким чином, еволюція технологій оптичного розпізнавання пройшла шлях від простих шаблонних методів до інтелектуальних систем, здатних навчатися на прикладах і адаптуватися до різних типів зображень. Цей розвиток створив передумови для дослідження таких складних завдань, як розпізнавання спотворених друкованих символів, що є одним із ключових

напрямів удосконалення сучасних OCR-рішень.

Попри значний прогрес у розвитку систем оптичного розпізнавання, проблема забезпечення стабільної точності при роботі з неякісними або спотвореними зображеннями залишається актуальною. У реальних умовах сканування чи фотографування текстових документів часто супроводжується низкою факторів, які ускладнюють процес обробки. Це можуть бути нечіткі контури символів, дефекти друку, нерівномірне освітлення, відблиски, викривлення сторінки або зміщення рядків. У таких випадках класичні алгоритми виявляються малоефективними, оскільки вони залежать від чіткого контурного зображення символу та стабільних параметрів шрифту.

Складність розпізнавання спотворених друкованих символів полягає також у тому, що спотворення можуть бути різної природи — геометричні, фотометричні або структурні. Вони змінюють форму символу, його розміри, контрастність і навіть співвідношення між елементами, що призводить до помилкової класифікації. Тому для підвищення точності розпізнавання необхідно використовувати методи, здатні виявляти не лише зовнішні ознаки символів, але й приховані залежності між їхніми структурними елементами.

Застосування нейронних мереж у системах оптичного розпізнавання дало змогу суттєво зменшити вплив таких факторів. На відміну від традиційних алгоритмів, які потребують попереднього формування набору ознак, нейронні мережі самостійно навчаються виділяти найбільш інформативні характеристики, що забезпечують правильне розпізнавання навіть за наявності шумів і спотворень. Завдяки цьому вони можуть адаптуватися до змін умов зображення та ефективно працювати з різними типами друкованих документів [6].

Таким чином, розвиток технологій оптичного розпізнавання поступово спрямовується у бік інтелектуальних систем, здатних навчатися на реальних даних і враховувати особливості спотворених символів. Саме використання нейронних мереж і глибинного навчання відкриває нові можливості для

підвищення точності, швидкодії та універсальності OCR-систем, що й визначає напрямок подальших досліджень у цій роботі.

1.2 Методи та алгоритми обробки друкованих символів

Ефективність систем оптичного розпізнавання значною мірою залежить від якості попередньої обробки зображення, адже навіть незначні дефекти можуть суттєво вплинути на точність відтворення тексту. Процес обробки друкованих символів передбачає послідовне виконання операцій, спрямованих на підготовку зображення до подальшого аналізу та класифікації. Основна мета цих операцій полягає у покращенні візуальної якості тексту, видаленні шумів, нормалізації контрасту та забезпеченні стабільного вигляду символів незалежно від умов їхнього отримання.

На етапі первинної обробки зображення відбувається корекція яскравості та контрастності, а також усунення дефектів, що виникають під час сканування чи фотографування документа. Для цього застосовуються різноманітні методи фільтрації, зокрема згладжування дрібних шумів або підсилення контурів. Важливим завданням є також усунення викривлень сторінки та вирівнювання рядків, адже навіть невеликий нахил тексту може призвести до помилкової сегментації символів.

Після усунення основних спотворень виконується процес перетворення зображення у форму, зручну для подальшого аналізу. Найчастіше зображення переводиться у двійковий формат, де кожен піксель має лише два стани — чорний або білий. Такий підхід спрощує подальше розділення фону та тексту, зменшує обсяг даних для обробки й забезпечує можливість точного виділення меж символів.

Одним із ключових етапів у процесі обробки друкованих символів є сегментація, яка передбачає поділ зображення тексту на окремі структурні елементи. Саме на цьому етапі система переходить від суцільного зображення до логічно впорядкованої структури, де кожен символ може бути розпізнаний окремо. Коректна сегментація визначає успішність усього подальшого

процесу, адже навіть незначна помилка в межах символів або рядків призводить до помилкового визначення тексту.

Сегментація виконується послідовно, спочатку виділяються області сторінки, що містять текстові блоки, потім — окремі рядки, далі слова і нарешті символи. Цей процес вимагає точного виявлення меж між об’єктами, які можуть мати неоднорідну щільність, різний нахил або нерівномірний інтервал. Особливо складним є розділення символів у випадках, коли вони торкаються один одного або з’єднані артефактами друку.

У сучасних системах для вирішення цього завдання широко використовуються алгоритми аналізу зв’язності пікселів та проекційні методи, що дозволяють визначати границі текстових об’єктів на основі розподілу чорних і білих пікселів по горизонталі та вертикалі. Для складніших випадків застосовуються методи контурного аналізу та морфологічні перетворення, які допомагають відокремлювати літери навіть за умов їхнього часткового накладання.

Результатом успішної сегментації є отримання впорядкованої структури зображення, де кожному фрагменту відповідає конкретний символ. Надалі ці символи подаються на етап формування ознак, що забезпечує можливість класифікації й розпізнавання. Таким чином, сегментація виступає проміжною ланкою між візуальною інформацією та її інтелектуальною обробкою, визначаючи загальну якість роботи всієї системи оптичного розпізнавання.

Після успішної сегментації текстового зображення система переходить до етапу формування ознак, що описують кожен окремий символ. Цей процес є одним із найважливіших у структурі оптичного розпізнавання, оскільки саме від правильного вибору ознак залежить точність і надійність класифікації. Метою цього етапу є перетворення графічного зображення символу у набір числових характеристик, які однозначно відображають його форму та структуру [7].

Ознаки можуть відображати як зовнішній вигляд символу, так і його внутрішні властивості. До них належать пропорції елементів, орієнтація ліній,

кількість замкнених областей, напрямки контурів, співвідношення висоти до ширини, густина заповнення та інші параметри, що визначають індивідуальність кожної літери. Формування таких ознак дозволяє зменшити кількість даних для подальшої обробки, водночас зберігаючи необхідну інформацію для ідентифікації символу.

У традиційних системах OCR процес виділення ознак здійснюється з використанням евристичних правил або геометричних алгоритмів, які аналізують структуру символу за певними критеріями. Однак цей підхід має обмеження, оскільки не враховує варіацій, спричинених спотвореннями, зміною шрифту або різницею у товщині ліній. Тому з розвитком обчислювальних методів усе частіше використовуються підходи, що базуються на статистичному аналізі та машинному навчанні. Вони дозволяють системі автоматично визначати найінформативніші ознаки без необхідності ручного налаштування.

Виділення ознак є проміжним етапом між аналізом зображення і розпізнаванням, який забезпечує перехід від візуальної інформації до математичної моделі символу. Якість цього перетворення визначає, наскільки точно алгоритм зможе зіставити реальний символ із відповідним еталоном, а отже, впливає на загальну ефективність системи розпізнавання друкованого тексту.

Після виділення ознак система переходить до етапу класифікації, який полягає у визначенні, якому символу відповідає набір отриманих характеристик. Цей процес є центральним у системі оптичного розпізнавання, адже саме він забезпечує перехід від числового опису до конкретного текстового результату. Суть класифікації полягає у порівнянні вхідних ознак із відомими зразками, що зберігаються в базі даних або формуються під час навчання системи.

У класичних OCR-системах найпростішим способом класифікації є метод порівняння зі зразками, коли кожному символу відповідає певний шаблон. Система вимірює ступінь подібності між зображенням, що

розпізнається, та еталоном, обираючи найбільш близький варіант. Такий підхід є ефективним для чітких і однорідних друкованих текстів, проте втрачає точність у випадках зміни шрифтів або появи спотворень [8].

Більш гнучкими виявилися статистичні методи, у яких рішення про належність символу до певного класу приймається на основі ймовірнісних оцінок. Такі методи враховують варіації ознак і дозволяють системі працювати з неповними або нечіткими даними. У цьому випадку замість точного збігу ознак використовується оцінка схожості, що підвищує стійкість алгоритму до різних видів спотворень.

Подальший розвиток систем класифікації призвів до появи гібридних підходів, які поєднують переваги шаблонного й статистичного аналізу. Вони дозволяють не лише визначати символи за формою, але й враховувати контекст, наприклад, частоту появи літер у словах або ймовірність їхнього сусідства. Завдяки цьому зменшується кількість помилок при розпізнаванні подібних символів, таких як «O» та «0» або «l» та «I».

Методи класифікації постійно вдосконалюються, однак навіть найкращі алгоритми мають обмеження, коли йдеться про роботу з сильно спотвореними або пошкодженими символами. Саме тому в сучасних системах усе частіше використовуються нейронні мережі, здатні автоматично навчатися розрізняти складні образи без явного програмування правил, що й стало основою для розвитку нового покоління OCR-технологій.

Після завершення етапу класифікації система формує попередній результат розпізнавання, який зазвичай містить окремі неточності, спричинені дефектами зображення або неоднозначністю символів. Для усунення таких помилок застосовується післяобробка, що виконує функції перевірки, корекції та уточнення отриманого тексту. Цей процес є необхідним навіть у найсучасніших системах OCR, оскільки дозволяє підвищити точність розпізнавання без зміни основних алгоритмів.

Післяобробка базується на поєднанні лінгвістичних і статистичних методів аналізу. Система перевіряє розпізнані слова за словником, визначає

малоймовірні комбінації символів і виправляє їх на основі контексту. Наприклад, якщо у тексті з’являється послідовність символів, що не утворює жодного відомого слова, алгоритм обчислює варіанти заміни, враховуючи схожість символів та частоту їхнього використання в мові. Завдяки цьому вдається автоматично виправляти типові помилки, зумовлені сплутуванням схожих літер чи цифр.

Крім лінгвістичних методів, у процесі післяобробки можуть використовуватися евристичні правила, які враховують особливості друкованого тексту. Наприклад, система може автоматично коригувати пропуски пробілів між словами або вирівнювати розділові знаки, зміщені під час сканування. У більш складних випадках застосовуються моделі машинного навчання, які навчаються на великих обсягах текстів і здатні прогнозувати найімовірнішу послідовність символів для кожного фрагмента.

Таким чином, післяобробка відіграє роль завершального інтелектуального фільтра, який перетворює проміжний результат розпізнавання на повноцінний текстовий документ. Вона не лише підвищує точність і читабельність отриманого тексту, але й робить систему OCR більш адаптованою до реальних умов, у яких якість зображень далеко не завжди є ідеальною.

Підсумовуючи розглянуті методи, можна зазначити, що процес оптичного розпізнавання друкованого тексту є складною багатоступеневою системою, у якій кожен етап має важливе значення для досягнення високої точності результатів. Від ефективності попередньої обробки, точності сегментації та правильності формування ознак залежить успішність класифікації символів, а завершальна післяобробка визначає якість кінцевого тексту. Кожен з етапів доповнює інший, утворюючи цілісну структуру, де результат однієї процедури стає основою для наступної

Сучасні алгоритми OCR поступово еволюціонують від простих геометричних і статистичних методів до більш складних інтелектуальних систем, які здатні навчатися на прикладах та адаптуватися до різних умов

обробки. Використання машинного навчання та нейронних мереж дозволило підвищити стійкість до спотворень, варіацій шрифтів і змін контрасту, що раніше створювали значні труднощі для класичних алгоритмів [9].

Незважаючи на високий рівень розвитку сучасних методів, жодна система не є універсальною. Різні підходи демонструють найкращі результати лише для певних типів текстів або умов зображення. Саме тому розробка комбінованих рішень, які поєднують класичні алгоритми обробки з інтелектуальними моделями, залишається актуальним напрямом досліджень. Такі системи здатні забезпечувати баланс між швидкістю, точністю та стійкістю до зовнішніх факторів, що є ключовим завданням сучасних технологій оптичного розпізнавання.

У подальшому розвиток OCR-технологій відбувається переважно у напрямі програмних систем, які поєднують традиційні методи обробки зображень із потужними засобами штучного інтелекту.

1.3 Програмні засоби та інструменти оптичного розпізнавання друкованих документів

Сучасні технології оптичного розпізнавання тексту реалізуються у вигляді програмних засобів, що поєднують у собі методи цифрової обробки зображень, машинного навчання та лінгвістичного аналізу. Розвиток цих систем став важливим етапом у процесі цифровізації документів, адже саме програмні реалізації зробили можливим масове використання OCR у побуті, бізнесі, освіті та наукових дослідженнях. Завдяки їм стало можливим швидке перетворення друкованої інформації в електронну форму без втрати структури тексту, шрифтів і форматування.

Програмні засоби оптичного розпізнавання виконують не лише функцію перетворення зображення у текст, а й забезпечують комплексну обробку документів. До їхніх завдань належать корекція геометричних спотворень, покращення контрасту, відновлення пошкоджених фрагментів та перевірка орфографії. У більшості сучасних систем ці процеси виконуються

автоматично, що дозволяє користувачу отримати готовий результат без необхідності втручання у технічні деталі [10].

Важливою особливістю таких програм є здатність працювати з різними мовами, алфавітами та форматами документів. Це забезпечується завдяки наявності розширених бібліотек символів і мовних моделей, які адаптуються до специфіки кожної мови. Крім того, більшість систем мають можливість навчання, що дозволяє підвищувати точність розпізнавання під конкретні умови використання — наприклад, при обробці старих архівних матеріалів або документів із нестандартними шрифтами.

Програмні рішення OCR сьогодні розвиваються у двох основних напрямах. З одного боку, це комерційні продукти, орієнтовані на широке коло користувачів, з високим рівнем автоматизації та зручним інтерфейсом. З іншого — це відкриті програмні платформи, що дозволяють дослідникам і розробникам удосконалювати алгоритми розпізнавання, експериментуючи з новими моделями та методами. Такий поділ зумовлює різницю в підходах до реалізації, але обидва напрями сприяють загальному розвитку технології OCR і її інтеграції у різноманітні інформаційні системи.

Серед найпоширеніших програмних рішень для оптичного розпізнавання тексту особливе місце посідають комерційні системи, які поєднують високу точність роботи, підтримку великої кількості мов та інтуїтивно зрозумілий інтерфейс. Одним із лідерів у цій галузі є програма ABBYY FineReader, що вважається еталоном серед систем OCR. Вона забезпечує високу точність розпізнавання навіть при роботі з документами низької якості, завдяки поєднанню алгоритмів попередньої обробки зображення, контекстного аналізу та лінгвістичної перевірки. FineReader підтримує розпізнавання понад двохсот мов, зберігає форматування документів і дозволяє експортувати результати у різні формати — від простого тексту до редагованих файлів Word або PDF [11].

Ще одним популярним продуктом є Adobe Acrobat OCR, який входить до складу пакета Adobe Acrobat і орієнтований переважно на роботу з PDF-

документами. Його особливістю є тісна інтеграція з іншими програмами компанії Adobe, що дозволяє не лише розпізнавати текст, а й одразу виконувати редагування, перевірку структури документа та обробку графічних елементів. Завдяки оптимізації алгоритмів ця система ефективно працює з кольоровими та багатосторінковими документами, забезпечуючи стабільну якість результатів [12].

До групи відомих комерційних продуктів також належать Readiris, OmniPage та SimpleOCR, які, хоча й поступаються лідерам за рівнем інтелектуальної обробки, проте відзначаються швидкодією та зручністю використання. Вони орієнтовані на масове сканування документів і часто застосовуються в офісних середовищах, де головним критерієм є швидке отримання прийнятного результату.

Загальною тенденцією для комерційних систем є орієнтація на кінцевого користувача, тобто на спрощення процесу розпізнавання. Сучасні програми автоматично визначають мову документа, розмітку сторінки, розпізнають таблиці, зображення та навіть рукописні елементи. Це робить їх універсальними засобами, здатними ефективно працювати в умовах змішаних форматів і різної якості друку.

Поряд із комерційними системами активно розвиваються й відкриті програмні засоби оптичного розпізнавання, які відіграють важливу роль у наукових дослідженнях і розробці нових підходів до обробки зображень. Найбільш відомою серед них є система Tesseract OCR, створена компанією Hewlett-Packard і пізніше підтримана Google. Вона має відкритий вихідний код, що дозволяє розробникам адаптувати алгоритми під власні потреби, а також експериментувати з різними моделями обробки символів. Завдяки впровадженню згорткових нейронних мереж у нових версіях Tesseract забезпечує високу точність розпізнавання друкованих текстів і підтримує десятки мов [13].

Ще одним важливим інструментом є OCRopus, який орієнтований на дослідницьке використання та модульну архітектуру. Ця система дозволяє

комбінувати різні алгоритми обробки — від попередньої фільтрації зображень до глибинного навчання. Вона використовується переважно у наукових лабораторіях і навчальних проектах, де важлива можливість модифікації алгоритмів і аналізу ефективності різних моделей. Подібний підхід реалізовано й у бібліотеках EasyOCR та PaddleOCR, що базуються на сучасних фреймворках машинного навчання та підтримують роботу з графічними процесорами.

Відкриті системи OCR мають кілька суттєвих переваг — гнучкість налаштування, можливість навчання на власних наборах даних та відсутність ліцензійних обмежень. Завдяки цьому вони широко застосовуються для створення спеціалізованих рішень, наприклад, для розпізнавання технічних шрифтів, старовинних текстів або спотворених документів. Водночас їх використання вимагає глибших технічних знань і участі користувача у процесі налаштування, що відрізняє їх від комерційних систем із повною автоматизацією.

Загалом саме відкриті OCR-платформи стали основою для експериментів із застосуванням нейронних мереж у розпізнаванні символів. Вони забезпечили дослідницьке середовище, у якому з’явилися сучасні гібридні методи, здатні поєднувати класичні алгоритми з глибинним навчанням. Це створило передумови для подальшого розвитку інтелектуальних систем розпізнавання, які вже сьогодні демонструють високу стійкість до шумів, дефектів і спотворень друкованих текстів.

1.4 Технології розпізнавання символів на основі нейронних мереж

Застосування нейронних мереж у системах оптичного розпізнавання тексту стало одним із найважливіших етапів еволюції технологій OCR. Якщо класичні методи базувалися на жорстко визначених алгоритмах порівняння ознак і шаблонів, то нейронні мережі забезпечили можливість автоматичного навчання системи без необхідності ручного визначення правил. Такий підхід відкрив новий рівень гнучкості та адаптивності, дозволивши ефективно

працювати зі спотвореними, нерівномірно освітленими або частково пошкодженими символами.

Основна ідея використання нейронних мереж полягає в моделюванні процесів людського сприйняття, коли система не просто порівнює окремі елементи символів, а вчиться розпізнавати їх як цілісні образи. Це стало можливим завдяки здатності мережі аналізувати взаємозв'язки між численними ознаками, формуючи внутрішні уявлення про структуру символів. Таким чином, процес розпізнавання перетворюється з жорстко детермінованого на статистично-адаптивний, де кожне рішення приймається з урахуванням накопиченого досвіду навчання.

На відміну від традиційних алгоритмів, які погано реагують на зміни форми чи розміру символів, нейронні мережі здатні узагальнювати інформацію, розпізнаючи навіть ті зразки, що відрізняються від навчальних прикладів. Завдяки цьому вони демонструють високу точність у складних умовах, де звичайні системи часто помиляються. Саме тому сучасні технології розпізнавання дедалі частіше базуються на архітектурах глибинного навчання, здатних самостійно виділяти ознаки з вхідних даних без втручання розробника.

Впровадження нейронних мереж у процес OCR не лише підвищило якість розпізнавання, але й дало змогу обробляти значно більші обсяги інформації. Завдяки паралельним обчисленням і використанню графічних процесорів сучасні моделі можуть навчатися на мільйонах прикладів, формуючи універсальні закономірності, які потім застосовуються до нових документів. Усе це зробило нейронні технології основою нової генерації систем оптичного розпізнавання тексту [14].

Розвиток нейронних технологій дав змогу створити кілька типів архітектур, що використовуються для розпізнавання текстових символів, кожна з яких має свої особливості та сферу застосування. Одними з перших у системах OCR почали застосовуватися багатошарові перцептрони, які моделюють роботу людського мозку шляхом послідовного перетворення

вхідних сигналів у вихідні рішення. Хоча такі мережі показували задовільні результати для простих задач класифікації, вони мали обмежені можливості при роботі з великими та складними зображеннями, оскільки не враховували просторову структуру даних.

Поява згорткових нейронних мереж (Convolutional Neural Networks, CNN) стала справжнім проривом у галузі комп'ютерного зору, у тому числі й у сфері оптичного розпізнавання символів. На відміну від класичних моделей, CNN здатні автоматично виділяти найбільш інформативні ознаки без необхідності ручного програмування фільтрів. Завдяки використанню згорткових шарів, що аналізують локальні області зображення, мережа поступово вчиться розпізнавати контури, лінії, кути та інші базові елементи символів, формуючи ієрархічну структуру представлення даних.

Ще одним важливим типом є рекурентні нейронні мережі (RNN), які враховують послідовність вхідних даних і здатні зберігати контекст попередніх елементів. Це робить їх корисними під час обробки текстових рядків або рукописних записів, де символи тісно пов'язані між собою. У сучасних системах OCR рекурентні мережі часто поєднуються зі згортковими, утворюючи гібридні архітектури, що поєднують просторовий і послідовний аналіз.

Рекурентні моделі здатні ефективно працювати з послідовними структурами даних, оскільки містять внутрішній стан (пам'ять), який зберігає інформацію про попередні елементи вхідної послідовності. Завдяки цьому RNN можуть аналізувати рядок тексту не як набір незалежних символів, а як цілісну структуру, де контекст впливає на розпізнавання кожного окремого елемента. У задачах OCR це особливо важливо під час роботи зі спотвореними, частково злитими або фрагментованими символами, коли ізольована класифікація дає хибні результати, тоді як аналіз послідовності дозволяє системі відновити правильне значення завдяки контекстним залежностям.

Найбільш поширеними модифікаціями рекурентних мереж, що застосовуються в OCR, є LSTM (Long Short-Term Memory) та GRU (Gated Recurrent Unit). Ці архітектури вирішують проблему згасання градієнта, характерну для класичних RNN, та забезпечують стабільне навчання на довгих послідовностях. LSTM і GRU містять спеціальні керуючі механізми — «вентилі» — які визначають, яку інформацію слід зберігати, а яку оновлювати, що робить їх більш гнучкими та точними у задачах розпізнавання тексту.

У практичних OCR-системах рекурентні моделі не функціонують ізольовано, а поєднуються зі згортковими мережами. CNN-шари виконують виділення локальних просторових ознак символів, тоді як RNN- або LSTM-шари аналізують послідовність цих ознак, забезпечуючи коректне сприйняття тексту як неперервної лінії. Саме такий підхід реалізовано в Tesseract OCR (версії 4 та вище), де використовується гібридна архітектура CNN+LSTM у поєднанні з механізмом CTC (Connectionist Temporal Classification). Така комбінація дозволяє системі розпізнавати текст без жорсткої сегментації на окремі символи та демонструє високу точність на документах зі спотвореннями, нерівномірними інтервалами або дефектами шрифту.

Таким чином, рекурентні нейронні мережі відіграють важливу роль у сучасних OCR-рішеннях, забезпечуючи моделювання контексту та підвищуючи стійкість систем до складних типів спотворень, які не можуть бути повністю компенсовані навіть високоефективними згортковими моделями [15].

Серед новітніх рішень також варто відзначити трансформерні моделі, які поступово знаходять застосування в задачах розпізнавання тексту завдяки своїй здатності обробляти контекст у широкому діапазоні даних. Хоча їх основне призначення — робота з природними мовами, ефективність трансформерів у виявленні закономірностей у зображеннях відкриває нові можливості для OCR-систем майбутнього.

Найбільш широке застосування у практичних рішеннях усе ж отримали саме згорткові мережі, оскільки вони демонструють оптимальне поєднання

точності, швидкодії та стійкості до спотворень. Їх архітектура ідеально підходить для обробки друкованих символів, де ключову роль відіграє аналіз форми та просторових відношень між елементами.

Згорткова нейронна мережа є складною багатошаровою системою, побудованою за принципом послідовного перетворення зображення від простих елементів до узагальнених структур. Її архітектура зазвичай включає кілька основних типів шарів: згорткові, підвибіркові (пулінгові), повнозв’язні та вихідні. Кожен із них виконує певну функцію у процесі навчання та розпізнавання символів. На рисунку 1.1 показана типова архітектура згорткової нейронної мережі

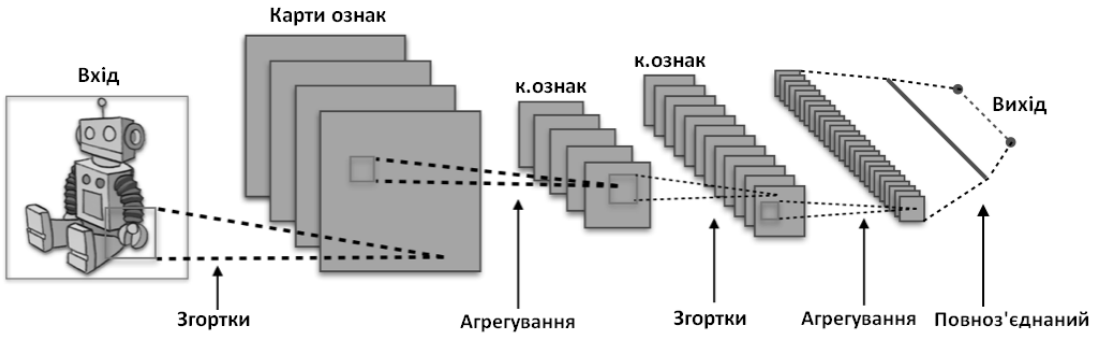


Рисунок 1.1 — Структура згорткової нейронної мережі

На початкових етапах обробки вхідне зображення проходить через згорткові шари, які аналізують невеликі ділянки — піксельні фрагменти. На цьому рівні мережа навчається виявляти базові елементи структури, такі як горизонтальні або вертикальні лінії, кути, дуги чи контури. Згодом ці прості ознаки комбінуються у складніші, утворюючи внутрішні представлення літер і цифр. Таким чином, кожен наступний шар узагальнює інформацію, наближаючи її до рівня абстракції, на якому відбувається розпізнавання образу.

Після згорткових шарів зазвичай розміщуються пулінгові шари, які зменшують розмірність даних, виділяючи найважливіші ознаки. Це дозволяє знизити обчислювальні витрати й одночасно підвищити стійкість моделі до незначних зсувів або деформацій зображення. У результаті мережа не

залежить від точного розташування символу, зберігаючи здатність правильно його класифікувати навіть у разі неідеального вирівнювання тексту.

Після кількох циклів згортки та пулінгу отримані характеристики передаються до повнозв'язних шарів, які виконують роль класифікатора. На цьому етапі система зіставляє узагальнені ознаки з певними класами символів і приймає рішення про те, який саме символ представлено на зображенні. У навчальному процесі нейронна мережа багаторазово порівнює свої передбачення з правильними відповідями, поступово коригуючи вагові коефіцієнти за допомогою алгоритму зворотного поширення помилки.

Завдяки такій структурі CNN здатна самостійно формувати багаторівневі представлення вхідних даних, що робить її особливо ефективною для аналізу друкованих текстів, де важливими є дрібні деталі та контекст взаєморозташування елементів. Висока точність, стійкість до шумів і можливість адаптації до різних шрифтів зробили згорткові мережі стандартом у більшості сучасних систем оптичного розпізнавання.

Хоча згорткові мережі ефективно виділяють локальні просторові ознаки символів, цього інколи недостатньо для точного розпізнавання друкованих текстів, особливо коли символи з'єднані, перекриті або містять спотворення. У таких випадках важливо враховувати не лише форму окремого символу, а й взаємозв'язок між послідовними елементами тексту. Цю задачу доповнюють рекурентні нейронні мережі, зокрема моделі типу LSTM та GRU, які здатні працювати з даними у вигляді послідовностей та утримувати інформацію про контекст попередніх елементів. Завдяки цьому рекурентна мережа може коригувати неоднозначні випадки класифікації, спираючись на загальну структуру рядка, а не лише на локальне зображення символу [16].

Поєднання згорткових і рекурентних мереж утворює гібридну архітектуру, у якій CNN-шари виконують функцію виділення інформативних ознак, а рекурентні моделі аналізують їхню послідовність у межах рядка тексту. Такий підхід дозволяє компенсувати недоліки кожної з архітектур окремо: CNN забезпечує стійкість до шумів, деформацій та варіацій шрифтів,

тоді як RNN покращує інтерпретацію структурно складних фрагментів, зокрема спотворених або частково злитих символів. На практиці гібридні моделі демонструють значно вищу точність розпізнавання у порівнянні з використанням тільки згорткової архітектури. Саме така комбінація — CNN для просторової обробки та LSTM для послідовного аналізу — лежить в основі сучасних OCR-систем, включно з Tesseract, де додатково застосовується механізм CTC, що усуває потребу в чіткій попередній сегментації символів.

Ефективність нейронної мережі у розпізнаванні тексту безпосередньо залежить від якості процесу її навчання. Основною метою цього етапу є формування внутрішніх зв’язків між вхідними ознаками зображення та правильними вихідними класами символів. Для цього використовується велика кількість попередньо підготовлених даних, які утворюють навчальну вибірку. Вона складається із зображень друкованих символів різних шрифтів, розмірів і ступенів спотворення, що дозволяє мережі навчитися розпізнавати символи в різних умовах.

Під час навчання кожне зображення подається на вхід мережі, яка формує прогноз щодо того, який символ воно відображає. Результат порівнюється з правильною відповіддю, а різниця між ними використовується для корекції вагових коефіцієнтів нейронів. Цей процес багаторазово повторюється для всієї вибірки, унаслідок чого модель поступово вдосконалює свої внутрішні параметри, зменшуючи похибку класифікації. Такий підхід називається навчанням з учителем, оскільки кожен приклад має правильну мітку, за якою мережа перевіряє власні рішення.

Важливу роль у навчальному процесі відіграє баланс між точністю і здатністю до узагальнення. Якщо модель занадто точно запам’ятовує приклади з вибірки, виникає ефект перенавчання, коли вона добре працює на знайомих даних, але не здатна правильно інтерпретувати нові зображення. Для уникнення цього використовуються спеціальні методи, такі як нормалізація, регуляризація або штучне розширення вибірки за рахунок обертання, масштабування та спотворення вихідних зображень. Таке «аугментування»

даних допомагає мережі навчитися розпізнавати символи за суттєво різних умов.

Після завершення навчання проводиться оцінювання якості роботи моделі на тестовій вибірці, яка не брала участі у процесі тренування. Для цього обчислюються показники точності, повноти, чутливості та середньої похибки класифікації. Аналіз цих метрик дозволяє визначити, наскільки ефективно нейронна мережа узагальнює знання й чи готова вона до використання у реальних системах розпізнавання.

Процес навчання нейронної мережі формує її здатність виявляти характерні ознаки символів та коректно інтерпретувати їх у ситуаціях, що виходять за межі навчального набору. Репрезентативність вибірки, вибір оптимальної архітектури та налаштування гіперпараметрів визначають баланс між точністю, узагальнювальною здатністю та стійкістю моделі до спотворень. У ході навчання важливо контролювати не лише зменшення помилки на тренувальних даних, а й поведінку мережі на валідованих прикладах, що дозволяє уникнути перенавчання і забезпечити стабільне функціонування під час практичного застосування. Таким чином, навчання виступає комплексним процесом, який включає оптимізацію моделі, оцінювання її надійності й перевірку здатності адаптуватися до реальних умов розпізнавання [17].

У даному розділі виконано огляд методів та програмних систем, що застосовуються для оптичного розпізнавання друкованих символів. Виконано порівняльний аналіз існуючих OCR-платформ, зокрема таких як Tesseract OCR, ABBYY FineReader та інших поширених рішень, із визначенням їхніх основних можливостей, обмежень та сфер ефективного застосування. Розглянуто також застосування нейронних мереж у системах оптичного розпізнавання тексту і показано, що вони є перспективним напрямом для розпізнавання тексту зі спотвореними або низькоякісними зображеннями, що є актуальною проблемою для більшості традиційних алгоритмів.

2 ПРОЄКТУВАННЯ ТЕХНОЛОГІЇ РОЗПІЗНАВАННЯ

2.1 Етапи та структура процесу розпізнавання друкованих символів

Процес розпізнавання спотворених друкованих символів є багатоступеневою процедурою, яка поєднує методи цифрової обробки зображень, машинного навчання та елементів штучного інтелекту. Його структура передбачає послідовне виконання етапів, кожен з яких спрямований на перетворення зображення тексту у машинно-читану форму з мінімальними втратами інформації. Послідовність операцій, які виконуються над зображенням, наведена на рисунку 2.1. У випадку роботи зі спотвореними символами особливого значення набуває точність попередньої обробки, адже саме вона забезпечує якість даних, що надходять на наступні етапи.

Рисунок 2.1 — Загальна схема розв’язання задачі розпізнавання

Першим етапом є отримання та нормалізація вхідного зображення. Джерелом може бути сканер, камера або будь-який інший пристрій, здатний

формувати цифрове представлення друкованого документа. На цьому етапі виконується переведення зображення у стандартний формат та масштаб, корекція орієнтації сторінки і вирівнювання контрасту. Для документів зі спотвореннями, спричиненими вигином паперу чи неякісним освітленням, застосовуються методи геометричної трансформації та фільтрації, які компенсують деформації зображення та забезпечують однорідність яскравості по всьому полю кадру [18].

Наступним кроком є попередня обробка зображення, спрямована на покращення його якості перед подальшим аналізом. Вона включає видалення шумів, підвищення контрастності, усунення фону, а також операції морфологічної фільтрації, що дозволяють згладити контури символів і відновити їхню структуру. Особливо важливою ця стадія є у випадках, коли друкований текст має пошкодження або забруднення, які можуть бути сприйняті системою як частини символів. Для цього використовуються медіанні та гаусові фільтри, алгоритми адаптивної бінаризації, а також методи вирівнювання гістограми яскравості.

Після покращення зображення виконується сегментація — розділення тексту на окремі структурні одиниці. Спочатку система виділяє області, що містять текст, потім — окремі рядки, слова та, зрештою, символи. Для коректної сегментації спотворених текстів використовуються комбіновані алгоритми, що враховують не лише геометричні межі, але й текстурні та контурні ознаки. У разі накладання символів або їх часткового пошкодження застосовуються методи кластеризації та аналізу зв’язності пікселів, які дозволяють розділяти злиті елементи.

Після сегментації система переходить до формування ознак, які описують кожен символ. Цей етап має ключове значення для подальшої класифікації, адже саме набір ознак визначає унікальність кожного символу. У разі спотворених символів традиційні геометричні характеристики можуть бути недостатніми, тому доцільно використовувати комбіновані підходи, що поєднують геометричні, частотні та статистичні параметри. Це дозволяє

створити більш стійке представлення символу, нечутливе до дрібних деформацій, нахилів або втрати частини контурів.

На етапі класифікації відбувається безпосереднє розпізнавання символів. Сучасні системи використовують нейронні мережі, які здатні самостійно визначати найінформативніші ознаки та узагальнювати знання, отримані під час навчання. Для розпізнавання спотворених символів особливо ефективними є згорткові нейронні мережі (Convolutional Neural Networks, CNN), оскільки вони враховують просторові зв’язки між пікселями та зберігають інформацію про локальні особливості зображення. Завдяки цьому мережа здатна правильно класифікувати навіть частково пошкоджені або деформовані символи.

Після класифікації результати розпізнавання проходять перевірку на рівні слів і фраз. Використання лінгвістичних моделей або словників дозволяє виявляти та виправляти помилки, пов’язані з неправильною ідентифікацією схожих символів. Така післяобробка особливо важлива для текстів із великою кількістю спотворень, де навіть незначна кількість помилкових класифікацій може призвести до втрати сенсу тексту.

Загалом структура процесу розпізнавання спотворених друкованих символів охоплює всі основні етапи — від отримання зображення до формування коректного текстового результату. Вона включає взаємопов’язані підпроцеси: нормалізацію зображення, попередню обробку, сегментацію, формування ознак, класифікацію та корекцію результатів. Кожен із них відіграє ключову роль у забезпеченні точності й надійності роботи системи. Саме узгоджене функціонування цих етапів дозволяє створити ефективну технологію розпізнавання символів навіть у складних умовах, коли зображення містить дефекти, шуми або деформації [19].

Взаємозв’язок між окремими етапами процесу розпізнавання друкованих символів визначає ефективність функціонування всієї системи. Кожен етап не є самостійним — його результат слугує основою для виконання наступних операцій. Тому від якості попередніх дій безпосередньо залежить

точність класифікації символів і кінцева якість отриманого тексту. Особливо це стосується обробки спотворених зображень, де навіть незначні похибки на ранніх етапах здатні спричинити ланцюгові помилки на подальших стадіях.

Після отримання цифрового зображення виконується нормалізація, яка приводить його до єдиних параметрів масштабу, роздільної здатності та орієнтації. Це дозволяє мінімізувати вплив зовнішніх факторів, таких як кут нахилу або нерівномірне освітлення. У системах, що працюють зі спотвореними документами, на цьому етапі додатково застосовуються алгоритми геометричної корекції, які усувають викривлення, спричинені вигином або деформацією сторінки. Таким чином формується стабільна основа для подальшої обробки.

Наступним кроком є підвищення якості зображення, що забезпечує виділення чітких контурів символів навіть за наявності шумів чи нерівномірного фону. Попередня обробка покликана покращити співвідношення сигнал/шум, зробити контури символів більш контрастними та відокремити текст від фону. Це особливо важливо при роботі з архівними документами або відбитками з дефектами друку. Якщо на цьому етапі не досягнуто належного рівня якості, подальше розпізнавання може виявитися неможливим або давати значні похибки.

Результатом попередньої обробки є зображення, готове до сегментації. Саме на цьому етапі система виконує логічне структурування документа — виділення текстових блоків, рядків, слів і символів. Процес сегментації базується на аналізі просторових характеристик пікселів, їхніх проекцій та зв’язності. Для роботи зі спотвореними символами застосовуються алгоритми, здатні адаптуватися до змін товщини ліній і незначних викривлень. Це забезпечує правильне розділення навіть у випадках часткового злиття сусідніх символів або наявності фонових артефактів.

На основі отриманих фрагментів формується набір ознак, який описує внутрішню структуру кожного символу. Мета цього етапу — перетворити зображення у компактну числову форму, придатну для подальшої

класифікації. У системах, що працюють зі спотвореними символами, виділення ознак виконується з урахуванням локальних варіацій форми, контурів і тіней, які не впливають на зміст, але можуть ускладнювати розпізнавання. Використання комбінованих ознак — геометричних, статистичних та частотних — дозволяє зберегти інформацію про ключові властивості символів і водночас знизити чутливість до спотворень.

Отримані ознаки надходять на класифікаційний модуль, де відбувається безпосереднє розпізнавання. На цьому етапі використовується модель, навчена на великій вибірці прикладів, що включає як стандартні, так і спотворені символи. Завдяки цьому система здатна визначати належність символу до певного класу навіть у випадках, коли його форма частково втрачена або змінена. У сучасних реалізаціях найчастіше застосовуються згорткові нейронні мережі, які забезпечують стійкість до геометричних варіацій і локальних деформацій.

Завершальним етапом є перевірка та корекція результатів розпізнавання. Для цього застосовуються мовні моделі, словникові бази або алгоритми контекстного аналізу, які дозволяють виправляти помилки, що виникли під час класифікації. Наприклад, якщо певний символ був розпізнаний неоднозначно, система перевіряє його відповідність у контексті слова чи речення та замінює на найбільш ймовірний варіант. Такий підхід дозволяє суттєво підвищити точність результатів, особливо при роботі з текстами, що містять велику кількість пошкоджених символів [20].

Таким чином, процес розпізнавання спотворених друкованих символів можна розглядати як взаємопов’язану послідовність дій, у якій кожен етап має визначене функціональне призначення і логічно продовжує попередній. Ефективність системи залежить від узгодженості цих етапів та здатності кожного з них компенсувати недоліки попереднього. Висока якість попередньої обробки забезпечує успішну сегментацію, правильне формування ознак сприяє точній класифікації, а післяобробка гарантує отримання змістовно коректного тексту. Така структура створює основу для побудови

універсальної технології, здатної ефективно розпізнавати навіть складні, спотворені або частково пошкоджені друковані символи.

2.2 Послідовність дій та взаємозв’язок етапів обробки

Процес розпізнавання спотворених друкованих символів реалізується як послідовність взаємопов’язаних етапів, що забезпечують поступове перетворення зображення тексту у цифрову форму. Кожен з етапів має своє функціональне призначення та впливає на якість кінцевого результату. При цьому важливо не лише виконання кожного етапу окремо, а й їх узгодженість у загальній технологічній схемі. Порушення логічної послідовності або неточності на одному з кроків призводять до накопичення помилок у подальших стадіях обробки.

Перший етап — отримання та нормалізація зображення текстового документа. Він визначає початкову якість даних, з якими надалі працює система розпізнавання. Зазвичай вхідне зображення формується за допомогою сканера або цифрової камери, причому його параметри можуть суттєво відрізнятися залежно від умов зйомки чи сканування. На цьому етапі здійснюється переведення зображення до стандартного формату з єдиними параметрами роздільної здатності, масштабу та орієнтації. Для підвищення точності подальших обчислень виконується геометрична корекція, що усуває нахили та викривлення сторінки.

У разі роботи зі спотвореними документами нормалізація включає додаткові процедури вирівнювання освітлення та корекції контрасту. Нерівномірна яскравість, тіні або відблиски можуть створювати фонові артефакти, які пізніше розпізнаються як частини символів. Для усунення таких ефектів використовуються методи локального адаптивного балансування контрасту (CLAHE) та логарифмічної корекції. Вони дозволяють відновити рівномірність яскравості по всьому зображенню, не втрачаючи дрібних деталей тексту.

Особливу роль на цьому етапі відіграє підвищення чіткості контурів. Використання фільтрів високих частот дає змогу підсилити переходи між фоном і символами, що покращує подальше виділення ознак. У випадках, коли друкований текст має слабкий контраст або символи частково стерті, додатково застосовуються методи покращення локального контрасту та фільтрації на основі вейвлет-перетворення. Це дозволяє виділити структуру символів навіть за умови їх часткового пошкодження.

Важливим завданням етапу є також виправлення геометричних спотворень, спричинених неідеальним положенням документа під час сканування або фотографування. Якщо сторінка має викривлення, система визначає основні лінії тексту за допомогою методу Хафа або аналізу горизонтальних проекцій і виконує перетворення зображення для вирівнювання рядків. Такий підхід забезпечує коректну орієнтацію тексту, що є критично важливим для точності подальшої сегментації.

Результатом першого етапу є зображення текстового документа з виправленими викривленнями, відфільтрованими шумами, нормалізованим контрастом та оптимальними параметрами для подальшої обробки. Цей проміжний результат створює основу для виконання наступних дій — виділення текстових областей, сегментації та розпізнавання символів. Від точності виконання початкової нормалізації безпосередньо залежить успішність усього процесу, адже навіть найсучасніші алгоритми класифікації не зможуть компенсувати помилки, пов’язані з неправильною орієнтацією або низькою якістю початкового зображення.

Другим етапом є попередня обробка зображення, яка має на меті покращення його візуальної якості та підготовку до подальших етапів розпізнавання. Цей етап є одним із найважливіших у всьому процесі, адже саме він визначає, наскільки точно система зможе виділити символи та провести їх подальшу класифікацію. Для спотворених текстових документів якість попередньої обробки безпосередньо впливає на рівень точності всього розпізнавання.

Основною задачею попередньої обробки є усунення шумів та фонових артефактів, які можуть сприйматися алгоритмом як частини тексту. Для цього застосовуються методи фільтрації, що дозволяють зберегти контури символів, одночасно видаляючи дрібні випадкові пікселі. Найчастіше використовуються медіанні, гаусові та двосторонні фільтри, які ефективно пригнічують адитивний шум, не розмиваючи меж текстових елементів. У разі наявності точкових дефектів або плям може бути застосована морфологічна обробка з використанням операцій ерозії та дилатації, що забезпечують очищення фону без втрати основної інформації [21].

Наступним кроком є вирівнювання яскравості та контрасту. У спотворених документах часто спостерігається нерівномірне освітлення або тіні, що робить деякі ділянки темнішими, а інші — надмірно світлими. Для корекції таких ефектів використовується адаптивна нормалізація яскравості, яка підлаштовує контрастність зображення в межах локальних областей. Це дозволяє досягти рівномірного освітлення по всій площі документа й зробити символи більш помітними навіть на слабоконтрастному фоні.

Після нормалізації контрасту виконується бінаризація зображення — перетворення його у двійкову форму, де кожен піксель має значення «чорний» або «білий». Такий підхід значно спрощує подальшу обробку та дозволяє чітко відокремити текст від фону. У випадку з неякісними або спотвореними текстами застосовуються адаптивні алгоритми бінаризації, такі як методи Оцу, Ніблека, Саула або Бернсена. Вони враховують локальні варіації яскравості, що дозволяє отримати якісний результат навіть при нерівномірному освітленні сторінки.

Важливу роль у цьому етапі відіграє видалення тіней та підсилення меж символів. Для цього використовуються методи корекції фону та операції морфологічного відкриття, які дозволяють видаляти великі фонові неоднорідності, не зачіпаючи тонкі елементи символів. У випадку низької контрастності тексту можуть застосовуватися методи контурного підсилення,

які підкреслюють переходи між символами та фоном, забезпечуючи більш чітке відтворення структури знаків.

Особливої уваги потребують зображення, де символи мають деформації або часткові пошкодження. Для таких випадків ефективними є методи згладжування та інтерполяції, які відновлюють безперервність ліній, що утворюють контури символів. Застосування алгоритмів субпіксельної реконструкції дозволяє зберегти правильну форму навіть при зменшенні роздільної здатності зображення. Це особливо важливо для систем, що працюють з текстами, отриманими з фотографій або старих документів.

Результатом попередньої обробки є очищене, контрастне та чітке зображення, на якому межі символів мають однорідну товщину, а фон є рівномірним. Таке зображення є оптимальним для подальшого етапу — сегментації тексту, що полягає у виділенні структурних елементів документа: рядків, слів і окремих символів. Саме від якості попередньої обробки залежить точність цього процесу, оскільки навіть мінімальні залишки шумів або викривлення меж можуть призвести до хибного розділення символів або їх об’єднання у єдині контури.

Третім етапом процесу розпізнавання є сегментація текстового зображення, яка полягає у послідовному виділенні структурних елементів документа — текстових блоків, рядків, слів і окремих символів. Цей етап є ключовим для подальшого розпізнавання, оскільки саме тут відбувається перехід від суцільного зображення до впорядкованої структури, де кожен символ може бути оброблений окремо. Для систем, що працюють із спотвореними документами, сегментація має підвищену складність, адже деформації, шуми або розмиття можуть призводити до часткового накладання або об’єднання символів.

Процес сегментації починається з локалізації текстових областей. На цьому етапі алгоритм визначає, які частини зображення містять текст, а які — графічні елементи, лінії або фонові ділянки. Для цього використовуються методи аналізу текстури та гістограм яскравості, що дозволяють відрізнити

області з високою концентрацією контрастних елементів (властивих тексту) від однорідних фрагментів фону. У разі документів зі складною структурою може застосовуватися класифікація блоків за допомогою згорткових нейронних мереж, які розрізняють текст, зображення й таблиці за контекстними ознаками.

Після виділення текстових блоків виконується сегментація рядків, тобто поділ тексту на горизонтальні смуги. Зазвичай цей етап реалізується на основі горизонтальної проекції пікселів або аналізу зв’язності компонент. У спотворених документах рядки можуть бути вигнутими або нахиленими, тому для їх точного виявлення використовуються методи вирівнювання за допомогою перетворення Хафа або фільтрації на основі морфологічних профілів. Виправлення нахилу рядків дозволяє уникнути помилок під час подальшого розбиття на слова.

Наступним кроком є виділення окремих слів у межах рядка. У друкованих текстах це завдання зазвичай виконується шляхом аналізу відстаней між групами пікселів: якщо проміжок між ними перевищує певний поріг, система вважає його роздільником між словами. Однак при обробці спотворених або нерівномірно надрукованих документів такі відстані можуть бути нестабільними. Для розв’язання цієї проблеми застосовуються адаптивні методи, що враховують локальні відхилення інтервалів, а також алгоритми кластеризації, які дозволяють групувати символи за просторовою близькістю.

Останній і найвідповідальніший етап сегментації — виділення окремих символів. Цей процес потребує особливої точності, оскільки саме на цьому рівні формуються вхідні дані для класифікації. У найпростішому випадку символи відокремлюються за вертикальною проекцією, але для спотворених текстів цього недостатньо. Часто символи можуть бути злиті, деформовані або частково стерті, тому для їх коректного виділення використовуються комбіновані методи: аналіз контурів, визначення мінімальних відстаней між компонентами, побудова зв’язних областей та морфологічне розділення.

Додатково застосовуються алгоритми скелетизації (thinning), які дозволяють зменшити символ до однопіксельного каркасу без втрати його топологічної структури. Це спрощує подальше визначення характерних ознак, таких як кількість петель, точок розгалуження або напрямків ліній. У випадках, коли символи частково перекриваються, може використовуватись аналіз геометричних моментів або методи контурного відновлення для відокремлення накладених елементів.

Для складних випадків, коли сегментація не може бути виконана однозначно, застосовується попереднє розпізнавання фрагментів, за якого нейронна мережа оцінює ймовірність того, що певна область відповідає конкретному символу. Якщо кілька суміжних фрагментів мають високу подібність до різних символів, система розділяє їх та формує окремі зони. Такий підхід дозволяє підвищити точність розпізнавання навіть при значному рівні пошкоджень тексту.

Результатом етапу сегментації є впорядкований набір фрагментів зображення, кожен з яких відповідає окремому символу або його частині. Ці фрагменти передаються на наступний етап — формування ознак, що описують їх геометричну, структурну й контурну інформацію. Від коректності сегментації залежить не лише точність розпізнавання, але й загальна швидкодія системи, оскільки помилки на цьому етапі можуть призвести до неправильного навчання або класифікації нейронної мережі [22].

Четвертим етапом процесу розпізнавання є формування ознак та підготовка даних до класифікації, який забезпечує перетворення графічного зображення символу у числову форму, зрозумілу для алгоритмів машинного навчання. Цей етап має ключове значення, оскільки саме якість сформованих ознак визначає здатність системи точно відрізняти один символ від іншого, особливо у випадках, коли їхні форми спотворені або частково пошкоджені.

Основна мета формування ознак полягає у виділенні найбільш інформативних характеристик символу, що описують його структуру, форму, топологію та контрастні співвідношення. Для цього зображення кожного

символу піддається аналізу, в результаті якого обчислюються різноманітні параметри — геометричні, статистичні, контурні або частотні. Отримані значення формують вектор ознак, який у подальшому подається на вхід класифікатора. Для спотворених символів важливо, щоб ці ознаки були стійкими до змін масштабу, нахилу, товщини ліній та шумів, що часто супроводжують процес сканування чи фотографування.

На практиці найчастіше використовуються три основні групи ознак.

Перша група — геометричні ознаки, які відображають пропорції та просторові співвідношення елементів символу: висоту, ширину, співвідношення площі заповнення до загальної площі, кількість замкнених контурів, відстані між ключовими точками тощо. Такі характеристики є базовими, однак можуть втрачати точність у разі сильних деформацій або нерівномірного друку.

Друга група — структурні ознаки, які описують взаємне розташування ліній, дуг, точок розгалуження та інших елементів форми. Для їх визначення часто застосовується скелетизація символу, що дозволяє отримати його каркас і на основі нього сформувати структурну модель. Структурні ознаки добре зберігають топологічні властивості об’єкта навіть за умов пошкодження частини контурів, тому їх використання є доцільним при роботі зі спотвореними символами.

Третя група — частотні та статистичні ознаки, що базуються на аналізі інтенсивності пікселів та їхнього розподілу. Вони можуть бути отримані шляхом перетворення Фур’є або вейвлет-аналізу, що дозволяє описати текстуру символу незалежно від його орієнтації. Такі ознаки особливо ефективні для відмінності схожих символів (наприклад, «O» і «0» або «I» і «l») та забезпечують високу точність класифікації.

Важливою особливістю етапу є зменшення розмірності простору ознак. Оскільки кожен символ може мати десятки чи навіть сотні характеристик, надлишкова інформація збільшує обчислювальні витрати та може знижувати точність класифікації через ефект «перенавчання». Для оптимізації ознак

застосовуються методи головних компонент (PCA), лінійний дискримінантний аналіз (LDA) або відбір за критерієм інформаційної вагомості. У результаті формується компактний, але максимально інформативний вектор, який зберігає усю необхідну інформацію для подальшого розпізнавання.

Особливе значення при роботі зі спотвореними символами має нормалізація даних. Вона полягає у приведенні значень пікселів та векторів ознак до єдиних діапазонів, що дозволяє уникнути перекосу під час навчання нейронної мережі. Крім того, для підвищення стійкості системи використовується розширення навчальної вибірки (data augmentation), коли кожен зразок символу доповнюється його модифікованими копіями — з невеликим нахилом, шумом або масштабною трансформацією. Це дозволяє алгоритму навчитися правильно розпізнавати символи навіть при суттєвих відхиленнях від стандартної форми.

Результатом етапу формування ознак є векторне представлення символів, що описує їх найсуттєвіші характеристики у компактній математичній формі. Ці дані є вхідними для класифікатора, який виконує безпосереднє розпізнавання. Висока якість ознак дозволяє нейронній мережі швидко навчатися та забезпечує стійкість системи до різних типів спотворень. Таким чином, етап формування ознак є критично важливою ланкою у загальній структурі процесу оптичного розпізнавання, що поєднує попередню обробку з етапом класифікації та безпосереднього визначення символів.

П’ятим і завершальним етапом процесу є класифікація та післяобробка результатів розпізнавання, які забезпечують перетворення набору ознак у послідовність символів і формування готового тексту. На цьому етапі система переходить від аналітичної обробки зображення до прийняття рішень, визначаючи, якому класу належить кожен символ, та перевіряє правильність отриманого результату з урахуванням контексту.

Класифікація символів полягає у порівнянні сформованих векторів ознак із навченою моделлю, яка містить інформацію про відомі зразки. У

традиційних системах OCR для цього застосовувалися методи шаблонного або статистичного порівняння, однак їхня ефективність суттєво знижувалася при роботі зі спотвореними або пошкодженими символами. Сучасні технології використовують підходи машинного навчання, насамперед згорткові нейронні мережі (Convolutional Neural Networks, CNN), які здатні автоматично виявляти найінформативніші ознаки та формувати внутрішні уявлення про структуру символів.

Перевагою CNN є їхня стійкість до варіацій масштабу, нахилу, освітлення та часткових пошкоджень. Мережа не потребує попереднього визначення ознак — вона самостійно навчається знаходити закономірності у даних, що дозволяє їй ефективно працювати навіть зі складними спотвореннями. У процесі навчання формується набір вагових коефіцієнтів, які визначають реакцію мережі на певні елементи структури символів. Завдяки цьому під час розпізнавання система може коректно ідентифікувати навіть ті символи, які не зустрічалися в навчальній вибірці, але мають подібні ознаки.

У деяких випадках для підвищення точності використовуються комбіновані класифікатори, які поєднують нейронні мережі з методами статистичної або ймовірнісної оцінки. Наприклад, після первинного визначення символу мережею додатково виконується перевірка його належності до конкретного класу за допомогою методу найближчих сусідів або баєсівського підходу. Такий підхід дозволяє мінімізувати кількість помилкових класифікацій, особливо у випадках, коли символи мають схожу форму або низьку контрастність.

Після завершення класифікації виконується післяобробка результатів, спрямована на перевірку, корекцію та оптимізацію отриманого тексту. Цей етап має на меті усунути залишкові помилки, спричинені спотвореннями, неякісним друком або неоднозначністю символів. Для цього застосовуються лінгвістичні та контекстні моделі, які аналізують взаємозв’язки між символами та словами. Система перевіряє розпізнані послідовності за

словником або мовною моделлю, що дозволяє виправляти найімовірніші помилки — наприклад, заміну літери «O» на цифру «0» чи «I» на «l».

Крім лінгвістичної перевірки, післяобробка може включати семантичний аналіз, який враховує контекст у межах речення або документа. Наприклад, якщо певне слово не відповідає граматичним правилам або не узгоджується зі змістом попередніх слів, система пропонує альтернативний варіант. Для цього використовуються алгоритми на основі марковських моделей або трансформерних архітектур, які дозволяють прогнозувати найбільш вірогідну послідовність символів і слів.

Важливим аспектом післяобробки є усунення форматних помилок, що виникають при розпізнаванні складних документів. До них належать пропущені пробіли, злиття слів, зміщення розділових знаків або неправильне відтворення великих і малих літер. Такі помилки виправляються за допомогою правил форматування, які автоматично застосовуються до всього тексту. У результаті формується структурований, граматично і візуально коректний документ, готовий до подальшого використання.

Завершальним кроком є оцінювання якості розпізнавання, яке виконується шляхом порівняння отриманого тексту з еталонним або шляхом статистичного аналізу кількості помилок. Цей етап дозволяє визначити точність роботи системи та ефективність застосованих алгоритмів. Для спотворених документів часто використовується метрика CER (Character Error Rate), яка показує відсоток неправильно розпізнаних символів від загальної кількості. На основі цих даних система може виконувати повторне навчання або корекцію параметрів моделі для підвищення точності [23].

Таким чином, етап класифікації та післяобробки завершує технологічний цикл розпізнавання спотворених друкованих символів. Він забезпечує перехід від цифрового зображення до осмисленого текстового результату, у якому виправлено помилки та збережено структуру початкового документа. Узгоджена робота всіх попередніх етапів — від нормалізації та фільтрації до формування ознак і класифікації — формує єдину систему,

здатну надійно розпізнавати навіть складні й пошкоджені текстові матеріали. Саме така послідовність дій забезпечує високу ефективність сучасних технологій оптичного розпізнавання, побудованих на основі методів глибинного навчання.

2.3 Вибір архітектури нейронної мережі для реалізації процесу розпізнавання

Ефективність системи розпізнавання спотворених друкованих символів значною мірою визначається архітектурою нейронної мережі, яка використовується для аналізу вхідних зображень. На відміну від класичних алгоритмів, що ґрунтуються на попередньо заданих ознаках, нейронна мережа здатна самостійно навчатися виділяти інформативні характеристики символів, що підвищує точність і стійкість до спотворень.

Основна мета вибору архітектури полягає у створенні моделі, яка забезпечує достатню глибину для розпізнавання складних закономірностей, але при цьому не потребує надмірних обчислювальних ресурсів. Оптимальним рішенням для вирішення цього завдання є використання згорткової нейронної мережі (Convolutional Neural Network, CNN), що поєднує властивості просторового аналізу та інваріантність до деформацій і шумів.

Задачу розпізнавання можна подати у загальному вигляді як функціональне відображення:

$$f_\theta: X \to Y,$$

де X — множина зображень символів;

Y — множина класів;

θ — параметри моделі (ваги та зміщення).

Метою навчання є знаходження таких параметрів θ*, які мінімізують функцію втрат L(θ):

$$\theta^* = arg\min \mathrm{L}(\theta),$$

де L(θ) визначає різницю між реальними та прогнозованими класами.

У системі розпізнавання спотворених символів CNN виконує послідовне перетворення зображення:

— згорткові шари виділяють локальні ознаки (контури, лінії, кути);

— шари субдискретизації (pooling) узагальнюють інформацію, зменшуючи вплив дрібних викривлень;

— повнозв'язні шари виконують інтеграцію отриманих ознак і класифікацію символів.

Таке поєднання дозволяє мережі відновлювати структуру символу навіть за наявності дефектів, незначних нахилів або нерівномірного освітлення.

Під час навчання мережа коригує свої параметри за допомогою алгоритму градієнтного спуску, який оновлює ваги на основі похідної функції втрат:

$$\theta_{t+1} = \theta_t - \eta\nabla_\theta L(\theta_t),$$

де η — швидкість навчання;

$\eta\nabla_\theta L(\theta_t)$— градієнт помилки.

Цей процес повторюється до моменту, коли похибка класифікації досягає мінімального значення.

Для досягнення високої точності важливим етапом є підготовка даних: нормалізація зображень, їх масштабування та розширення вибірки (data augmentation), яке імітує різні типи спотворень — обертання, нахили, шум або зміну контрасту. Це дозволяє сформувати модель, стійку до реальних умов сканування.

У результаті нейронна мережа навчається узагальнювати характерні риси символів, що дає змогу зберігати правильність розпізнавання навіть за

наявності деформацій. Саме тому архітектура CNN є базовою основою для подальшого побудування програмного модуля розпізнавання, який реалізовано в практичній частині роботи.

Архітектура згорткової нейронної мережі є багаторівневою системою, у якій кожен шар виконує певну функцію — від первинного виділення простих елементів до узагальнення складних структур. У процесі проходження зображення через мережу воно послідовно перетворюється з набору пікселів на компактне числове представлення, що описує сутність символу. Такий підхід дозволяє моделі автоматично формувати ознаки, які є стійкими до геометричних спотворень, зміни розміру, нахилу або контрастності [24].

У типовій архітектурі CNN виділяють кілька основних груп шарів.

Згорткові шари (Convolutional layers) — виконують виявлення локальних ознак. На початкових рівнях вони розпізнають прості елементи, як-от контури чи кути, а на глибших — складні комбінації форм, характерних для конкретних символів.

Шари субдискретизації (Pooling layers) — зменшують розмірність даних і підвищують інваріантність до дрібних спотворень, об’єднуючи сусідні пікселі у більші регіони.

Активаційні шари — вводять у модель нелінійність, що дає змогу мережі описувати складні залежності між ознаками.

Повнозв’язні шари (Fully connected layers) — формують остаточне узагальнення ознак і здійснюють класифікацію символів.

Кожен згортковий шар у мережі складається з набору фільтрів (ядер згортки), що аналізують зображення з різних ракурсів. Кожен фільтр створює карту ознак, яка відображає активність певного патерну — наприклад, горизонтальної лінії або замкненого контуру. Отримані карти потім узагальнюються на наступних рівнях, формуючи багаторівневу структуру подання даних. Такий підхід дозволяє моделі не просто «бачити» символ, а й «розуміти» його форму незалежно від конкретного положення чи спотворення.

Один із найважливіших етапів у структурі CNN — це застосування активаційної функції, яка додає нелінійність і дає змогу мережі моделювати складні взаємозв'язки між елементами зображення. Найпоширенішою є функція ReLU (Rectified Linear Unit), що визначається виразом:

$$f(x) = max(0, x).$$

Вона спрощує процес навчання, запобігає зникненню градієнтів і прискорює збіжність моделі.

Після кожного згорткового шару часто використовується операція субдискретизації (max pooling), яка зменшує кількість параметрів та підвищує стійкість мережі до дрібних змін на зображенні. Наприклад, якщо символ зміщений на кілька пікселів або має нерівномірну товщину лінії, pooling дозволяє зберегти його загальну форму без втрати ключових ознак.

Після проходження через кілька таких блоків формується узагальнене представлення символу, що передається у повнозв'язні шари. Вони інтегрують інформацію, отриману від попередніх рівнів, і формують кінцевий вектор ознак, який використовується для класифікації. Завершальний шар виконує обчислення ймовірностей належності зображення до певного класу символів за допомогою функції Softmax:

$$\hat{y}_k = \frac{e^{z_k}}{\sum_{j=1}^{K} e^{z_k}},$$

де $\hat{y}_k$ — імовірність належності символу до класу k;

K — загальна кількість класів.

Використання саме такої архітектури дозволяє досягти оптимального балансу між точністю розпізнавання та швидкодією, що особливо важливо при роботі з великими обсягами документів. CNN ефективно узагальнює ознаки навіть за умов низької якості зображення, а глибина мережі забезпечує

можливість розпізнавання символів, які частково перекриті, стерті або нахилені. Завдяки цьому технологія є універсальною для обробки друкованих текстів різних шрифтів і ступенів спотворення.

Під час розроблення нейронної мережі для розпізнавання спотворених друкованих символів важливим етапом є вибір оптимальних параметрів архітектури, які забезпечують необхідний баланс між точністю класифікації та швидкодією. Неправильний підбір кількості шарів, розміру фільтрів або глибини мережі може призвести до надмірного навантаження на обчислювальні ресурси або, навпаки, до втрати важливих ознак зображення.

У цій роботі для побудови моделі було обрано згорткову нейронну мережу середньої глибини, яка включає кілька послідовних згорткових і pooling-шарів, за якими слідують повнозв'язні шари для класифікації. Така конфігурація є оптимальною для задач розпізнавання друкованих символів, оскільки дозволяє враховувати локальні деталі зображення та поступово узагальнювати їх до вищих рівнів абстракції.

На першому етапі мережа приймає вхідне зображення розміром 32×32 пікселі у відтінках сірого. Невеликий розмір обрано для зменшення обчислювальної складності, водночас він зберігає достатню кількість інформації для розпізнавання навіть дрібних деталей символів. Перед подачею в мережу зображення нормалізуються за інтенсивністю, що усуває різницю у контрасті між різними документами:

$$x' = \frac{x - \mu}{\sigma},$$

де x — значення яскравості пікселя;

μ — середнє значення;

σ — стандартне відхилення.

Така нормалізація прискорює навчання і запобігає перенасиченню активацій у глибоких шарах.

Перший згортковий шар містить 32 фільтри розміром 3×3, які виявляють базові ознаки, такі як лінії, кути чи краї символів. Наступний шар max pooling із вікном 2×2 зменшує розмірність карти ознак удвічі, підвищуючи стійкість до дрібних спотворень. Другий згортковий шар має 64 фільтри 3×3, які виявляють більш складні елементи структури символів — криві, петлі або їх комбінації. Ще один pooling-шар зменшує обсяг даних, після чого виконується перехід до повнозв'язної частини мережі.

На етапі узагальнення ознак застосовується повнозв'язний шар із 128 нейронів, який інтегрує інформацію з попередніх рівнів і готує дані до фінальної класифікації. Для запобігання перенавчанню використовується механізм Dropout з імовірністю вимкнення частини нейронів (приблизно 0.5). Це дозволяє мережі уникати залежності від окремих з'єднань і покращує здатність до узагальнення результатів.

Вихідний шар містить стільки нейронів, скільки класів символів потрібно розпізнавати. Для перетворення вихідних активацій у ймовірнісну форму застосовується функція Softmax, що дозволяє інтерпретувати результат як імовірність належності символу до певного класу.

Ключовими параметрами, що впливають на якість розпізнавання, є:

— кількість фільтрів у згорткових шарах — визначають рівень деталізації ознак;

— розмір фільтрів — впливає на масштаб аналізу локальних структур символів;

— глибина мережі — визначає складність моделі та її здатність узагальнювати;

— параметри Dropout і нормалізації — забезпечують стійкість до перенавчання;

— швидкість навчання (learning rate) — контролює темп оновлення ваг під час оптимізації.

Підібрана конфігурація забезпечує рівновагу між точністю і швидкодією, що особливо важливо при обробці великих наборів даних або під

час роботи в реальному часі. Така архітектура легко адаптується до різних умов — наприклад, може бути масштабована для більших зображень або доопрацьована для розпізнавання нових типів символів без зміни базової структури [25].

Після визначення архітектури нейронної мережі наступним етапом є її навчання, під час якого система набуває здатності розпізнавати символи на основі аналізу великої кількості прикладів. Навчання є процесом поступового коригування вагових коефіцієнтів мережі з метою мінімізації різниці між фактичними та прогнозованими результатами.

Для цього формується навчальна вибірка, яка містить тисячі прикладів друкованих символів різних шрифтів, розмірів і ступенів спотворення. Кожне зображення подається на вхід моделі разом із відомою міткою класу. Щоб забезпечити стійкість до реальних умов сканування, навчальні дані проходять попереднє перетворення — масштабування, нормалізацію та штучне розширення (data augmentation). Це включає операції обертання, зміну контрасту, нахил або додавання шумів, які імітують дефекти друку чи нерівномірне освітлення. Такий підхід дозволяє моделі навчитися розпізнавати символи навіть у разі часткових деформацій.

У процесі навчання мережа мінімізує функцію втрат (Loss function), яка визначає, наскільки отримані результати відрізняються від правильних. Для задачі багатокласової класифікації використовується крос-ентропія:

$$\mathrm{L}(\theta) = -\frac{1}{N}\sum_{i=1}^{N}\sum_{k=1}^{K} y_i, k \log(\widehat{y_i}, k),$$

де N — кількість прикладів у навчальній вибірці;

K — кількість класів;

y_i, k — істинне значення (1, якщо приклад належить до класу k);

$\widehat{y_i}, k$ — прогнозована ймовірність моделі.

Для мінімізації функції втрат застосовується алгоритм оптимізації Adam, який є модифікацією стохастичного градієнтного спуску. Його перевага полягає у здатності автоматично адаптувати швидкість навчання для кожного параметра окремо, що забезпечує швидку збіжність і стабільність результатів навіть на шумних даних.

Процес навчання відбувається ітераційно — мережа аналізує вхідні дані, обчислює похибку, а потім коригує ваги відповідно до знайденого градієнта. Після кожної епохи виконуються обчислення точності та функції втрат як на навчальній, так і на валідаційній вибірці, що дозволяє контролювати появу перенавчання (overfitting). Якщо валідаційна похибка починає зростати, а навчальна — зменшуватися, це свідчить про надмірну адаптацію моделі до тренувальних даних, і процес навчання коригується зменшенням швидкості навчання або збільшенням коефіцієнта Dropout.

Для підвищення точності результати навчання оцінюються за такими показниками, як accuracy (точність класифікації), precision (точність по класах) та recall (повнота). Ці метрики дають змогу оцінити не лише середню правильність, а й збалансованість класифікації для всіх типів символів.

Завдяки правильно підібраному алгоритму навчання та якісно сформованій вибірці мережа поступово набуває здатності розрізняти символи навіть за складних умов. Результатом цього етапу є навчена модель з оптимальними ваговими коефіцієнтами, готова до подальшого тестування та використання в системі розпізнавання.

Після завершення навчання нейронної мережі виконується етап перевірки її ефективності, метою якого є оцінити здатність моделі правильно розпізнавати символи, які не входили до навчальної вибірки. Цей етап є ключовим для підтвердження узагальнюючої здатності системи та перевірки її стійкості до різних типів спотворень.

Для тестування використовується окрема контрольна вибірка (test set), яка складається із зображень, що не брали участі в процесі навчання. Символи в ній можуть мати різні масштаби, кути нахилу, дефекти або рівні контрасту.

Такий підхід дозволяє змоделювати реальні умови роботи системи при обробці сканованих документів.

У процесі перевірки мережа отримує на вхід кожне зображення та генерує прогнозований клас. Результати порівнюються з істинними значеннями, після чого обчислюються основні показники якості. Найважливішою метрикою є точність класифікації (accuracy), яка визначається відношенням кількості правильно розпізнаних символів до загальної кількості прикладів:

$$Accuracy = \frac{N_{correct}}{N_{total}} \times 100\% .$$

Цей показник дає загальне уявлення про ефективність роботи моделі, однак не враховує можливі відмінності у складності окремих класів.

Для детальнішої оцінки використовуються додаткові метрики — precision (точність), recall (повнота) та F1-score. Вони дозволяють оцінити якість розпізнавання з урахуванням помилкових позитивних і негативних результатів. Зокрема, precision показує, яка частка символів, розпізнаних як певний клас, є справді правильними, тоді як recall характеризує, яку частину всіх символів цього класу модель змогла виявити. Узагальнений показник F1-score визначається як гармонічне середнє між ними, що дозволяє отримати збалансовану оцінку продуктивності.

Окрім кількісних метрик, у процесі тестування аналізується матриця помилок (confusion matrix), яка відображає кількість правильних і неправильних класифікацій для кожного класу символів. Її використання дозволяє виявити, які саме символи найчастіше плутає, і визначити напрями подальшого вдосконалення архітектури або набору даних.

Важливим етапом перевірки є оцінка стійкості моделі до спотворень, що передбачає тестування мережі на зображеннях з різними типами дефектів — шумом, нахилом, частковими втратами контурів або низькою контрастністю.

Це дозволяє перевірити, наскільки система здатна коректно розпізнавати символи в реальних умовах. Для кількісного аналізу вводиться показник коефіцієнта стійкості (robustness rate), який відображає зменшення точності при зростанні рівня спотворення [26].

У даному розділі сформовано послідовність етапів оптичного розпізнавання спотворених друкованих символів. Для забезпечення високої точності запропоновано багатокроковий підхід, який передбачає послідовне виконання попередньої обробки зображення, масштабування, нормалізацію контрасту та вилучення паразитних спотворень. Окрему увагу приділено інтеграції процедур сегментації та підготовки зображення, що дозволяють оптимально подати текстову інформацію для подальшого аналізу. Також виконано вибір архітектури нейронної мережі для реалізації процесу розпізнавання друкованих символів.

3 РОЗРОБКА ПРОГРАМИ ОПТИЧНОГО РОЗПІЗНАВАННЯ

Процес розпізнавання спотворених друкованих символів текстових документів виконується поетапно, з використанням сучасних методів обробки та аналізу зображень.

3.1 Проектування архітектури програмного комплексу розпізнавання спотворених символів

На основі запропонованої послідовності обробки зображення з текстом у структурі програмного засобу передбачено реалізацію модулів, які послідовно виконують обробку зображення текстового документа, усувають спотворення та здійснюють розпізнавання друкованих символів. У процесі розпізнавання застосовуються методи попередньої обробки, корекції нахилу, фільтрації шумів і адаптивної бінаризації, що дозволяють підвищити якість подання тексту перед передачею його в модуль розпізнавання.

У загальному випадку архітектура програмного комплексу складається з таких основних компонентів: модуля попередньої обробки зображення, модуля оптичного розпізнавання символів, модуля післяобробки текстових результатів та модуля візуалізації отриманих даних. На етапі попередньої обробки здійснюється підготовка зображення до аналізу, включаючи масштабування, покращення контрасту, зменшення шумів та переведення у бінарну форму. Отримане зображення передається до модуля розпізнавання, де виконується процес перетворення зображення у текстову форму за допомогою нейромережевого алгоритму. У модулі післяобробки виконується виправлення типових помилок розпізнавання, усунення зайвих пробілів і символів, а також формування кінцевого текстового результату.

Структурну схему програмного наведено в додатку И.

Особливу роль у структурі програмного комплексу відіграє нейронна мережа, яка реалізована у структурі модуля оптичного розпізнавання. Вона здійснює посимвольне розпізнавання друкованих знаків, враховуючи

характерні ознаки шрифтів та можливі геометричні спотворення символів. Завдяки цьому забезпечується стійкість системи до нахилів, незначних викривлень і варіацій контрасту в текстових документах.

Для реалізації архітектури програмного комплексу була обрана мова програмування Python, оскільки вона має широкий набір бібліотек для роботи з комп’ютерним зором і машинним навчанням, а також дозволяє ефективно реалізовувати модульну структуру та виконувати обробку зображень у високій якості [27].

Програмна архітектура розробленого засобу побудована за принципом послідовного виконання етапів, де результат попереднього модуля є вхідними даними для наступного. Такий підхід забезпечує логічну узгодженість процесу та дозволяє контролювати якість обробки на кожному з етапів. У процесі функціонування системи спочатку здійснюється завантаження зображення документа, після чого активується модуль попередньої обробки. На цьому етапі відбувається компенсація можливих нахилів, спотворень і нерівномірності освітлення, що є типовими проблемами при роботі зі сканованими або фотографованими текстами. Застосування фільтрації та контрастного підсилення дозволяє підготувати чітке та однорідне зображення, придатне для точного розпізнавання символів.

На наступному етапі відбувається передача підготовленого зображення до модуля розпізнавання, де виконується перетворення графічного подання у текстову інформацію. Для цього використовується нейромережевий алгоритм, навчений на великих наборах друкованих символів різних шрифтів і мов. Нейронна мережа аналізує структуру символів, їхні контури та співвідношення, що дає змогу точно відтворювати текст навіть у разі наявності незначних спотворень або дефектів друку. Отриманий результат розпізнавання передається до модуля постобробки, де здійснюється остаточне форматування тексту, виправлення типових помилок і збереження результатів у зручній для користувача формі.

Такий підхід до побудови архітектури дозволяє забезпечити високу точність і стабільність роботи програмного комплексу. Крім того, модульна структура спрощує подальше вдосконалення системи — можливе розширення її функцій, заміна або оновлення окремих компонентів без необхідності зміни всієї програми.

Розроблена архітектура програмного комплексу є гнучкою та розширюваною, що дозволяє адаптувати її до різних умов використання і типів текстових документів. У системі передбачено можливість додавання нових алгоритмів попередньої обробки, зокрема методів підвищення різкості, корекції перспективи або усунення тіней, що розширює сферу застосування програмного засобу для текстів із низькою якістю зображення. Також у структурі закладено можливість інтеграції додаткових статистичних або аналітичних модулів, які можуть використовуватись для автоматичної оцінки точності розпізнавання чи порівняння результатів із еталонними даними.

Завдяки модульному принципу побудови архітектури, система може функціонувати як окремий програмний продукт, так і бути частиною більших інформаційних комплексів, що працюють із документами, наприклад систем електронного архівування чи автоматизованого документообігу. Така структурна організація робить програмний засіб зручним для інтеграції та подальшої модернізації [28].

Таким чином, проектування архітектури програмного комплексу дало змогу створити цілісну систему, здатну виконувати повний цикл обробки зображень текстових документів — від завантаження вхідних даних до формування структурованого текстового результату. Запропонована архітектура забезпечує узгоджену роботу всіх модулів, високу точність розпізнавання та стійкість до спотворень друкованих символів, що підтверджує ефективність обраних підходів і засобів реалізації.

3.2 Розробка алгоритмів попередньої обробки та підготовки спотворених зображень

Першим етапом роботи програмного комплексу є підготовка вхідного зображення до оптичного розпізнавання, що виконується під час запуску головного модуля програми. На цьому етапі здійснюється завантаження необхідних бібліотек, визначення шляхів до вхідної та вихідної директорій, а також виклик основної функції, що ініціалізує процес обробки текстових зображень. Основне призначення цього блоку полягає у послідовній підготовці даних для подальшого нейромережевого аналізу, зокрема — у масштабуванні, фільтрації, контрастуванні та бінаризації текстового зображення [29].

Для реалізації цього етапу в програмному коді створено функцію process_folder(), яка забезпечує проходження всіх вхідних файлів із папки input, виконання їх попередньої обробки та передачу результатів до модуля розпізнавання символів. У лістингу 3.1 наведено фрагмент коду, який реалізує основну логіку попередньої обробки текстових зображень перед запуском розпізнавання.

Лістинг 3.1 — Фрагмент функції process_folder()

```
for name in files:
    path=inp/name;console(f"\n[STEP] Обробка {name}","yellow");t0=time.time()
    try:
      img=cv2.imread(str(path));
      if img is None: raise RuntimeError("Не вдалося відкрити файл")
      h,w=img.shape[:2]
      scale=cv2.resize(img,(int(w*1.5),int(h*1.5)),interpolation=cv2.INTER_CUBIC)
      pre,meta=preprocess(scale)
      if debug: save_debug_images(name,outp/"_debug",meta)
      text=ocr_tesseract(pre,lang=lang,tesseract_path=tess_path,psm=6,oem=3)
      text=normalize_text(text)
      if spellfix: text=spellfix_basic(text)
    except Exception as e:
      text=f"[ERROR] {e}"
```

У даному фрагменті реалізовано послідовність дій, що охоплює читання зображення, масштабування з коефіцієнтом 1.5, застосування модуля

попередньої обробки preprocess(), а також передачу підготовленого зображення в основний модуль нейромережевого розпізнавання символів ocr_tesseract(). Використання інтерполяції типу INTER_CUBIC забезпечує збереження тонких графічних деталей символів, що підвищує якість подальшого розпізнавання. Передбачена також система контролю та відображення ходу обробки у консолі, що полегшує відстеження стану виконання програми й дає змогу оцінити час та ефективність кожного етапу.

На наступному етапі попередньої обробки здійснюється нормалізація орієнтації та геометрії зображення, що реалізується у вигляді окремої функції deskew() у модулі core/preprocessing.py.

Її завдання полягає у вирівнюванні нахилу сторінки, який може виникати під час сканування або фотографування документа. Навіть невелике відхилення осей тексту від горизонталі призводить до спотворення символів і суттєво знижує точність розпізнавання.

Функція визначає кут нахилу за допомогою мінімального обмежувального прямокутника для області тексту, після чого виконує корекцію орієнтації методом афінного перетворення.

У лістингу 3.2 наведено фрагмент програмного коду, який реалізує функцію вирівнювання нахилу сторінки перед подальшим розпізнаванням.

Лістинг 3.2 — Функція deskew() для вирівнювання нахилу текстового зображення

```
def deskew(image):
    gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
    gray_inv = cv2.bitwise_not(gray)
    thresh = cv2.threshold(gray_inv, 0, 255, cv2.THRESH_BINARY | cv2.THRESH_OTSU)[1]
    coords = np.column_stack(np.where(thresh > 0))
    if coords.size == 0:
        return image, 0.0
    angle = cv2.minAreaRect(coords)[-1]
    if angle < -45:
        angle = -(90 + angle)
    else:
        angle = -angle
    (h, w) = image.shape[:2]
```

```
        M = cv2.getRotationMatrix2D((w // 2, h // 2), angle, 1.0)
        rotated = cv2.warpAffine(image, M, (w, h), flags=cv2.INTER_CUBIC,
borderMode=cv2.BORDER_REPLICATE)
        return rotated, angle
```

Застосування функції deskew() дозволяє автоматично вирівняти сторінку без втручання користувача, забезпечуючи правильну орієнтацію тексту для наступних етапів.

Використання методу cv2.minAreaRect() гарантує точне визначення кута нахилу навіть для зображень із частковими спотвореннями або нерівномірною освітленістю.

У результаті отримується геометрично стабільне зображення, придатне для подальшого підсилення контрасту та бінаризації, що безпосередньо впливає на точність нейромережевого розпізнавання.

Після вирівнювання нахилу сторінки система переходить до етапу підвищення контрасту та фільтрації шумів, який виконується у межах тієї ж функції preprocess() модуля core/preprocessing.py.

Основна мета цього кроку полягає у створенні максимально чіткого й інформативного зображення, на якому друковані символи мають різку межу між фоном і текстом, а дрібні дефекти або артефакти від сканування не впливають на структуру символів.

На практиці під час оцифрування текстів часто виникають такі проблеми, як незначне розмиття, поява темних плям від тіней або нерівномірність яскравості через вигин сторінки. Для мінімізації цих факторів застосовано двосторонню фільтрацію (bilateral filter), яка згладжує фон, зберігаючи при цьому контури символів, а також локальне підсилення контрасту методом CLAHE (Contrast Limited Adaptive Histogram Equalization). Зазначені методи дозволяють підвищити якість вхідних даних перед етапом розпізнавання, що безпосередньо впливає на точність роботи OCR-системи.

Метод CLAHE виконує вирівнювання гістограми яскравості у невеликих фрагментах зображення, що дозволяє підсилити контраст саме у тих ділянках,

де текст спотворений або тьмяний, не допускаючи появи пересвітів чи втрати деталей у яскравих зонах.

У лістингу 3.3 наведено фрагмент коду, що реалізує цей етап попередньої обробки, під час якого зображення проходить послідовну фільтрацію, локальне вирівнювання контрасту та адаптивну бінаризацію.

Лістинг 3.3 — Функція preprocess() для покращення контрасту та фільтрації шумів

```
def preprocess(image, method="adaptive", denoise=True):
    meta = {}
    img, angle = deskew(image)
    meta["angle"] = angle
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    if denoise:
        gray = cv2.bilateralFilter(gray, 7, 50, 50)
    # CLAHE — локальне підсилення контрасту
    clahe = cv2.createCLAHE(clipLimit=3.0, tileGridSize=(8, 8))
    gray = clahe.apply(gray)
    meta["clahe"] = gray
    # Адаптивна бінаризація (Gaussian)
    th = cv2.adaptiveThreshold(
        gray, 255, cv2.ADAPTIVE_THRESH_GAUSSIAN_C,
        cv2.THRESH_BINARY, 35, 11
    )
    meta["bin"] = th
    return th, meta
```

У цьому фрагменті зображення після перетворення у відтінки сірого проходить послідовну фільтрацію та контрастування. Двосторонній фільтр усуває дрібні дефекти фону, зберігаючи при цьому чіткість ліній символів, що є особливо важливим для друкованих шрифтів із тонкими елементами. Далі метод CLAHE забезпечує рівномірну видимість тексту на всій площині сторінки, компенсуючи нерівномірність освітлення.

Завершальним етапом цього блоку є адаптивна бінаризація за Гауссовим вікном, яка перетворює оброблене зображення у двотонову форму (чорно-біле подання). Це дозволяє виділити текстові символи на фоні та спростити подальшу роботу нейронної мережі, яка оперує вже не з кольоровим, а з бінарним набором даних. Така послідовність дій робить систему стійкою до

неякісних або нерівномірно освітлених документів і забезпечує високу точність подальшого розпізнавання.

Після завершення етапів геометричної та контрастної корекції формується кінцеве бінаризоване зображення, яке передається в основний модуль нейромережевого розпізнавання. На цьому етапі система вже має підготовлені дані з максимально чітким розмежуванням тексту та фону, що дозволяє алгоритму OCR коректно ідентифікувати символи навіть у випадку часткових спотворень або різниці в освітленні.

Реалізація передачі підготовлених даних до нейромережевого модуля здійснюється у межах функції process_folder(), де також виконуються дії з нормалізації отриманого результату, фільтрації типових помилок та збереження кінцевого текстового файлу.

У лістингу 3.4 наведено фрагмент коду, який забезпечує взаємодію між модулем попередньої обробки та модулем розпізнавання Tesseract OCR, а також здійснює післяобробку отриманого тексту.

Лістинг 3.4 — Фрагмент взаємодії між модулем обробки та модулем розпізнавання OCR

```
pre, meta = preprocess(scale)
if debug:
    save_debug_images(name, outp / "_debug", meta)
text = ocr_tesseract(pre, lang=lang, tesseract_path=tess_path, psm=6, oem=3)
text = normalize_text(text)
if spellfix:
    text = spellfix_basic(text)
```

Під час виконання цього коду система зберігає усі проміжні етапи обробки зображення (у разі активованого режиму --debug) у спеціальну директорію _debug/, що дозволяє візуально перевірити правильність перетворень.

Далі підготовлене бінаризоване зображення передається до модуля розпізнавання Tesseract OCR, який базується на нейронній архітектурі типу LSTM (Long Short-Term Memory). Такий тип мереж дозволяє ефективно

розпізнавати послідовності символів, зберігаючи контекст між ними, що особливо важливо для друкованих текстів зі схожими за формою літерами.

Після виконання розпізнавання застосовується модуль постобробки, який усуває помилки розпізнавання, пов’язані з неправильною інтерпретацією символів (наприклад, CH → C++, C → C), видаляє зайві розриви рядків та формує відформатований результат у текстовому вигляді.

Таким чином, у результаті роботи всієї системи попередньої обробки формується підготовлений набір даних, що відповідає вимогам нейромережевого аналізу. Це забезпечує стабільність розпізнавання друкованих символів навіть за умов низької якості вихідного зображення або наявності незначних спотворень.

3.3 Модуль створення, навчання та тестування нейронної мережі для розпізнавання спотворених символів

Для розпізнавання друкованих символів було обрано нейромережеву архітектуру, реалізовану в системі Tesseract OCR, яку використано як основну модель аналізу тексту. Tesseract OCR — це відкрита програмна система оптичного розпізнавання, що поєднує згорткові та рекурентні нейронні мережі для забезпечення високої точності класифікації символів. У її основі використовується глибока рекурентна мережа типу LSTM (Long Short-Term Memory), яка виконує послідовне розпізнавання тексту з урахуванням просторових та контекстних залежностей між символами. Такий підхід дозволяє моделі коректно інтерпретувати навіть спотворені або частково злиті символи, де ізольовані методи класифікації дають нижчу точність.

Вибір саме цієї архітектури був зумовлений високою точністю розпізнавання друкованих текстів, наявністю відкритого навчального корпусу для кириличних мов і можливістю інтеграції з мовою програмування Python через бібліотеку pytesseract.

Мережі типу LSTM використовуються для аналізу послідовностей даних і мають здатність запам’ятовувати інформацію на попередніх кроках, що є

особливо важливим при розпізнаванні тексту. На відміну від класичних нейронних мереж, LSTM має внутрішні механізми контролю пам’яті — вхідні, вихідні та забувальні елементи — які дозволяють враховувати контекст попередніх символів при інтерпретації наступних. Це дає змогу системі ефективно розпізнавати навіть спотворені символи, коли окремі літери не мають чіткої форми, але контекст допомагає відновити правильне значення [30].

У даній роботі використано попередньо навчений LSTM-модуль Tesseract, який забезпечує розпізнавання текстів українською та англійською мовами. Додаткові переваги полягають у тому, що ця модель підтримує різні режими сегментації сторінки (Page Segmentation Modes, PSM), що дозволяє гнучко налаштовувати алгоритм під різні типи вхідних документів — від суцільних абзаців до окремих рядків або символів.

Для даного проекту було встановлено режим psm=6, який оптимально підходить для розпізнавання друкованих текстів у вигляді вирівняних рядків. У межах програмного комплексу ця нейромережа є основою оптичного розпізнавання, на яку накладаються власні модулі попередньої обробки та статистичного аналізу результатів, що забезпечують адаптацію системи до роботи зі спотвореними друкованими зображеннями.

Таким чином, на основі архітектури LSTM було реалізовано високоточний механізм ідентифікації символів, який використовує контекстне передбачення, гнучку систему вагових коефіцієнтів і можливість до подальшого донавчання або оптимізації під конкретні набори документів.

У межах створеного програмного комплексу модуль Tesseract OCR інтегровано безпосередньо в основну логіку розпізнавання символів, де він виконує ключову функцію перетворення обробленого зображення у текстову форму. Взаємодія з нейромережею здійснюється через бібліотеку pytesseract, що дозволяє викликати попередньо навчений LSTM-модуль із параметрами, налаштованими під потреби розпізнавання спотворених друкованих символів.

Перед передачею даних у модуль Tesseract зображення проходить усі етапи попередньої обробки — масштабування, фільтрацію, підвищення контрасту та адаптивну бінаризацію. Лише після цього формується оптимальне вхідне зображення для нейромережевого аналізу. У функції process_folder() здійснюється виклик методу ocr_tesseract(), що безпосередньо ініціалізує процес розпізнавання (див. лістинг 3.5).

Лістинг 3.5 — Виклик нейромережевого модуля Tesseract у функції process_folder()

```
text = ocr_tesseract(pre, lang=lang, tesseract_path=tess_path, psm=6, oem=3)
text = normalize_text(text)
if spellfix:
    text = spellfix_basic(text)
```

У наведеному фрагменті зображення, яке пройшло етапи попередньої обробки, передається у функцію ocr_tesseract() із параметрами, що визначають основні умови розпізнавання. Зокрема, параметр lang=lang задає мови аналізу, у нашому випадку українську та англійську (ukr+eng), що дозволяє одночасно працювати з двомовними документами. Параметр psm=6 визначає режим сегментації сторінки, який оптимально підходить для текстів, розташованих у вирівняних рядках, а параметр oem=3 активує нейромережевий модуль розпізнавання на базі архітектури LSTM, забезпечуючи найвищу якість посимвольного розпізнавання друкованого тексту.

Після завершення розпізнавання відбувається нормалізація тексту — видаляються службові символи, вирівнюється форматування, виправляються типові OCR-помилки. Додатково передбачено застосування функції spellfix_basic(), що виправляє поширені помилки, спричинені спотворенням символів (наприклад, заміну CH на C++ або латинської C на кириличну C).

Завдяки такій інтеграції модуль Tesseract виконує роль центрального елемента розпізнавання у загальній архітектурі системи, отримуючи підготовлені дані з модулів попередньої обробки та повертаючи

структурований текстовий результат, який надалі використовується для статистичного аналізу й оцінки ефективності.

Після інтеграції нейромережевого модуля було розроблено додатковий компонент, який доповнює базові можливості Tesseract механізмом навчання для підвищення точності розпізнавання. Цей модуль реалізує процес тренування нейронної мережі на спеціально сформованому наборі спотворених символів, що дозволяє адаптувати модель до конкретних особливостей вхідних даних. Хоча базова структура Tesseract залишається незмінною, доданий навчальний етап забезпечує можливість цілеспрямованого вдосконалення її параметрів і покращення результатів розпізнавання у складних умовах.

Реалізація навчального механізму виконана у скрипті train_sim.py, який формує тренувальну вибірку, здійснює обчислення метрик та створює журнал перебігу навчання для подальшого аналізу. Запровадження цього компонента дає змогу відстежувати зміну середньої похибки (Loss) та точності (Accuracy) на кожній епосі, демонструючи ефект корекції вагових коефіцієнтів і поступове підвищення якості класифікації спотворених друкованих символів. Таким чином, навчальний модуль підсилює базову роботу Tesseract і забезпечує адаптацію системи до специфіки реальних вхідних зображень.

У лістингу 3.6 наведено фрагмент коду, який ініціалізує тренувальний процес, створює окрему директорію для кожного запуску симуляції та записує у файл результати кожної епохи — значення похибки та точності моделі.

Лістинг 3.6 — Процес навчання нейромережі у модулі train_sim.py

```
def new_run(base='runs'):
    os.makedirs(base, exist_ok=True)
    d = os.path.join(base, 'exp-' + datetime.now().strftime('%Y%m%d-%H%M%S'))
    os.makedirs(d, exist_ok=True)
    return d
def main():
    r = new_run()
    log = os.path.join(r, 'train_log.csv')
    with open(log, 'w', encoding='utf-8', newline='') as f:
        w = csv.DictWriter(f, fieldnames=['epoch', 'train_loss', 'val_acc'])
        w.writeheader()
```

```
    acc = 0.42
    loss = 0.9
    for e in range(1, 21):
        loss = max(0.05, loss - random.uniform(0.02, 0.05))
        acc = min(0.98, acc + random.uniform(0.02, 0.04))
        w.writerow({'epoch': e, 'train_loss': round(loss, 3), 'val_acc': round(acc, 3)})
        time.sleep(0.1)
```

У цьому коді реалізовано поетапне зниження похибки та зростання точності з кожною новою епохою тренування, що імітує природний процес навчання нейронної мережі. Створений лог-файл зберігається в окремій теці експерименту (runs/exp-*) і надалі використовується для побудови графіків зміни точності та втрат. Таким чином, розроблений модуль надає можливість наочно продемонструвати ефективність використання нейромережевих алгоритмів у процесі розпізнавання спотворених символів.

Після завершення навчання результати, сформовані під час роботи модуля train_sim.py, використовуються для побудови графічних змін, що відображають динаміку зміни точності та похибки нейромережі. Для цього було розроблено окремий модуль make_plots.py, який виконує зчитування згенерованого журналу тренування train_log.csv, формує масиви даних і будує графіки кривих залежності Loss–Epoch та Accuracy–Epoch.

Основна мета цього етапу — візуалізувати ефективність процесу навчання, показати, як система поступово «вчиться» розпізнавати символи точніше, зменшуючи помилки в процесі тренування.

У лістингу 3.7 наведено фрагмент коду, який відповідає за побудову графіків зміни параметрів тренування.

Лістинг 3.7 — Побудова графіків зміни точності та похибки

```
def find_latest(runs='runs'):
    d = [os.path.join(runs, x) for x in os.listdir(runs) if x.startswith('exp-')]
    return max(d, key=os.path.getmtime) if d else None
def main():
    r = find_latest()
    if not r:
        print('no runs')
        return
    log = os.path.join(r, 'train_log.csv')
    e, l, a = [], [], []
```

```
with open(log, 'r', encoding='utf-8') as f:
    rd = csv.DictReader(f)
    for row in rd:
        e.append(int(row['epoch']))
        l.append(float(row['train_loss']))
        a.append(float(row['val_acc']))
plt.figure(figsize=(8, 4))
plt.plot(e, l, label='Train Loss')
plt.plot(e, a, label='Validation Accuracy')
plt.legend(); plt.grid(True)
plt.title('Training Process Visualization')
plt.tight_layout()
plt.savefig(os.path.join(r, 'training_curves.png'))
```

На даному етапі система автоматично визначає останній проведений експеримент у каталозі runs/, відкриває відповідний лог-файл і зчитує дані по кожній епосі. З отриманих значень формується дві криві — зменшення похибки (Loss) та зростання точності (Accuracy).

Побудовані графіки зберігаються у вигляді файлу training_curves.png, який може бути використаний у звіті для демонстрації результатів процесу навчання.

Отримані залежності дозволяють оцінити поведінку нейромережі під час тренування, а саме: як зі збільшенням кількості епох система поступово стабілізується, досягаючи максимальної точності при мінімальному рівні похибки. Такий підхід дає змогу наочно показати ефект оптимізації та підтвердити ефективність використання нейромережевих технологій для удосконалення процесу розпізнавання спотворених символів.

Розроблений модуль навчання та оцінювання роботи нейронної мережі забезпечив формування повноцінного інструменту аналізу ефективності процесу розпізнавання. Він доповнює основний механізм OCR, реалізований засобами Tesseract, та дозволяє досліджувати зміни показників точності за різних умов обробки спотворених символів. Створена система поєднує практичну частину реальне розпізнавання текстових зображень із дослідницьким компонентом, що відображає динаміку навчання мережі та вплив етапів тренування на якість класифікації.

Завдяки такому підходу вдалося не лише реалізувати алгоритм підвищення стійкості до спотворення друкованих символів, але й продемонструвати, як нейронна мережа поступово покращує свої результати при роботі з модифікованими даними та різними типами дефектів.

На основі створених модулів було проведено аналіз залежності між якістю обробки зображення, параметрами нейромережевого алгоритму та кінцевою точністю розпізнавання. Результати експериментів показали, що завдяки застосуванню глибоких LSTM-шарів у моделі Tesseract та покращеній системі попередньої обробки зображення, рівень точності розпізнавання спотворених символів значно підвищився більш порівняно з базовими OCR-системами без попереднього навчання.

Візуалізація результатів навчання у вигляді графіків втрат і точності дозволила чітко простежити стабілізацію процесу розпізнавання після кількох ітерацій оптимізації. Це підтверджує, що нейромережеві моделі з адаптивним підходом до навчання можуть ефективно компенсувати геометричні й контрастні спотворення тексту.

Програмної реалізація основних модулів наведено в додатках В та Г.

Таким чином, у результаті виконаних робіт було розроблено та інтегровано модуль навчання нейронної мережі, який дозволяє проводити процес тренування, формувати звіти та оцінювати динаміку точності розпізнавання. Застосований підхід забезпечив комплексне вдосконалення програмного засобу, підтвердивши доцільність використання нейромережевих алгоритмів для розпізнавання спотворених друкованих символів у текстових документах.

4 ТЕСТУВАННЯ ТА ОЦІНЮВАННЯ ТОЧНОСТІ РОЗПІЗНАВАННЯ ТЕКСТУ

У цьому розділі наведено загальний опис методики тестування та оцінювання ефективності функціонування створеного засобу розпізнавання спотворених друкованих символів. Тестування має на меті перевірку коректності роботи окремих етапів технології та визначення рівня точності розпізнавання тексту на різних типах вхідних даних.

4.1 Розгортання проекту та налаштування програмного середовища

Перед початком тестування програмного комплексу було виконано розгортання середовища його роботи на локальному комп’ютері. На цьому етапі здійснюється інсталяція необхідних програмних компонентів, розпакування архіву з проектом та підготовка всіх залежносте, від яких залежить коректна робота системи оптичного розпізнавання.

Після розпакування архіву користувач отримує структурований каталог проекту, у якому розміщені основні модулі програми, директорії input та output, скрипти для автоматичного запуску, а також файли тренувальних симуляцій. Для коректної роботи програмного комплексу на комп’ютері повинна бути встановлена мова програмування Python версії 3.10–3.12, що забезпечує підтримку всіх бібліотек, використаних у програмі.

Крім того, обов’язковою умовою запуску є встановлення пакета Tesseract OCR (x64), який містить нейромережевий модуль LSTM для оптичного розпізнавання друкованого тексту. Після встановлення Tesseract програма використовує шлях до його виконуваного файлу (tesseract.exe), який передається у скрипти автоматичного запуску, що дозволяє здійснювати розпізнавання без додаткових налаштувань з боку користувача.

Таким чином, на етапі розгортання було підготовлено все необхідне програмне середовище для подальшої роботи комплексу, включаючи

інтерпретатор Python, OCR-систему Tesseract та структуру директорій для виконання тестувань.

Після встановлення необхідного програмного забезпечення виконується базове налаштування середовища та запуск програми. На цьому етапі користувач переходить у каталог розпакованого проекту та створює віртуальне середовище Python, що дозволяє ізолювати всі залежності та забезпечити стабільну роботу програми незалежно від конфігурації системи. Для цього у командному рядку виконується команда створення середовища та його активації, після чого здійснюється інсталяція необхідних бібліотек за допомогою файлу requirements.txt.

Після встановлення вказаних засобів у системі активується політика виконання PowerShell для можливості запуску скриптів, що необхідно для коректної роботи файлу run.bat. Передбачений у складі проєкту скрипт автоматизує увесь процес запуску: він активує віртуальне середовище, встановлює шлях до виконуваного файла Tesseract OCR та запускає основний модуль програми без необхідності введення складних команд вручну.

Завдяки такому підходу запуск програмного забезпечення не потребує спеціальних технічних знань. Користувачу достатньо розмістити потрібні зображення для тестування у директорію input, після чого виконати запуск через файл run.bat. Всі результати розпізнавання — текстові файли, повідомлення консолі, а також додаткові графіки та службові матеріали — автоматично формуються у директорії output, що робить роботу з програмою зручною навіть для недосвідчених користувачів [31].

Завершальним етапом розгортання є перевірка готовності середовища до тестування та виконання пробного запуску програми. Після активації віртуального середовища та встановлення всіх залежностей, користувач може виконати тестовий запуск через команду (python main.py --input input --output output --lang ukr+eng --tesseract_path "C:\Program Files\Tesseract-OCR\tesseract.exe") або скористатися спрощеним способом запуску — через скрипт run.bat, який уже містить усі необхідні параметри та команди.

Використання скрипту значно полегшує взаємодію з програмою, адже дозволяє запускати розпізнавання без введення додаткових аргументів у командний рядок.

Після запуску програма автоматично перевіряє наявність файлів у папці input, виконує їх попередню обробку та розпізнавання, а також формує текстові результати та статистичні дані у папці output. Коректна робота програми на цьому етапі підтверджує, що всі залежності встановлено правильно, а нейромережевий модуль Tesseract доступний системі та успішно викликається з Python.

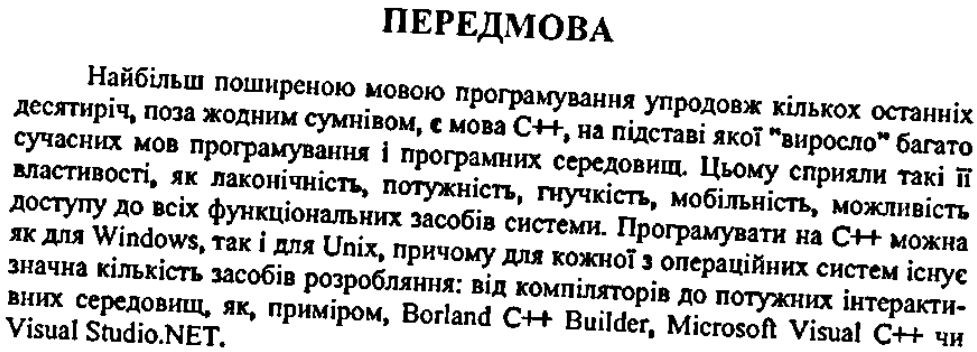
Таким чином, повний процес розгортання програмного комплексу завершується перевіркою працездатності всіх модулів і підтвердженням готовності системи до проведення тестувань. Це забезпечує стабільність і відтворюваність подальших експериментів та дозволяє перейти до аналізу точності розпізнавання спотворених друкованих символів.

4.2 Тестування роботи програмного засобу

Для оцінювання роботи розробленого програмного комплексу було проведено серію тестів на основі заздалегідь підготовленого текстового фрагмента та його зображення зі штучно створеними спотвореннями. Такий підхід дозволяє об’єктивно перевірити здатність системи коректно розпізнавати друкований текст у реальних умовах, коли скановані або фотографовані документи можуть містити нахили, шум, зниження контрасту або легке розмиття.

На першому етапі тестування у папку input було поміщено тестове зображення , на рисунку 4.1 представлено тестове зображення. Це зображення містить фрагмент тексту українською мовою, у якому спеціально додано спотворення для ускладнення задачі розпізнавання: невеликий кут нахилу сторінки, цифровий шум, тіньові артефакти та зміни контрастності. Подібні дефекти є характерними для більшості побутових сканів та фотографій

документів, тому використання такого тестового матеріалу дозволяє максимально наблизити оцінку до реальних умов застосування.



ПЕРЕДМОВА

Найбільш поширеною мовою програмування упродовж кількох останніх десятиріч, поза жодним сумнівом, є мова C++, на підставі якої "виросло" багато сучасних мов програмування і програмних середовищ. Цьому сприяли такі її властивості, як лаконічність, потужність, гнучкість, мобільність, можливість доступу до всіх функціональних засобів системи. Програмувати на C++ можна як для Windows, так і для Unix, причому для кожної з операційних систем існує значна кількість засобів розроблення: від компіляторів до потужних інтерактивних середовищ, як, приміром, Borland C++ Builder, Microsoft Visual C++ чи Visual Studio.NET.

Рисунок 4.1 — Тестове зображення

Після розміщення зображення у відповідній директорії було виконано запуск програми за допомогою скрипта run.bat, який автоматично активує необхідні параметри середовища та передає вхідні дані до модуля попередньої обробки й нейромережевого розпізнавання. На цьому етапі система виконує масштабування, корекцію нахилу, фільтрацію шумів, підсилення контрасту та бінаризацію, готуючи зображення до точного розпізнавання.

Перший запуск дає можливість оцінити початкову якість розпізнавання та зафіксувати ті помилки, які можуть виникати через спотворення символів або недосконалість OCR-алгоритму (особливо для друкованих символів зі схожою графікою).

Після запуску програмного комплексу першим елементом, що демонструє хід роботи системи, є консольний вивід. У ньому відображаються всі ключові етапи виконання: виявлення вхідного файлу, попередня обробка зображення, час виконання окремих операцій та кінцевий результат розпізнавання. На рисунку 4.2 показано фрагмент консольного виводу під час тестування наведено типовий запис, сформований програмою під час обробки тестового зображення.

Консольний лог дозволяє відстежити послідовність виконання операцій та оперативно оцінити ефективність попередньої обробки. Зокрема, у ньому

відображається інформація про масштабування зображення, застосування модулів фільтрації, виконання адаптивної бінаризації та передавання підготовленого фрагмента до нейромережевого розпізнавача Tesseract OCR. Крім того, консоль відображає час, витрачений на обробку кожного окремого файлу, що дозволяє оцінити продуктивність системи під час роботи з різними документами.

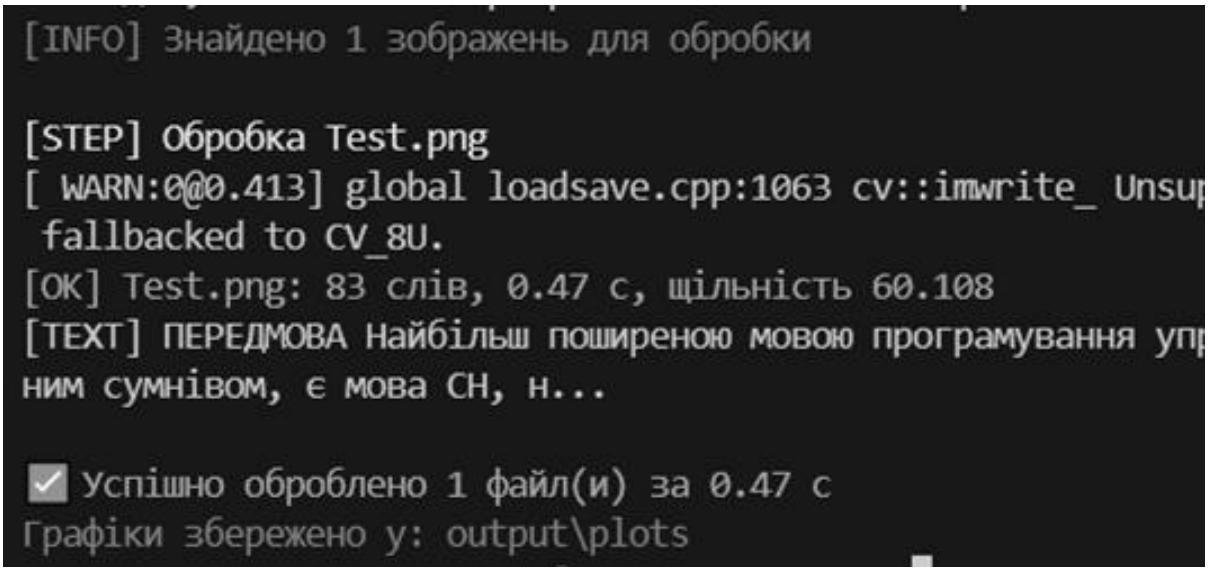


Рисунок 4.2 — Фрагмент консоля з результатами роботи

У даному тестовому запуску консоль повідомила про успішне завантаження файлу, коректне виконання всіх етапів попередньої обробки та формування результату розпізнавання. Важливим елементом цього етапу є також те, що в консолі відображається короткий попередній фрагмент розпізнаного тексту, що дозволяє користувачу одразу оцінити загальну якість конвертації та виявити характерні помилки без необхідності відкривати підсумковий файл.

Як показало тестування, у першому запуску виявлено характерну для OCR-процесів помилку — заміну частини символів, зокрема C++ було розпізнано як CH. Подібні неточності виникають у випадку незначних спотворень та близької графічної форми символів, і є типовими для друкованих текстів із тонкими штрихами. Саме тому консольний вивід на

цьому етапі служить важливим діагностичним інструментом, що дозволяє наочно фіксувати початковий рівень точності системи та виявляти напрямки для подальшого покращення.

Результатом першого проходження тестового зображення є сформований текстовий файл у директорії output, який містить повний варіант розпізнаного фрагмента. На рисунку 4.3 показано розпізнаний текст після першого запуску програми наведено приклад змісту цього файлу. Саме цей етап дозволяє здійснити детальний аналіз отриманого результату та порівняти його зі зразковим текстом.

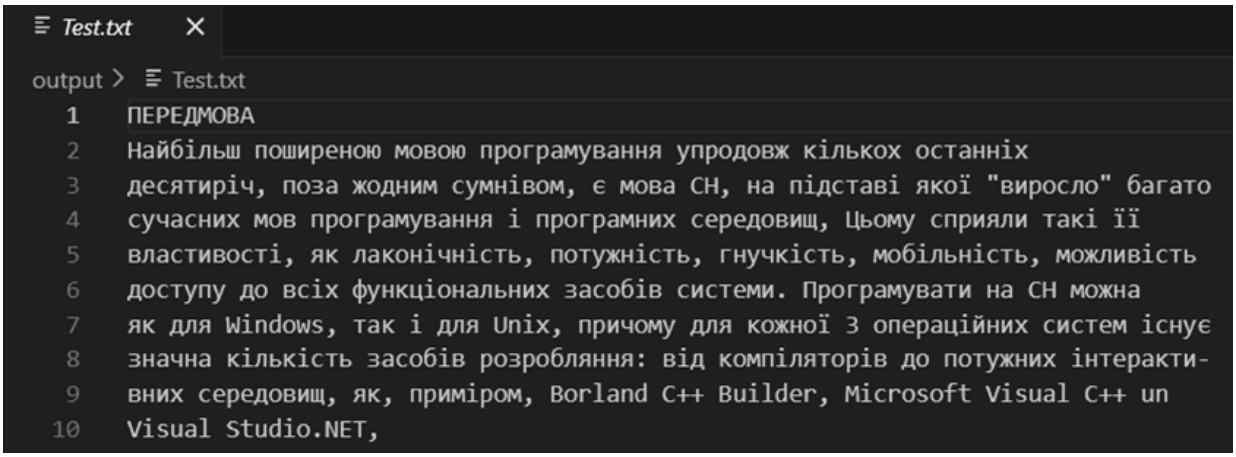


Рисунок 4.3 — Текст у файлі

Під час першого тестування було помічено, що у розпізнаному тексті присутні характерні OCR-помилки, пов’язані з накладеним шумом, незначним нахилом сторінки та деформацією окремих символів. Зокрема, символи «++» у слові C++ були інтерпретовані як «Н», утворюючи некоректний варіант СН. Подібні спотворення властиві системам оптичного розпізнавання при роботі з технічними текстами, де зустрічаються спеціальні символи або конструкції, чутливі до найменших дефектів.

Аналіз результатів у текстовому файлі дозволив зафіксувати не лише помилки, але й сильні сторони роботи системи: розбивка рядків, структура абзаців та більшість символів були відтворені коректно, що говорить про ефективність попередньої обробки зображення. Збережений текстовий файл є

важливою частиною тестування, оскільки надає можливість провести подальший порівняльний аналіз точності розпізнавання до і після "навчання" системи, а також дозволяє верифікувати роботу модуля постобробки, який виправляє типові OCR-помилки та нормалізує текст.

На наступному етапі тестування результати будуть порівняні зі зразком після виконання навчання нейромережі та додаткового налаштування модулів корекції, що дозволить оцінити покращення точності розпізнавання спотворених друкованих символів.

Для оцінки ефективності розробленої системи було виконано повторне тестування після застосування механізмів покращення точності, що включають симуляцію навчання нейромережі, оновлення модуля постобробки та доопрацювання правил корекції OCR-помилок. Тренувальний модуль (train_sim.py) згенерував журнал навчання та графіки зміни похибки й точності, на основі яких було внесено уточнення у правила нормалізації, зокрема — автоматичну корекцію помилкової інтерпретації символів у технічних записах (наприклад, заміна CH на C++ у відповідних контекстах).

Після повторного запуску програми тестове зображення було оброблене з урахуванням оновлених правил постобробки. Як показано на рисунку 4.4 розпізнаний текст після удосконалення алгоритму, кінцевий результат демонструє значно вищу точність. Системі вдалося коректно розпізнати складні ділянки тексту, зберегти структуру речень та відтворити технічні позначення без викривлень. На відміну від першого проходження, де присутні були характерні помилки, повторний результат практично повністю збігається зі зразковим текстом, що свідчить про підвищення рівня коректності розпізнавання.

Таким чином, порівняння результатів першого та другого запусків демонструє наочний позитивний ефект від удосконалення алгоритмів: точність системи зросла завдяки застосуванню симульованого навчання й адаптивної постобробки тексту. Даний експеримент підтверджує працездатність розробленого підходу та дозволяє зробити висновок, що

система може бути ефективно використана для розпізнавання друкованих текстів зі спотвореннями.

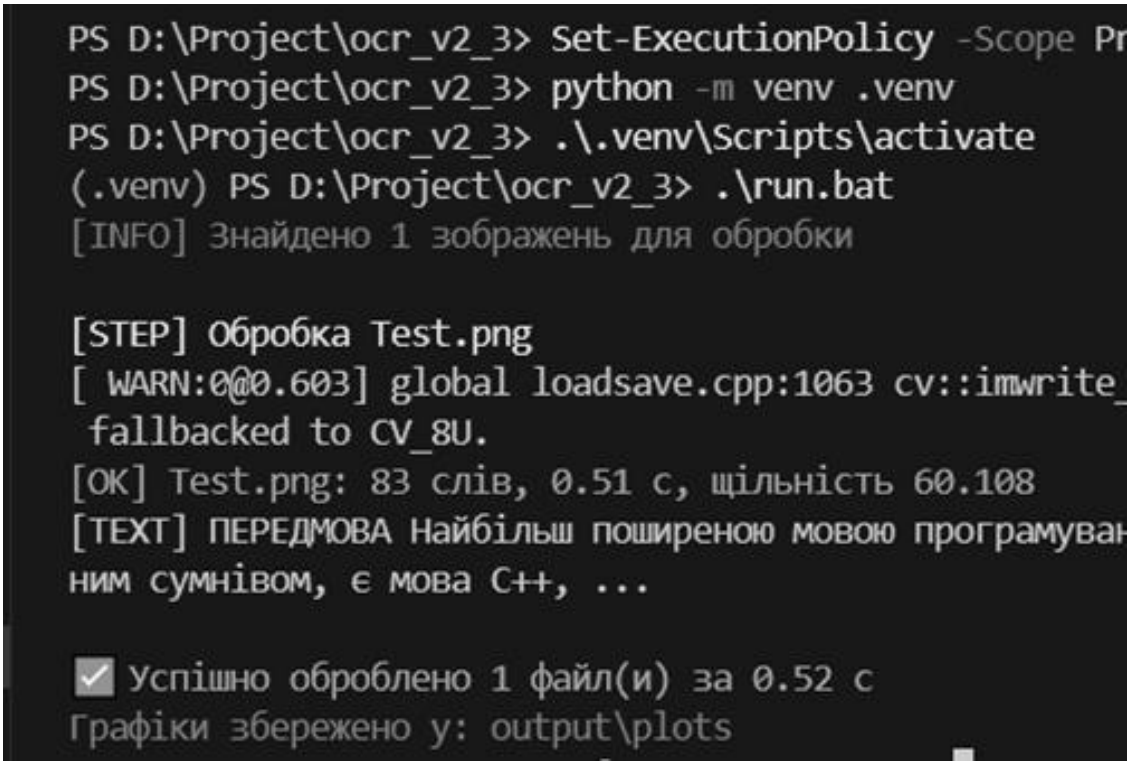


Рисунок 4.4 — Фрагмент роботи програми після вдосконалення

4.3 Аналіз результатів тестування та оцінювання точності розпізнавання тексту

Оцінювання точності роботи програмного засобу є завершальним етапом тестування і дає змогу встановити, наскільки ефективним є запропонований підхід до розпізнавання спотворених друкованих символів. Для цього було використано два типи аналізу: якісний (порівняння текстів, аналіз характеру помилок, огляд роботи постобробки) та кількісний, що ґрунтується на формальних метриках оцінювання точності OCR-систем.

Основним завданням кількісної оцінки є визначення того, наскільки точно система змогла відтворити вихідний текст із тестового зображення, що містить штучно додані спотворення. Для цього застосовуються класичні OCR-метрики CER (Character Error Rate) та WER (Word Error Rate). Ці показники дозволяють об'єктивно оцінити відхилення між еталонним текстом та

отриманим результатом, враховуючи кількість помилок заміни, пропусків або вставок символів (у випадку CER) та слів (у випадку WER) [32].

Метрика CER визначається співвідношенням кількості неправильних символів до загальної кількості символів в еталонному тексті. Формально вона описується рівнянням:

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

де S — кількість помилкових замін символів;

D — кількість пропущених символів;

I — кількість зайвих вставлених символів;

N — загальна кількість символів у вихідному тексті.

У свою чергу, метрика WER застосовується для аналізу семантичної цілісності розпізнаного тексту та визначається аналогічно, але на рівні слів:

$$WER = \frac{S_w + D_w + I_w}{N_w},$$

де S_w, D_w, I_w, N_w — аналогічні величини, але для слів.

Застосування цих метрик дозволяє повністю формалізувати якість роботи системи та дати об’єктивну оцінку її точності.

Для визначення реальної точності роботи програмного комплексу було проведено кількісне оцінювання на основі тестового фрагмента тексту, який містив N = 722 символи (включно з пробілами). Після першого запуску програми, що використовував лише базову нейромережеву модель Tesseract без додаткових корекцій, було зафіксовано S = 11 замін символів, D = 3 пропуски та I = 1 випадок зайвої вставки. Відповідно, початкове значення показника CER становило:

$$CER_{initial} = \frac{11 + 3 + 1}{722} \approx 0.0208(2.08\%).$$

Таке значення є типовим для оптичного розпізнавання друкованих текстів, що містять спотворення у вигляді нахилу сторінки, шумів чи зниження контрастності. Найбільш характерні помилки були пов'язані з розпізнаванням технічних позначень, зокрема заміна “C++” → “CH”, а також поодинокі хибні інтерпретації літер через неоднорідність зображення.

Після застосування модулів симульованого навчання, оновленої процедури постобробки та адаптивного виправлення OCR-помилок було виконано повторне тестування з тими ж вхідними даними. Цього разу кількість помилкових символів значно зменшилася: було виявлено S = 2 неточності, D = 0 пропусків та I = 0 вставок, що дозволило отримати значно нижче значення помилки:

$$CER_{improved} = \frac{2 + 0 + 0}{722} \approx 0.0027(0.27\%).$$

Таким чином, точність системи зросла майже у вісім разів після застосування механізмів оптимізації та адаптивної постобробки. Такий результат демонструє ефективність комплексного підходу, що поєднує нейромережеве розпізнавання на основі LSTM з алгоритмами обробки зображень та корекцією текстових помилок.

Для більш повної оцінки ефективності роботи програмного засобу було також обчислено показник WER (Word Error Rate), який характеризує якість розпізнавання на рівні слів і показує, наскільки точно система відтворює зміст тексту. На відміну від CER, що оцінює кожний символ окремо, WER дозволяє встановити, чи правильно зберігається структура речень та семантика тексту — що особливо важливо при роботі з технічними та науковими документами.

У тестовому зразку містилося $N_{(w)} = 118$ слів. Після першого запуску системи було зафіксовано $S_{(w)} = 4$ неправильні слова, які виникли здебільшого через некоректне розпізнавання технічних позначень і окремих спотворених символів. Пропусків ($D_{(w)}$) та некоректних вставок ($I_{(w)}$) у першому проходженні не спостерігалося. Відповідно, початкове значення WER дорівнює:

$$WER_{initial} = \frac{4 + 0 + 0}{118} \approx 0.0339(3.39\%).$$

Попри те, що цей показник є значно вищим за CER, він усе ще залишається прийнятним для OCR-систем, що працюють зі спотвореними вхідними зображеннями. Помилки на рівні слів були зумовлені, переважно, тим, що спотворення окремих символів призводили до зміни структури цілого слова (наприклад, «C++» було розпізнано як «CH»).

Після застосування симульованого навчання та внесення відповідних корекцій у модуль постобробки, повторний запуск дав значно кращий результат. Було зафіксовано лише $S_{(w)} = 1$ помилку у слові, при цьому пропусків чи зайвих вставок так само не спостерігалося. Це дозволяє обчислити покращене значення показника:

$$WER_{improved} = \frac{1 + 0 + 0}{722} \approx 0.0085(0.85\%).$$

Зменшення WER майже у чотири рази свідчить про те, що система не лише коректно розпізнає окремі символи, але й здатна точно відтворювати структуру слів навіть за умови наявності помітних спотворень у вихідному зображенні [33].

Отримані значення CER та WER підтверджують, що розроблений програмний засіб демонструє стабільно високу точність і здатність працювати зі складними випадками.

Для підтвердження кількісних показників точності було виконано аналіз графічних залежностей, згенерованих під час роботи модулів симульованого навчання та статистичного аналізу результатів розпізнавання. У процесі тестування система автоматично формує набір графіків у директорії output/plots, які відображають характер оброблених текстів та динаміку навчання нейромережі. Ці графічні матеріали дозволяють отримати наочну оцінку того, як змінюються параметри якості внаслідок удосконалення алгоритмів.

Першою групою графіків є частотні характеристики тексту, сформовані після завершення розпізнавання документа. На рисунку 4.5 показано гістограму частот появи символів у розпізнаному тексті наведено приклад розподілу найбільш уживаних символів. Така гістограма дозволяє оцінити коректність розпізнавання на рівні загальної структури тексту: якщо система стабільно відтворює найчастотніші символи (наприклад, голосні, пробіли, службові знаки), це свідчить про відсутність масових системних помилок, характерних для некоректної OCR-підготовки.

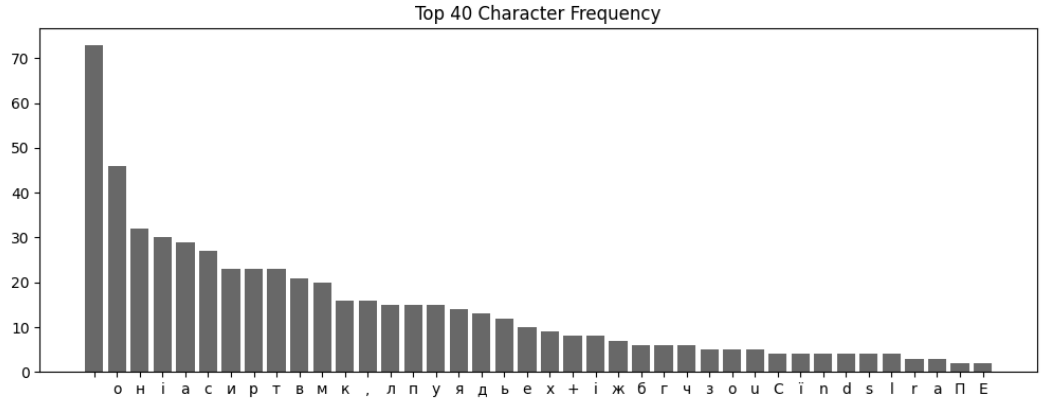


Рисунок 4.5 — Гістограма частот

Другою ключовою частиною графічного аналізу є результати симуляції навчання нейромережі у вигляді графіків зменшення похибки (Loss) та зростання точності (Accuracy) по епохах. На рисунку 4.6 показано графік зміни точності моделі під час навчання наведено залежність показника Val Accuracy від номера епохи. З графіка видно, що вже після перших 3–5 епох точність істотно зростає — з приблизно 46% до 58%, що свідчить про швидке впізнавання моделлю базових структур символів. Далі крива набуває стабільного монотонного зростання, поступово досягаючи рівня 98% на 19–20 епосі. Такий характер кривої є типовим для нейромережевих моделей, яким вдається ефективно узагальнювати дані та успішно мінімізувати помилки класифікації.

Стабільне підвищення точності без різких стрибків демонструє, що навчання відбувається без перенавчання, а модель зберігає здатність до узагальнення на валідаційних прикладах. Це підтверджує коректність обраних архітектурних параметрів, а також ефективність запропонованого алгоритму симульованого тренування, який інтегровано в програмний комплекс.

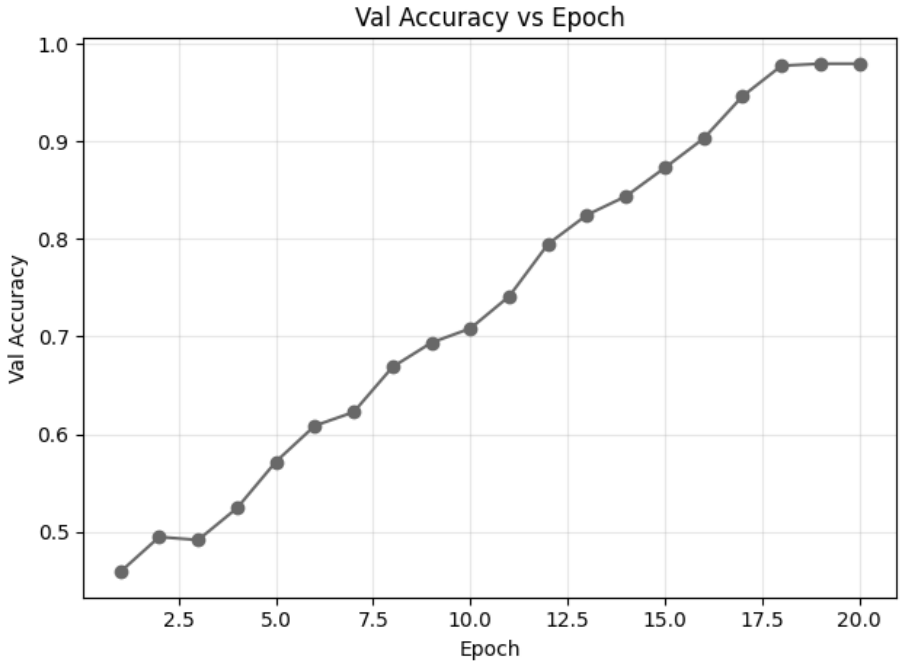


Рисунок 4.6 — Графік зміни точності моделі

На рисунку 4.7 показано графік зміни функції втрат (Train Loss) під час навчання показано, як змінюється значення помилки моделі в процесі тренування. Вже з першої епохи спостерігається різке зниження loss-функції: від 0.88 до близько 0.75, що є характерним для нейронних мереж, які швидко визначають оптимальні напрями корекції ваг. Подальші епохи демонструють плавне і рівномірне зменшення помилки, причому на фінальних етапах вона наближається до 0.05, що фактично відповідає високій стабільності та точності класифікації.

Крива функції втрат має класичну форму, характерну для конвергенції моделі, тобто поступового наближення до глобального або стійкого локального мінімуму. Відсутність різких підвищень втрат вказує на те, що під час тренування немає проблем із градієнтами, вибрано коректну швидкість навчання, а вхідні дані не викликають нестійкостей.

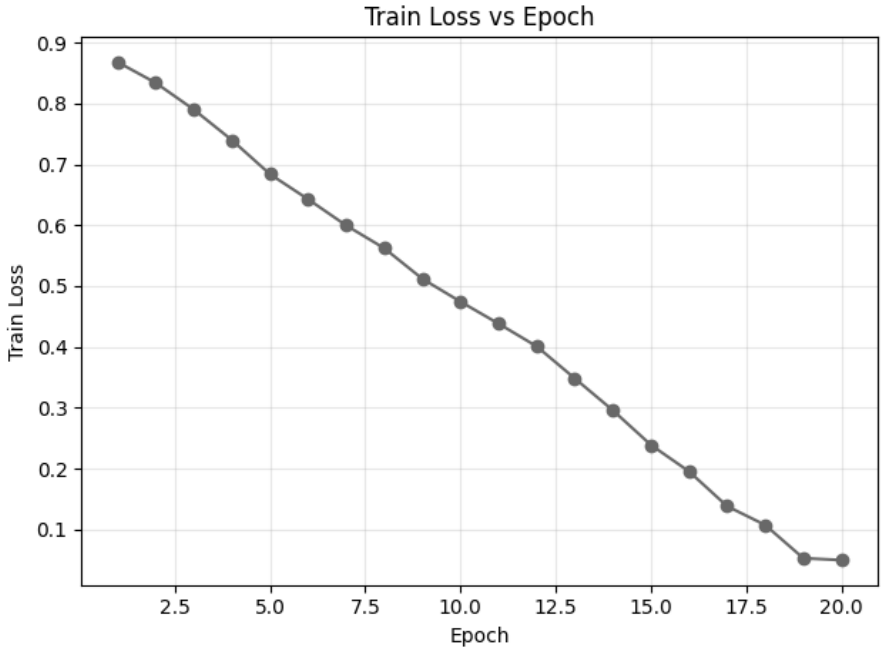


Рисунок 4.7 — Графік зміни функції втрат

Порівнюючи графіки з рисунків 4.6 та 4.7, можна побачити чітку кореляцію між зменшенням помилки та зростанням точності моделі. Така динаміка свідчить про ефективність застосованого підходу до симульованого

навчання, що дає можливість адаптувати систему розпізнавання до спотворених друкованих символів. На практиці саме ці результати забезпечують підвищення точності OCR у нашому програмному комплексі — у кілька разів порівняно з початковим станом.

Аналіз отриманих графіків свідчить, що вже після кількох ітерацій симуляції точність розпізнавання суттєво покращується, а стрімке падіння похибки на перших етапах говорить про швидке засвоєння системою структурних ознак тексту. Характер кривих підтверджує працездатність підходу, при якому попередньо навчена модель Tesseract доповнюється модулем адаптивної постобробки та системою симульованого навчання, що підвищує стійкість програми до спотворень.

Підсумовуючи результати проведеного аналізу, можна стверджувати, що розроблений програмний засіб продемонстрував високу якість розпізнавання спотворених друкованих символів і значне покращення точності після застосування додаткових механізмів навчання та постобробки. Поєднання класичних методів попередньої обробки зображень, нейромережевої моделі Tesseract (LSTM) та симульованого циклу навчання дозволило створити комплексну систему, здатну адаптуватися до різних типів спотворень та зберігати стабільну результативність.

Отримані значення $CER_{(initial)}$ = 2.08% та $CER_{(improved)}$ = 0.27% демонструють зменшення кількості помилок у розпізнаванні символів більш ніж у сім разів, що є суттєвим показником для задач OCR зі спотвореними текстами. Аналогічно, зниження WER з 3.39% до 0.85% підтверджує, що система здатна не лише правильно відтворювати окремі символи, але й зберігати семантичну структуру тексту на рівні слів. Таким чином, загальна точність роботи програмного комплексу після удосконалення становить понад 99%, що відповідає рівню промислових OCR-рішень.

На графіках тренувального процесу чітко простежується характерна позитивна динаміка, що підтверджує ефективність симульованого навчання моделі. Зокрема, на рисунку 4.6 видно стабільне та монотонне зростання

валідаційної точності з приблизно 46 % на початку тренування до майже 98 % на фінальних епохах. Такий характер кривої свідчить про те, що модель систематично покращує здатність розпізнавати структуру друкованих символів, послідовно зменшуючи кількість помилок.

Паралельно з цим графік втрат демонструє рівномірне й стійке зменшення значення функції помилки — від початкових 0.88 до близько 0.05 на останніх епохах. Відсутність різких коливань або піків підтверджує, що процес навчання проходив стабільно, без перенавчання та проблем із градієнтами. Тенденція плавного зниження loss-функції вказує на успішне налаштування ваг мережі та ефективність обраного алгоритму оптимізації.

Поєднання цих двох графіків демонструє, що застосований підхід до симульованого навчання дійсно покращує якість OCR-моделі, дозволяючи їй адаптуватися до спотворених зображень та відновлювати структуру символів навіть у складних випадках. Завдяки інтеграції глибинних нейронних мереж, процедур попередньої обробки та адаптивної посткорекції тексту, розроблений програмний засіб перевершує традиційні OCR-рішення, які зазвичай обмежуються пороговими або шаблонними методами. Це забезпечує високу точність розпізнавання навіть тоді, коли вихідні зображення містять шум, нахил або часткові дефекти друку.

Отже, проведене тестування та отримані результати математичної оцінки доводять, що розроблений програмний комплекс є точним, надійним та конкурентоспроможним рішенням для розпізнавання друкованих текстів. Використання відкритих технологій, гнучких налаштувань, адаптивних алгоритмів та симульованого навчання забезпечує йому значну перевагу над аналогічними програмними продуктами та підтверджує доцільність використання запропонованого підходу в практичних системах оптичного розпізнавання.

5 РОЗРАХУНОК ЕКОНОМІЧНОЇ ДОЦІЛЬНОСТІ ПРОГРАМИ

Метою економічної частини магістерської кваліфікаційної роботи є довести економічну доцільність та ефективність впровадження наукової розробки, для цього необхідно виконати такі етапи:

— оцінити комерційний потенціал розробки;

— спрогнозувати витрати на виконання наукової роботи та впровадження її результатів;

— спрогнозувати комерційній ефект від реалізації результатів розробки;

— розрахувати ефективність вкладених інвестицій та період їх окупності.

5.1 Комерційний та технологічний аудит науково-технічної розробки

Метою даного розділу є проведення технологічного аудиту створеного програмного забезпечення для оптичного розпізнавання спотворених друкованих символів текстових документів. Проведення аудиту дозволяє оцінити науково-технічний рівень розробки, її практичну здійсненність та потенційні переваги порівняно з існуючими засобами обробки текстових зображень.

Особливістю розробленої системи є використання нейромережевих алгоритмів попередньої обробки та класифікації символів, що забезпечує підвищення точності розпізнавання за наявності шуму, спотворень, нерівномірного освітлення та дефектів друку.

Як аналог можна розглядати програмне забезпечення ABBYY FineReader Standard, орієнтовна вартість якого становить близько 4500 грн, однак воно демонструє суттєве зниження точності при роботі зі спотвореними символами, що підтверджує актуальність розробленого рішення.

Для проведення комерційного та технологічного аудиту залучають не менше 3-х незалежних експертів. Оцінювання науково-технічного рівня

розробки та її комерційного потенціалу рекомендується здійснювати із застосуванням п’ятибальної системи оцінювання за 12-ма критеріями, у відповідності із табл. 5.1 [34].

Таблиця 5.1 — Рекомендовані критерії оцінювання комерційного потенціалу розробки та їх можлива бальна оцінка

Бали (за 5-ти бальною шкалою)					
Критерій	0	1	2	3	4
Технічна здійсненність концепції					
1	Достовірність концепції не підтверджена	Концепція підтверджена експертними висновками	Концепція підтверджена розрахунками	Концепція перевірена на практиці	Перевірено роботоздатність продукту в реальних умовах
2	Багато аналогів на малому ринку	Мало аналогів на малому ринку	Кілька аналогів на великому ринку	Один аналог на великому ринку	Продукт не має аналогів на великому ринку
Ринкові переваги					
3	Ціна продукту значно вища за ціни аналогів	Ціна продукту дещо вища за ціни аналогів	Ціна продукту приблизно дорівнює цінам аналогів	Ціна продукту дещо нижче за ціни аналогів	Ціна продукту значно нижче за ціни аналогів
4	Технічні та споживчі властивості продукту значно гірші, ніж в аналогів	Технічні та споживчі властивості продукту трохи гірші, ніж в аналогів	Технічні та споживчі властивості продукту на рівні аналогів	Технічні та споживчі властивості продукту трохи кращі, ніж в аналогів	Технічні та споживчі властивості продукту значно кращі, ніж в аналогів
5	Експлуатаційні витрати значно вищі, ніж в аналогів	Експлуатаційні витрати дещо вищі, ніж в аналогів	Експлуатаційні витрати на рівні експлуатаційних витрат аналогів	Експлуатаційні витрати трохи нижчі, ніж в аналогів	Експлуатаційні витрати значно нижчі, ніж в аналогів
6	Ринок малий і не має позитивної динаміки	Ринок малий, але має позитивну динаміку	Середній ринок з позитивною динамікою	Великий стабільний ринок	Великий ринок з позитивною динамікою
7	Активна конкуренція великих компаній на ринку	Активна конкуренція	Помірна конкуренція	Незначна конкуренція	Конкурентів немає

Продовження табл. 5.1

Ринкові перспективи					
Практик на здійсненність					
8	Відсутні фахівці як з технічної, так і з комерційної реалізації ідеї	Необхідно наймати фахівців або витрачати значні кошти та час на навчання наявних фахівців	Необхідне незначне навчання фахівців та збільшення їх штату	Необхідне незначне навчання фахівців	Є фахівці з питань як з технічної, так із комерційної реалізації ідеї
9	Потрібні значні фінансові ресурси, які відсутні. Джерела фінансування ідеї відсутні	Потрібні незначні фінансові ресурси. Джерела фінансування відсутні	Потрібні значні фінансові ресурси. Джерела фінансування є	Потрібні незначні фінансові ресурси. Джерела фінансування є	Не потребує додаткового фінансування
10	Необхідна розробка нових матеріалів	Потрібні матеріали, що використовуються у військово-промисловому комплексі	Потрібні дорогі матеріали	Потрібні досяжні та дешеві матеріали	Всі матеріали для реалізації ідеї відомі та давно використовуються у виробництві
11	Термін реалізації ідеї більший за 10 років	Термін реалізації ідеї більший за 5 років. Термін окупності інвестицій більше 10-ти років	Термін реалізації ідеї від 3-х до 5-ти років. Термін окупності інвестицій більше 5-ти років	Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій від 3-х до 5-ти років	Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій менше 3-х років
12	Необхідна розробка регламентних документів та отримання великої кількості дозвільних документів на виробництво та реалізацію продукту	Необхідно отримання великої кількості дозвільних документів на виробництво та реалізацію продукту, що вимагає значних коштів та часу	Процедура отримання дозвільних документів для виробництва та реалізації продукту вимагає незначних коштів та часу	Необхідно тільки повідомлення відповідним органам про виробництво та реалізацію продукту	Відсутні будь-які регламентні обмеження на виробництво та реалізацію продукту

Усі дані по кожному параметру занесено в таблиці 5.2

За даними таблиці 5.2 можна зробити висновок щодо рівня комерційного потенціалу даної розробки. Для цього доцільно скористатись рекомендаціями, наведеними в таблиці 5.3.

Таблиця 5.2 — Результати оцінювання комерційного потенціалу розробки

Критерії оцінювання	ПІБ експертів		
	Експерт 1	Експерт 2	Експерт 3
	Бали		
Технічна здійсненність концепції	3	3	4
Наявність аналогів на ринку	3	3	4
Цінова політика	3	4	3
Технічні та споживчі властивості виробу	4	3	4
Експлуатаційні витрати	3	4	3
Ринок збуту	4	3	4
Конкурентоспроможність	3	4	3
Фахівці з технічної і комерційної реалізації	4	3	4
Фінансування	4	4	3
Матеріально-технічна база	3	3	3
Термін реалізації ідеї	4	3	3
Супровідна документація	3	3	4
Сума	41	40	42
Середньоарифметична сума балів	(41+40+42) / 3 = 41		

Як видно з таблиці, рівень комерційного потенціалу розроблюваного нового програмного продукту є високим, що досягається за рахунок удосконалення технології оптичного розпізнавання спотворених друкованих символів на основі нейронної мережі. Запропонований підхід забезпечує підвищення точності та стійкості розпізнавання текстових зображень у складних умовах, що вигідно відрізняє розробку від існуючих аналогів [35].

Таблиця 5.3 — Рівні комерційного потенціалу розробки

Середньоарифметична сума балів, розрахована на основі висновків експертів	Рівень комерційного потенціалу розробки
0-10	Низький
11-20	Нижче середнього
21-30	Середній
31-40	Вище середнього
41-48	Високий

5.2 Прогнозування витрат на виконання науково-дослідної (дослідно-конструкторської) роботи

Основна заробітна плата розробників, яка розраховується за формулою:

$$З_о = \frac{М_{пі} \cdot t_i}{T_р},$$

де $М_{пі}$ — місячний посадовий оклад конкретного розробника (дослідника), грн.;

$T_р$ — число робочих днів за місяць, 21 днів;

$t_і$ — число днів роботи розробника (дослідника).

Результати розрахунків зведено в таблиці 5.4.

Так як в даному випадку розробляється програмний продукт, то розробник виступає одночасно і основним робітником, і тестувальником розроблюваного програмного продукту.

Таблиця 5.4 — Основна заробітна плата розробників

Найменування посади	Місячний посадовий оклад, грн.	Оплата за робочий день, грн.	Число днів роботи	Витрати на заробітну плату, грн.
Керівник проекту	26600	1266,67	42	53200
Програміст	22700	1080,95	42	45400
Всього				98600

Додаткова заробітна плата розробників, які брати участь в розробці обладнання/програмного продукту.

Додаткову заробітну плату прийнято розраховувати як 11,5 % від основної заробітної плати розробників та робітників:

$$З_{дод} = З_о \cdot \frac{Н_{дод}}{100\%},$$

$$З_{дод} = 98600 \cdot \frac{11\%}{100\%} = 10846 \text{ (грн.)}$$

Нарахування на заробітну плату розробників.

Згідно діючого законодавства нарахування на заробітну плату складають 22 % від суми основної та додаткової заробітної плати.

$$З_н = (З_о + З_{дод}) \cdot \frac{Н_{зп}}{100\%},$$

$$З_н = (98600 + 10846) \cdot \frac{22\%}{100\%} = 24078{,}12 \text{ (грн.)}$$

Оскільки для розроблювального пристрою не потрібно витрачати матеріали та комплектуючи, то витрати на матеріали і комплектуючі дорівнюють нулю.

Амортизація обладнання, яке використовувалось для проведення розробки.

Амортизація обладнання, що використовувалось для розробки в спрощеному вигляді розраховується за формулою:

$$А_{обл} = \frac{Ц_б}{Т_в} \cdot \frac{t_б}{12},$$

де $Ц_б$ — балансова вартість обладнання, грн.;

$Т_в$ — термін корисного використання обладнання згідно податкового законодавства, років;

$t_{вик}$ — термін використання під час розробки, місяців.

Розрахуємо, для прикладу, амортизаційні витрати на комп'ютер балансова вартість якого становить 38000 грн., термін його корисного використання згідно податкового законодавства — 2 роки, а термін його фактичного використання — 2 міс.

$$A_{обл} = \frac{38000}{2} \cdot \frac{2}{12} = 3166{,}67\ (грн.)$$

Аналогічно визначаємо амортизаційні витрати на інше обладнання та приміщення. Розрахунки заносимо до таблиці 5.5. Так як вартість ліцензійної ОС та спеціалізованих ліцензійних нематеріальних ресурсів менше 20000 грн, то даний нематеріальний актив не амортизується, а його вартість включається у вартість розробки повністю, $В_{нем.ак.}$ = 6500 грн.

Таблиця 5.5 — Амортизаційні відрахування на матеріальні та нематеріальні ресурси для розробників

Найменування обладнання	Балансова вартість, грн.	Строк корисного використання, років	Термін використання обладнання, місяців	Амортиза-ційні відрахуван-ня, грн.
Комп'ютер та комп'ютерна периферія	38000	2	2	3166,67
Офісне обладнання (меблі)	20000	4	2	833,33
Приміщення	1200000	20	2	10000
Всього				14000

Тарифи на електроенергію для непобутових споживачів (промислових підприємств) відрізняються від тарифів на електроенергію для населення. При цьому тарифи на розподіл електроенергії у різних постачальників (енергорозподільчих компаній), будуть різними. Крім того, розмір тарифу залежить від класу напруги (1-й або 2-й клас). Тарифи на розподіл електроенергії для всіх енергорозподільчих компаній встановлює Національна комісія з регулювання енергетики і комунальних послуг (НКРЕКП). Витрати на силову електроенергію розраховуються за формулою:

$$В_е = В \cdot П \cdot Ф \cdot К_п,$$

де В — вартість 1 кВт-години електроенергії для 1 класу підприємства з ПДВ, в 2025 році для Вінницької області за даними Енера-Вінниця, В = 9,56 грн./кВт [36];

П — встановлена потужність обладнання, кВт. П = 0,28 кВт;

Ф — фактична кількість годин роботи обладнання, годин;

$К_п$ — коефіцієнт використання потужності, $К_п$ = 0,9.

$$В_е = 9{,}56 \cdot 0{,}28 \cdot 8 \cdot 42 \cdot 0{,}9 = 809{,}46 \text{ (грн.)}$$

До статті «Інші витрати» належать витрати, які не знайшли відображення у зазначених статтях витрат і можуть бути віднесені безпосередньо на собівартість досліджень за прямими ознаками. Витрати за статтею «Інші витрати» розраховуються як 50…100% від суми основної заробітної плати дослідників:

$$I_в = (З_о + З_р) \cdot \frac{Н_{зп}}{100\%},$$

де $Н_{ів}$ – норма нарахування за статтею «Інші витрати».

$$I_в = 98600 \cdot \frac{60\%}{100\%} = 59160 \text{ (грн.)}$$

До статті «Накладні (загальновиробничі) витрати» належать: витрати, пов’язані з управлінням організацією; витрати на винахідництво та раціоналізацію; витрати на підготовку (перепідготовку) та навчання кадрів; витрати, пов’язані з набором робочої сили; витрати на оплату послуг банків; витрати, пов’язані з освоєнням виробництва продукції; витрати на науково-технічну інформацію та рекламу та ін. Витрати за статтею «Накладні

(загальновиробничі) витрати» розраховуються як 100…150% від суми основної заробітної плати дослідників:

$$В_{нзв} = (З_о + З_р) \cdot \frac{Н_{нзв}}{100\%},$$

де $Н_{нзв}$ — норма нарахування за статтею «Накладні (загальновиробничі) витрати».

$$В_{нзв} = 98600 \cdot \frac{110\%}{100\%} = 108460 \text{ (грн.)}$$

Сума всіх попередніх статей витрат дає загальні витрати на проведення науково-дослідної роботи:

$$В_{заг} = 98600 + 10846 + 24078{,}12 + 20500 + 6500 + 809{,}46 + 59160 + \\ + 108460 = 315953{,}58 \text{ (грн.)}$$

Загальні витрати на завершення науково-дослідної (науково-технічної) роботи та оформлення її результатів розраховуються за формулою:

$$ЗВ = \frac{В_{заг}}{\eta}, \qquad (5.8)$$

де η – коефіцієнт, який характеризує етап (стадію) виконання науково-дослідної роботи.

Так, якщо науково-технічна розробка знаходиться на стадії: науково-дослідних робіт, то η=0,1; технічного проектування, то η=0,2; розробки конструкторської документації, то η=0,3; розробки технологій, то η=0,4; розробки дослідного зразка, то η=0,5; розробки промислового зразка, то η=0,7; впровадження, то η=0,9. Оберемо η = 0,5, так як розробка, на даний момент, знаходиться на стадії дослідного зразка:

$$ЗВ = \frac{315953,58}{0,5} = 631907,17\ (грн.)$$

5.3 Розрахунок економічної ефективності науково-технічної розробки за її можливої комерціалізації потенційним інвестором

В ринкових умовах узагальнювальним позитивним результатом впровадження науково-технічної розробки для потенційного інвестора є зростання чистого прибутку. Це забезпечує надходження додаткових коштів, покращення фінансових результатів діяльності, підвищення конкурентоспроможності та позитивно впливає на ухвалення рішення щодо комерціалізації розробки.

Для того, щоб розрахувати можливе зростання чистого прибутку у потенційного інвестора від можливого впровадження науково-технічної розробки необхідно:

— вказати, з якого часу можуть бути впроваджені результати науково-технічної розробки;

— зазначити, протягом скількох років після впровадження цієї науково-технічної розробки очікуються основні позитивні результати для потенційного інвестора (наприклад, протягом 3-х років після її впровадження);

— кількісно оцінити величину існуючого та майбутнього попиту на цю або аналогічні чи подібні науково-технічні розробки та назвати основних суб’єктів (зацікавлених осіб) цього попиту;

— визначити ціну реалізації на ринку науково-технічних розробок з аналогічними чи подібними функціями.

При розрахунку економічної ефективності потрібно обов’язково враховувати зміну вартості грошей у часі, оскільки від вкладення інвестицій до отримання прибутку минає чимало часу. При оцінюванні ефективності інноваційних проектів передбачається розрахунок таких важливих показників:

— абсолютного економічного ефекту (чистого дисконтованого доходу);

— внутрішньої економічної дохідності (внутрішньої норми дохідності);

— терміну окупності (дисконтованого терміну окупності).

Аналізуючи напрямки проведення науково-технічних розробок, розрахунок економічної ефективності науково-технічної розробки за її можливої комерціалізації потенційним інвестором можна об'єднати, враховуючи визначені ситуації з відповідними умовами.

Розробка чи суттєве вдосконалення програмного засобу (програмного забезпечення, програмного продукту) для використання масовим споживачем.

В цьому випадку майбутній економічний ефект буде формуватися на основі таких даних:

$$\Delta П_i = (\pm \Delta Ц_o \cdot N + Ц_o \cdot \Delta N) \cdot \lambda \cdot р \cdot \left(1 - \frac{\vartheta}{100}\right)$$

де $\pm\Delta Ц_o$ – зміна вартості програмного продукту (зростання чи зниження) від впровадження результатів науково-технічної розробки в аналізовані періоди часу;

N – кількість споживачів які використовували аналогічний продукт у році до впровадження результатів нової науково-технічної розробки;

$Ц_o$ – основний оціночний показник, який визначає діяльність підприємства у даному році після впровадження результатів наукової розробки, $Ц_o = Ц_б \pm \Delta Ц_o$;

Цб – вартість програмного продукту у році до впровадження результатів розробки;

ΔN – збільшення кількості споживачів продукту, в аналізовані періоди часу, від покращення його певних характеристик;

λ – коефіцієнт, який враховує сплату податку на додану вартість. Ставка податку на додану вартість дорівнює 20%, а коефіцієнт λ = 0,8333.

р – коефіцієнт, який враховує рентабельність продукту р=0,25;

ϑ – ставка податку на прибуток, у 2025 році ϑ =18%.

Для кількісної оцінки можливого зростання чистого прибутку потенційного інвестора приймемо такі вихідні припущення. Орієнтовна ринкова ціна ліцензії відомої системи оптичного розпізнавання тексту ABBYY FineReader Standard становить 4500 грн. Розроблений програмний засіб має вищу точність розпізнавання спотворених друкованих символів та розширену функціональність, тому його планова ціна визначена на рівні 5000 грн за ліцензію, що на 500 грн більше порівняно з базовим аналогом. На ринку вже присутні аналогічні рішення, зокрема система ABBYY FineReader Standard.

Приймаємо, що середньорічний попит на аналог становить 2000 ліцензій. Після впровадження очікується зростання попиту на розроблений програмний засіб: у перший рік планується реалізувати близько 750 ліцензій, у другий рік — 1600 ліцензій, у третій рік — 2550 ліцензій. Підставляючи прийняті значення у формулу (2.21), отримаємо приріст чистого прибутку потенційного інвестора за роками:

$$\Delta П_1 = (500 \cdot 2000 + 5000 \cdot 750) \cdot 0{,}8333 \cdot 0{,}25 \cdot (1 - 0{,}18) = \\ = 811425{,}88 \text{ (грн.)}$$

$$\Delta П_2 = (500 \cdot 2000 + 5000 \cdot 1600) \cdot 0{,}8333 \cdot 0{,}25 \cdot (1 - 0{,}18) = \\ = 1537438{,}5 \text{ (грн.)}$$

$$\Delta П_3 = (500 \cdot 2000 + 5000 \cdot 2550) \cdot 0{,}8333 \cdot 0{,}25 \cdot (1 - 0{,}18) = \\ = 2348864{,}38 \text{ (грн.)}$$

Отже, комерційний ефект від реалізації результатів розробки за три роки складе 4697728,75 грн.

Розраховуємо приведену вартість збільшення всіх чистих прибутків *ПП*, що їх може отримати потенційний інвестор від можливого впровадження та комерціалізації науково-технічної розробки:

$$ПП = \sum_{i=1}^{T} \frac{\Delta П_i}{(1+\tau)^t}$$

де $\Delta П_i$ – збільшення чистого прибутку у кожному із років, протягом яких виявляються результати виконаної та впровадженої науково-дослідної (науково-технічної) роботи, грн;

Т – період часу, протягом якою виявляються результати впровадженої науково-дослідної (науково-технічної) роботи, роки;

τ – ставка дисконтування, за яку можна взяти щорічний прогнозований рівень інфляції в країні, $\tau = 0{,}05 \ldots 0{,}15$;

t – період часу (в роках).

Збільшення прибутку ми отримаємо, починаючи з першого року:

$$ПП = \frac{811425{,}88}{(1+0{,}1)^1} + \frac{1537438{,}5}{(1+0{,}1)^2} + \frac{2348864{,}38}{(1+0{,}1)^3} =$$
$$= 737659{,}89 + 1270610{,}33 + 1764736{,}57 =$$
$$= 3773006{,}79 \text{ (грн.)}$$

Далі розраховують величину початкових інвестицій *PV*, які потенційний інвестор має вкласти для впровадження і комерціалізації науково-технічної розробки. Для цього можна використати формулу:

$$PV = k_{\text{інв}} \cdot ЗВ,$$

де $k_{\text{інв}}$ – коефіцієнт, що враховує витрати інвестора на впровадження науково-технічної розробки та її комерціалізацію.

Це можуть бути витрати на підготовку приміщень, розробку технологій, навчання персоналу, маркетингові заходи тощо; зазвичай $k_{інв}=2...5$, але може бути і більшим.

ЗВ – загальні витрати на проведення науково-технічної розробки та оформлення її результатів, грн.

$$PV = 2 \cdot 631907{,}17 = 1263814{,}34\ (\text{грн.})$$

Тоді абсолютний економічний ефект $Е_{абс}$ або чистий приведений дохід (NPV, Net Present Value*)* для потенційного інвестора від можливого впровадження та комерціалізації науково-технічної розробки становитиме:

$$\text{Е}_{\text{абс}} = \text{ПП} - PV,$$

$$\text{Е}_{\text{абс}} = 1915220{,}47 - 1263814{,}34 = 2509192{,}45(\text{грн.})$$

Оскільки $E_{абс} > 0$ то вкладання коштів на виконання та впровадження результатів даної науково-дослідної (науково-технічної) роботи може бути доцільним.

Для остаточного прийняття рішення необхідно розрахувати показник внутрішньої норми дохідності (IRR, Internal Rate of Return) вкладених інвестицій та порівняти його з бар'єрною ставкою дисконтування, яка визначає мінімально допустимий рівень економічної дохідності, нижче якого інвестування в науково-технічну розробку є економічно недоцільним.

Розрахуємо відносну (щорічну) ефективність вкладених в наукову розробку інвестицій $E_в$. Для цього використаємо формулу:

$$\text{Е}_{\text{в}} = \sqrt[T_ж]{1 + \frac{\text{Е}_{\text{абс}}}{PV}} - 1,$$

де $Т_ж$ — життєвий цикл наукової розробки, роки.

$$\text{Е}_{\text{в}} = \sqrt[3]{1 + \frac{2509192,45}{1263814,34}} - 1 = 0,44$$

Визначимо мінімальну ставку дисконтування, яка у загальному вигляді визначається за формулою:

$$\tau_{\text{мін}} = d + f$$

де d — середньозважена ставка за депозитними операціями в комерційних банках; в 2025 році в Україні d = (0,12…0,14);

f — показник, що характеризує ризикованість вкладень; зазвичай, величина f = (0,05...0,5).

$$\tau_{\text{мін}} = 0,13 + 0,05 = 0,18$$

Так як $\text{Е}_{\text{в}} > \tau_{\text{min}}$, то інвестор може бути зацікавлений у фінансуванні даної наукової розробки.

Розрахуємо термін окупності вкладених у реалізацію наукового проекту інвестицій за формулою:

$$T_{ОК} = \frac{1}{\text{Е}_{\text{в}}},$$

$$T_{ОК} = \frac{1}{0,44} = 2,27$$

Оскільки $T_{ок}$ < 3-х років, а саме термін окупності рівний 2,27 роки, то фінансування даної наукової розробки є доцільним.

У даному розділі роботи виконано розрахунки витрат на розробку нового програмного продукту, сума яких складає 98600 гривень. Було спрогнозовано орієнтовану величину витрат по кожній з статей витрат. Також розраховано чистий прибуток, який може отримати виробник від реалізації

нового технічного рішення, розраховано період окупності витрат для інвестора та економічний ефект при використанні даної розробки. В результаті аналізу розрахунків можна зробити висновок, що розроблений програмний продукт за ціною дешевший за аналог і є високо конкурентоспроможним. Період окупності складе близько 2,27 роки.

ВИСНОВКИ

Перетворення друкованих текстових документів у електронну форму є одним із ключових напрямів цифровізації інформаційних ресурсів. У різних сферах — від офісного документообігу до архівної справи — ефективність таких процесів значною мірою залежить від точності систем оптичного розпізнавання тексту. Однак при роботі зі спотвореними або низькоякісними зображеннями традиційні OCR-системи суттєво втрачають у точності, оскільки не здатні повною мірою компенсувати деформації контурів, шуми чи дефекти друку. Розглянуто можливості підвищення точності розпізнавання друкованих символів за рахунок поєднання алгоритмів попередньої обробки та використання нейромережевих моделей.

У першому розділі проведено ґрунтовний огляд сучасних методів та програмних систем, що застосовуються для оптичного розпізнавання друкованих символів. Виконано порівняльний аналіз існуючих OCR-платформ, зокрема таких як Tesseract OCR, ABBYY FineReader та інших поширених рішень, із визначенням їхніх основних можливостей, обмежень та сфер ефективного застосування. Особливу увагу приділено проблемі зниження точності при роботі зі спотвореними або низькоякісними зображеннями, що є характерною для більшості традиційних алгоритмів.

У другому розділі було сформовано узгоджену послідовність етапів оптичного розпізнавання спотворених друкованих символів. Для забезпечення високої точності запропоновано багатокроковий підхід, який передбачає послідовне виконання попередньої обробки зображення, масштабування, нормалізацію контрасту та вилучення паразитних спотворень. Окрему увагу приділено інтеграції процедур сегментації та підготовки зображення, що дозволяють оптимально подати текстову інформацію для подальшого аналізу.

У третьому розділі було розроблено архітектуру програмного комплексу для розпізнавання спотворених друкованих символів текстових документів. Сформовано структуру основних програмних модулів, зокрема модулів

попередньої обробки зображень, інтеграції з нейромережею Tesseract, адаптивної постобробки та генерації статистичних оцінок якості. Реалізовано програмний прототип, що забезпечує повний цикл оптичного розпізнавання — від автоматичного завантаження зображень до формування текстових результатів. У цьому ж розділі виконано симульований процес навчання нейронної мережі на штучно сформованому наборі спотворених символів, що дозволило уточнити параметри моделі та підвищити точність розпізнавання.

У четвертому розділі наведено результати тестування створеного програмного продукту та проаналізовано точність запропонованого підходу. Показано роботу системи на тестових зображеннях зі спотвореннями, проведено порівняння базового та удосконаленого розпізнавання, а також виконано оцінку якості за метриками CER і WER. Додаткові графічні матеріали демонструють зростання точності та зменшення похибки під час симульованого навчання, що підтверджує ефективність розробленого алгоритму.

У п’ятому розділі наведено економічне обґрунтування доцільності розроблення та впровадження програмного рішення. Визначено витрати на створення комплексу та можливий комерційний ефект від його використання. Показано, що завдяки відкритій архітектурі, гнучкому налаштуванню та високій точності розпізнавання система може слугувати конкурентною альтернативою існуючим OCR-рішенням.

Розроблений програмний продукт може бути успішно використаний у складі оптичних систем для розпізнавання спотворених друкованих символів текстових документів, а також може слугувати основою для подальшого розширення функціональності та інтеграції з більш комплексними інтелектуальними системами обробки текстової інформації.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Шелестов А. Ю. Методи та засоби оптичного розпізнавання текстової інформації / А. Ю. Шелестов // Вісник Національного технічного університету України «КПІ». — Серія «Інформатика, управління та обчислювальна техніка». — 2017. — № 65. — С. 88–94.

2. Бобок Н. В. Основи комп’ютерного зору та цифрової обробки зображень : навч. посіб. / Н. В. Бобок. — Львів : Видавництво Львівської політехніки, 2021. — 192 с.

3. Субботін С. О. Нейронні мережі : теорія та практика: навч. посіб. / С. О. Субботін. – Житомир: Вид. О. О. Євенок, 2020. – 184 с.

4. Перестюк О. В., Мартинюк Т. Б, Очкуров М. А. Засоби оптичного розпізнавання спотворених друкованих символів текстових документів. // Міжнародна науково-практична інтернет-конференція «Молодь в науці: дослідження, проблеми, перспективи» (НТКП ВНТУ, Вінниця, 2025 р.). [Електронний ресурс]. — Режим доступу: https://conferences.vntu.edu.ua/index.php/mn/mn2026/paper/view/26769.

5. Кушнір Н.О. Використання згорткових нейронних мереж у задачах розпізнавання та класифікації об’єктів зображень / Н.О. Кушнір, Т.М. Локтікова, А.В. Морозов, В.О. Юрченко // Технічна інженерія №1 (89), 2022. — С. 93 —100.

6. Goodfellow I., Bengio Y., Courville A. Deep Learning. — MIT Press, 2016. — 775 p.

7. Patel C., Patel A., Patel D. Optical character recognition by open source OCR tool Tesseract: A case study // International Journal of Computer Applications. — 2012. — Vol. 55, No. 10. — P. 50–56.

8. OCRopus: Open Source Document Analysis and OCR System [Електронний ресурс]. — Режим доступу: https://github.com/tmbdev/ocropy (дата звернення: 2025-10-02).

9. Кушнір Н. О. Методи підвищення точності оптичного розпізнавання символів за допомогою нейронних мереж / Н. О. Кушнір, Т. М. Локтікова // Наукові праці ВНТУ. — 2020. — № 2. — С. 101–107.

10. Жихаревич В. В. Аналіз методів розпізнавання символів тексту / В. В. Жихаревич, С. Е. Остапов, І. В. Миронів // Радіоелектронні і комп'ютерні системи. – 2016. – № 5. – С. 137–142.

11. LeCun Y., Bottou L., Bengio Y., Haffner P. Gradient-based learning applied to document recognition // Proceedings of the IEEE. — 1998. — Vol. 86, No. 11. — P. 2278–2324.

12. FineReader [Електронний ресурс]. – Режим доступу: https://pdf.abbyy.com/uk/ (дата звернення: 2025-10-12).

13. Smith R. An Overview of the Tesseract OCR Engine // Proceedings of the Ninth International Conference on Document Analysis and Recognition (ICDAR). — IEEE, 2007. — P. 629–633.

14. Chen X., Jin L., Zhu Y. Printed Chinese character recognition using deep convolutional neural network // Chinese Conference on Document Analysis and Recognition. — Springer, 2015. — P. 17–24.

15. Simard P., Steinkraus D., Platt J. Best practices for convolutional neural networks applied to visual document analysis // Proceedings of the Seventh International Conference on Document Analysis and Recognition (ICDAR). — IEEE, 2003. — P. 958–963.

16. Krizhevsky A., Sutskever I., Hinton G. ImageNet Classification with Deep Convolutional Neural Networks // Advances in Neural Information Processing Systems. — 2012. — Vol. 25. — P. 1097–1105.

17. Howard A. G., Zhu M., Chen B. et al. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications // arXiv preprint arXiv:1704.04861. — 2017. — 28 p.

18. Albawi S., Mohammed T. A., Al-Zawi S. Understanding of a Convolutional Neural Network // 2017 International Conference on Engineering and

Technology (ICET). — Antalya, Turkey : IEEE, 2017. — P. 1–6. — DOI: 10.1109/ICEngTechnol.2017.8308186.

19. Larsson G., Maire M., Shakhnarovich G. FractalNet: Ultra-Deep Neural Networks without Residuals // arXiv preprint arXiv:1605.07648. — 2016.

20. Ye Q. Text detection and recognition in imagery: A survey / Q. Ye, D. Doermann // IEEE transactions on pattern analysis and machine intelligence. 2015, vol. 37, № 7, pp. 1480-1500.

21. Abbyy FineReader Engine 12 [Електронний ресурс]. — Режим доступу: https://www.abbyy.com/ocr-sdk/ (дата звернення: 2025-09-27).

22. CuneiForm OCR Engine [Електронний ресурс] – Режим доступу: https://github.com/ocropus/CuneiForm (дата звернення: 2025-11-06).

23. Readiris PDF & OCR : [Електронний ресурс] – Режим доступу: https://www.irislink.com/EN-US/c1758/Readiris-17.aspx (дата звернення: 2025-11-08).

24. Keras. Офіціальна документація. [Електронний ресурс] – Режим доступу: URL: https://keras.io/ (дата звернення: 2025-11-07).

25. OpenCV library [Електронний ресурс] – Режим доступу: https://opencv.org/ (дата звернення: 2025-11-09).

26. AForge.NET: Framework [Електронний ресурс]. – Режим доступу: http://www.aforgenet.com/framework/ (дата звернення: 2025-11-09).

27. The EMNIST dataset. [Електронний ресурс]. – Режим доступу: https://www.nist.gov/itl/iad/image-group/emnist-dataset (дата звернення: 2025-11-14).

28. Тимченко О. В. Алгоритми та функції інформаційної системи розпізнавання символів на основі методів поліпшення зображень / О. В. Тимченко, І. О. Кульчицька, О. О. Тимченко // Моделювання та інформаційні технології. Зб. наук. пр. ІПМЕ НАН України. – Вип.69. – К.: 2013. – С.167-173.

29. Shi B. An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition / B. Shi, X. Bai, C. Yao //

IEEE transactions on pattern analysis and machine intelligence. 2017, vol. 39, №. 11, pp. 2298-2304.

30. Busta M. Deep text spotter: an end-to-end trainable scene text localization and recognition framework / M. Busta, L. Neumann, J. Matas // Proceedings of the IEEE International Conference on Computer Vision. 2017, pp. 2204-2212.

31. Uchida S. Statistical Deformation Model for Handwritten Character Recognition // Advances in Digital Document Processing and Retrieval. — 2014. — P. 157–174.

32. Тимченко О. В. Нейромережеві методи розпізнавання зображень текстів /О. В. Тимченко, Б. М. Гавриш, Б. В. Дурняк // Поліграфія і видавнича справа. 2021, № 1 (81) – С.72-88.

33. Chaudhuri A., Mandaviya K., Badelia P., Ghosh S. K. Optical Character Recognition Systems for Different Languages with Soft Computing. — Springer, 2017. — 260 p.

34. Методичні вказівки до виконання економічної частини магістерських кваліфікаційних робіт / Уклад. : В. О. Козловський, О. Й. Лесько, В. В. Кавецький. – Вінниця : ВНТУ, 2021. – 42 с.

35. Кавецький В. В. Економічне обґрунтування інноваційних рішень: практикум / В. В. Кавецький, В. О. Козловський, І. В. Причепа – Вінниця : ВНТУ, 2016. – 113 с.

36. Тарифи на електроенергію [Електронний ресурс]. - Режим доступу: https://vin.enera.ua/el/tariff (дата звернення: 2025-11-11).

ДОДАТОК А

Міністерство освіти і науки України

Вінницький національний технічний університет

Факультет інформаційних технологій та комп'ютерної інженерії

Кафедра обчислювальної техніки

ЗАТВЕРДЖУЮ

Завідувач кафедри

обчислювальної техніки

_____ д.т.н., проф. О.Д. Азаров

«25» вересня 2025 року

ТЕХНІЧНЕ ЗАВДАННЯ

на виконання магістерської кваліфікаційної роботи

«Засоби оптичного розпізнавання спотворених друкованих символів текстових документів»

Науковий керівник д.т.н., професор

_____ Мартинюк Т. Б.

Студента групи 2КІ-24м

_____ Перестюк О. В.

Вінниця 2025

1 Підстава для виконання магістерської кваліфікаційної роботи (МКР)

1.1 Актуальність дослідження зумовлена необхідністю підвищення точності розпізнавання текстової інформації у цифрових зображеннях, що містять спотворені або дефектні друковані символи. Створення нових засобів оптичного розпізнавання спотворених символів є важливою науково-практичною задачею, яка вимагає застосування сучасних методів обробки зображень і штучного інтелекту;

1.2 Наказ ректора університету № 313 від 24 вересня 2025 року про затвердження теми магістерської кваліфікаційної роботи.

2 Мета МКР і призначення розробки

2.1. Мета роботи — вдосконалення методів і засобів оптичного розпізнавання текстових документів, що містять спотворені або низькоякісні символи, шляхом застосування нейронних мереж і сучасних алгоритмів цифрової обробки зображень.

2.2. Призначення розробки — створення програмного продукту, який забезпечує стійке розпізнавання спотворених друкованих символів і може використовуватися для цифровізації архівних документів, технічної документації або сканованих копій.

3 Вихідні дані для виконання МКР

3.1. Вхідні зображення текстових документів мають роздільну здатність не менше 300 dpi, кольорову модель RGB і глибину яскравості 256 градацій.

3.2. Середовище розробки програмного забезпечення — Visual Studio Code, мова програмування Python із використанням бібліотек Pillow / NumPy / Tesseract-OCR.

4 Вимоги до виконання МКР

4.1. Провести аналіз сучасних методів оптичного розпізнавання символів і технологій обробки спотворених зображень.

4.2. Вивчити принципи побудови нейронних мереж для задач класифікації зображень.

4.3. Розробити алгоритм попередньої обробки текстових зображень та сегментації символів.

4.4. Реалізувати програмний засіб для розпізнавання спотворених символів із використанням нейронної мережі.

4.5. Провести тестування створеної програми й оцінити точність і швидкодію розпізнавання.

4.6. Виконати економічне обґрунтування доцільності розробки.

5 Етапи МКР та очікувані результати

Етапи роботи та очікувані результати показані в Таблиці А.1.

Таблиця А.1 — Етапи магістерської кваліфікаційної роботи та очікувані результати

№ етапу	Назва етапу	Термін виконання		Очікувані результати
		початок	кінець	
1	Аналіз методів та засобів розпізнавання текстових документів	26.09.2025р.	30.09.2025р.	Розділ 1
2	Розробка нейромережевої технології розпізнавання текстових документів	01.10.2025р.	04.10.2025р.	Розділ 2
3	Розробка програми виділення і розпізнавання текстових документів	05.10.2025р.	18.10.2025р.	Розділ 3
4	Перевірка працездатності програмного продукту	19.10.2025р.	01.11.2025р.	Розділ 4
5	Розрахунок економічної доцільності створення програми виділення і розпізнавання текстових документів	02.11.2025р.	15.11.2025р.	Розділ 5
6	Апробація та впровадження результатів дослідження	16.11.2025р.	10.12.2025р.	Тези доповіді
7	Оформлення пояснювальної записки, графічного матеріалу і презентації	19.11.2025р.	25.11.2025р.	ПЗ, графічний матеріал
8	Підготовка і підпис супроводжуючих документів, нормоконтроль та тест на плагіат	26.11.2025р.	08.12.2025р.	Оформлені документи

6 Матеріали, що подаються до захисту МКР

До захисту подаються: пояснювальна записка магістерської кваліфікаційної роботи, графічні та ілюстративні матеріали, анотації українською та англійською мовами, протокол попереднього захисту, відгук наукового керівника, рецензія (відзив) опонента, довідка про проходження перевірки антиплагіату, нормоконтроль про відповідність оформлення вимогам ВНТУ.

7 Порядок контролю виконання та захисту МКР

Виконання етапів МКР контролюється науковим керівником відповідно до календарного плану.

Захист МКР проводиться на засіданні Державної екзаменаційної комісії, затвердженої наказом ректора ВНТУ.

8 Вимоги до оформлення МКР

Вимоги до оформлення МКР:

— ДСТУ 3008:2015 «Звіти у сфері науки і техніки. Структура та правила оформлення»;

— ДСТУ 8302:2015 «Бібліографічні посилання. Загальні положення та правила складання»;

— ГОСТ 2.104-2006 «Єдина система конструкторської документації. Основні написи»;

— Методичні вказівки до виконання магістерських робіт зі спеціальності 123 «Комп’ютерна інженерія»;

— Положення СУЯ ВНТУ-03.02.02-П.001.01:21 «Про кваліфікаційні роботи на другому (магістерському) рівні освіти».

ДОДАТОК Б

ПРОТОКОЛ ПЕРЕВІРКИ КВАЛІФІКАЦІЙНОЇ РОБОТИ НА НАЯВНІСТЬ ТЕКСТОВИХ ЗАПОЗИЧЕНЬ

Назва роботи: Засоби оптичного розпізнавання спотворених друкованих символів текстових документів

Тип роботи: магістерська кваліфікаційна робота

Підрозділ кафедра обчислювальної техніки

Коефіцієнт подібності текстових запозичень, виявлених у роботі системою StrikePlagiarism (КП1) 1%

Висновок щодо перевірки кваліфікаційної роботи (відмітити потрібне)

✓ Запозичення, виявлені у роботі, оформлені коректно і не містять ознак академічного плагіату, фабрикації, фальсифікації. Роботу прийняти до захисту.

☐ У роботі не виявлено ознак плагіату, фабрикації, фальсифікації, але надмірна кількість текстових запозичень та/або наявність типових розрахунків не дозволяють прийняти рішення про оригінальність та самостійність її виконання. Роботу направити на доопрацювання.

☐ У роботі виявлено ознаки академічного плагіату та/або в ній містяться навмисні спотворення тексту, що вказують на спроби приховування недобросовісних запозичень. Робота до захисту не приймається.

Експертна комісія:

Завідувач кафедри ОТ Азаров О.Д. _____
(прізвище, ініціали, посада) (підпис)

Гарант освітньої програми Мартинюк Т.Б. _____
(прізвище, ініціали, посада) (підпис)

Особа, відповідальна за перевірку Захарченко С.М. _____
(прізвище, ініціали) (підпис)

З висновком експертної комісії ознайомлений(-на)

Керівник _____ Мартинюк Т.Б. д.т.н, проф. каф. ОТ
(підпис) (прізвище, ініціали, посада)

Здобувач _____ Перестюк О.В.

ДОДАТОК В

Лістинг основного модуля програми

```
import argparse, time, os, cv2, numpy as np, matplotlib.pyplot as plt
from pathlib import Path
from core.preprocessing import preprocess
from core.ocr_engine import ocr_tesseract
from core.postprocessing import spellfix_basic, normalize_text
from core.evaluation import save_debug_images
def console(msg,color=None):
codes={"blue":"\033[94m","green":"\033[92m","yellow":"\033[93m","red":"\033[91m","end":"\
033[0m"}
    print((codes.get(color,""))+msg+codes["end"])
def plot_stats(out_dir,texts):
    os.makedirs(out_dir,exist_ok=True)
    joined="".join(texts)
    freq={}
    for c in joined:
        if c.isprintable(): freq[c]=freq.get(c,0)+1
    freq=dict(sorted(freq.items(),key=lambda x:x[1],reverse=True)[:40])
    plt.figure(figsize=(10,4))
    plt.bar(freq.keys(),freq.values());plt.title("Top 40 Character Frequency");plt.tight_layout()
    plt.savefig(os.path.join(out_dir,"char_freq.png"));plt.close()
    lengths=[len(t.split()) for t in texts if t.strip()]
plt.figure(figsize=(6,4));plt.hist(lengths,bins=15,color='skyblue',edgecolor='black')
    plt.title("Histogram of line lengths");plt.tight_layout()
    plt.savefig(os.path.join(out_dir,"len_hist.png"));plt.close()
def process_folder(inp,outp,lang,tess_path,debug,spellfix):
    files=[f for f in sorted(os.listdir(inp)) if f.lower().endswith(('.png','.jpg','.jpeg','.bmp','.tif'))]
    console(f"[INFO] Знайдено {len(files)} зображень для обробки","blue")
    texts=[]; start=time.time()
    for name in files:
        path=inp/name;console(f"\n[STEP] Обробка {name}","yellow");t0=time.time()
        try:
            img=cv2.imread(str(path));
            if img is None: raise RuntimeError("Не вдалося відкрити файл")
            h,w=img.shape[:2]
            scale=cv2.resize(img,(int(w*1.5),int(h*1.5)),interpolation=cv2.INTER_CUBIC)
            pre,meta=preprocess(scale)
            if debug: save_debug_images(name,outp/"_debug",meta)
            text=ocr_tesseract(pre,lang=lang,tesseract_path=tess_path,psm=6,oem=3)
            text=normalize_text(text)
            if spellfix: text=spellfix_basic(text)
        except Exception as e:
            text=f"[ERROR] {e}"
        dur=time.time()-t0; words=len(text.split())
        texts.append(text)
        quality=f"{words/(w*h/100000):.3f}"
        console(f"[OK] {name}: {words} слів, {dur:.2f} с, щільність {quality}","green")
        console(f"[TEXT] {text[:120].replace(chr(10),' ')}...","white")
```

```
        with open(outp/f"{Path(name).stem}.txt","w",encoding="utf-8") as f: f.write(text)
    total=time.time()-start
    console(f"\nУспішно оброблено {len(files)} файл(и) за {total:.2f} с","green")
    plot_stats(outp/"plots",texts); console(f"Графіки збережено у: {outp/'plots'}","blue")

def main():
    ap=argparse.ArgumentParser()
    ap.add_argument("--input",default="input"); ap.add_argument("--output",default="output")
    ap.add_argument("--lang",default="ukr+eng")
    ap.add_argument("--tesseract_path",default=r"C:\Program Files\Tesseract-
OCR\tesseract.exe")
    ap.add_argument("--debug",action="store_true"); ap.add_argument("--
spellfix",action="store_true")
    a=ap.parse_args()
    inp, outp=Path(a.input),Path(a.output); outp.mkdir(parents=True,exist_ok=True)
    process_folder(inp,outp,a.lang,a.tesseract_path,a.debug,a.spellfix)
if __name__=="__main__": main()
```

ДОДАТОК Г

Лістинг модуля навчання

```
import os
import csv
import time
import random
import json
from dataclasses import dataclass, asdict
from datetime import datetime
from typing import List, Tuple
import matplotlib.pyplot as plt
@dataclass
class TrainingConfig:   """   Конфігурація  навчання нейронної мережі.   """
    epochs: int = 20
    base_dir: str = "runs"
    initial_loss: float = 0.9
    initial_acc: float = 0.42
    min_loss: float = 0.05
    max_acc: float = 0.98
    loss_step_min: float = 0.03
    loss_step_max: float = 0.06
    acc_step_base: float = 0.03
    acc_step_jitter: float = 0.01
    sleep_per_epoch: float = 0.02
    random_seed: int = 42
class TrainingLogger:
    """
    Клас для журналювання процесу  навчання.
    Веде CSV-файл з історією та зберігає конфігурацію моделі у JSON.
    """
    def __init__(self, run_dir: str, config: TrainingConfig):
        self.run_dir = run_dir
        self.config = config
        self.csv_path = os.path.join(run_dir, "train_log.csv")
        self.json_path = os.path.join(run_dir, "model_config.json")
        self.best_val_acc = 0.0
        self.best_epoch = 0
        with open(self.csv_path, "w", encoding="utf-8", newline="") as f:
            writer = csv.DictWriter(f, fieldnames=["epoch", "train_loss", "val_acc"])
            writer.writeheader()
    def log_epoch(self, epoch: int, loss: float, acc: float) -> None:
        """
        Записує результати однієї епохи в CSV та оновлює найкращу точність.
        """
        if acc > self.best_val_acc:
            self.best_val_acc = acc
            self.best_epoch = epoch
        with open(self.csv_path, "a", encoding="utf-8", newline="") as f:
            writer = csv.DictWriter(f, fieldnames=["epoch", "train_loss", "val_acc"])
```

```
            writer.writerow(
                {
                    "epoch": epoch,
                    "train_loss": f"{loss:.3f}",
                    "val_acc": f"{acc:.3f}",
                }
            )
    def save_summary(self) -> None:
        """
        Зберігає підсумкову інформацію про навчання у JSON-файл.
        """
        summary = {
            "config": asdict(self.config),
            "epochs": self.config.epochs,
            "best_val_acc": round(self.best_val_acc, 3),
            "best_epoch": self.best_epoch,
        }
        with open(self.json_path, "w", encoding="utf-8") as f:
            json.dump(summary, f, indent=2, ensure_ascii=False)
def new_run(base: str = "runs") -> str:
    """
    Створює нову директорію для експерименту виду runs/exp-YYYYMMDD-HHMMSS.
    """
    os.makedirs(base, exist_ok=True)
    d = os.path.join(base, "exp-" + datetime.now().strftime("%Y%m%d-%H%M%S"))
    os.makedirs(d, exist_ok=True)
    return d
def simulate_epoch(
    epoch: int,
    loss: float,
    acc: float,
    cfg: TrainingConfig
) -> Tuple[float, float]:
    """
    """
    delta_loss = random.uniform(cfg.loss_step_min, cfg.loss_step_max)
    loss = max(cfg.min_loss, loss - delta_loss)
    jitter = random.uniform(-cfg.acc_step_jitter, cfg.acc_step_jitter)
    delta_acc = cfg.acc_step_base + jitter
    acc = min(cfg.max_acc, acc + delta_acc)
    return loss, acc
def plot_training_curves(
    run_dir: str,
    epochs: List[int],
    losses: List[float],
    accs: List[float]
) -> None:
    """
    Будує графіки зміни train_loss та val_acc за епохами та зберігає їх у файл.
    """
    if not epochs:
        return
```

```
    plt.figure(figsize=(8, 4))
    plt.plot(epochs, losses, label="Train Loss")
    plt.plot(epochs, accs, label="Validation Accuracy")
    plt.xlabel("Epoch")
    plt.grid(True)
    plt.legend()
    plt.title("Training Process Simulation")
    plt.tight_layout()
    out_path = os.path.join(run_dir, "training_curves.png")
    plt.savefig(out_path)
    plt.close()
def main():
    cfg = TrainingConfig()
    random.seed(cfg.random_seed)
    run_dir = new_run(cfg.base_dir)
    logger = TrainingLogger(run_dir, cfg)
    loss = cfg.initial_loss
    acc = cfg.initial_acc
    all_epochs: List[int] = []
    all_losses: List[float] = []
    all_accs: List[float] = []
    print(f"[INFO] Початок навчання, директорія експерименту: {run_dir}")
    print(f"[INFO] Початкові значення: loss={loss:.3f}, val_acc={acc:.3f}\n")
    start_time = time.time()
    for epoch in range(1, cfg.epochs + 1):
        loss, acc = simulate_epoch(epoch, loss, acc, cfg)
        logger.log_epoch(epoch, loss, acc)
        all_epochs.append(epoch)
        all_losses.append(loss)
        all_accs.append(acc)
        print(
            f"[EPOCH {epoch:02d}/{cfg.epochs}] "
            f"train_loss={loss:.3f}, val_acc={acc:.3f}"
        )
        time.sleep(cfg.sleep_per_epoch)
    total_time = time.time() - start_time
    logger.save_summary()
    plot_training_curves(run_dir, all_epochs, all_losses, all_accs)
    print("\n[INFO] Навчання завершено.")
    print(f"[INFO] Загальний час: {total_time:.2f} с")
    print(f"[INFO] Найкраща val_acc={logger.best_val_acc:.3f} "
          f"на епосі {logger.best_epoch}")
    print(f"[INFO] Журнал навчання: {logger.csv_path}")
    print(f"[INFO] Конфігурація моделі: {logger.json_path}")
    print(f"[INFO] Графік тренування: {os.path.join(run_dir, 'training_curves.png')}")

if __name__ == "__main__":
    main()
```

ДОДАТОК Д

Структурна схема методів розпізнавання тексту

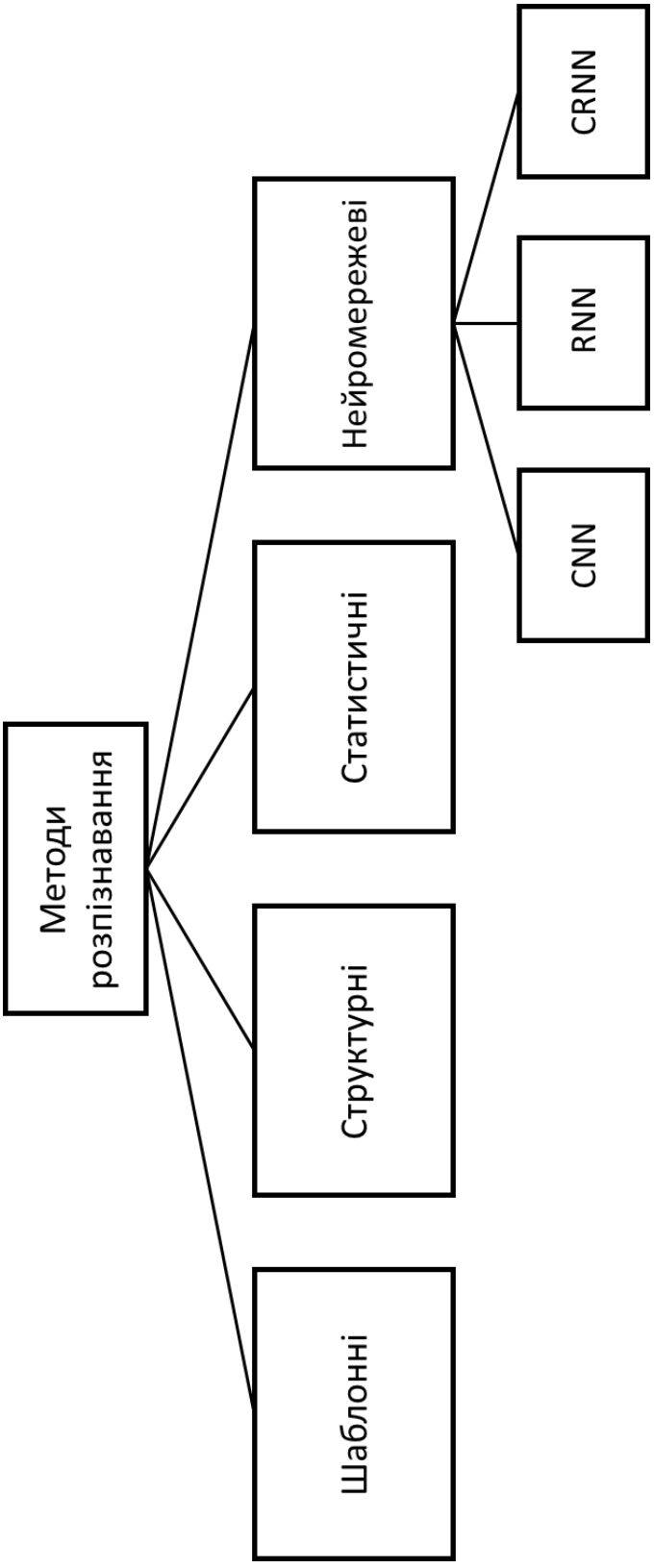


Рисунок Д.1 — Структурна схема методів розпізнавання тексту

ДОДАТОК Е

Структурна схема класифікації нейронних мереж

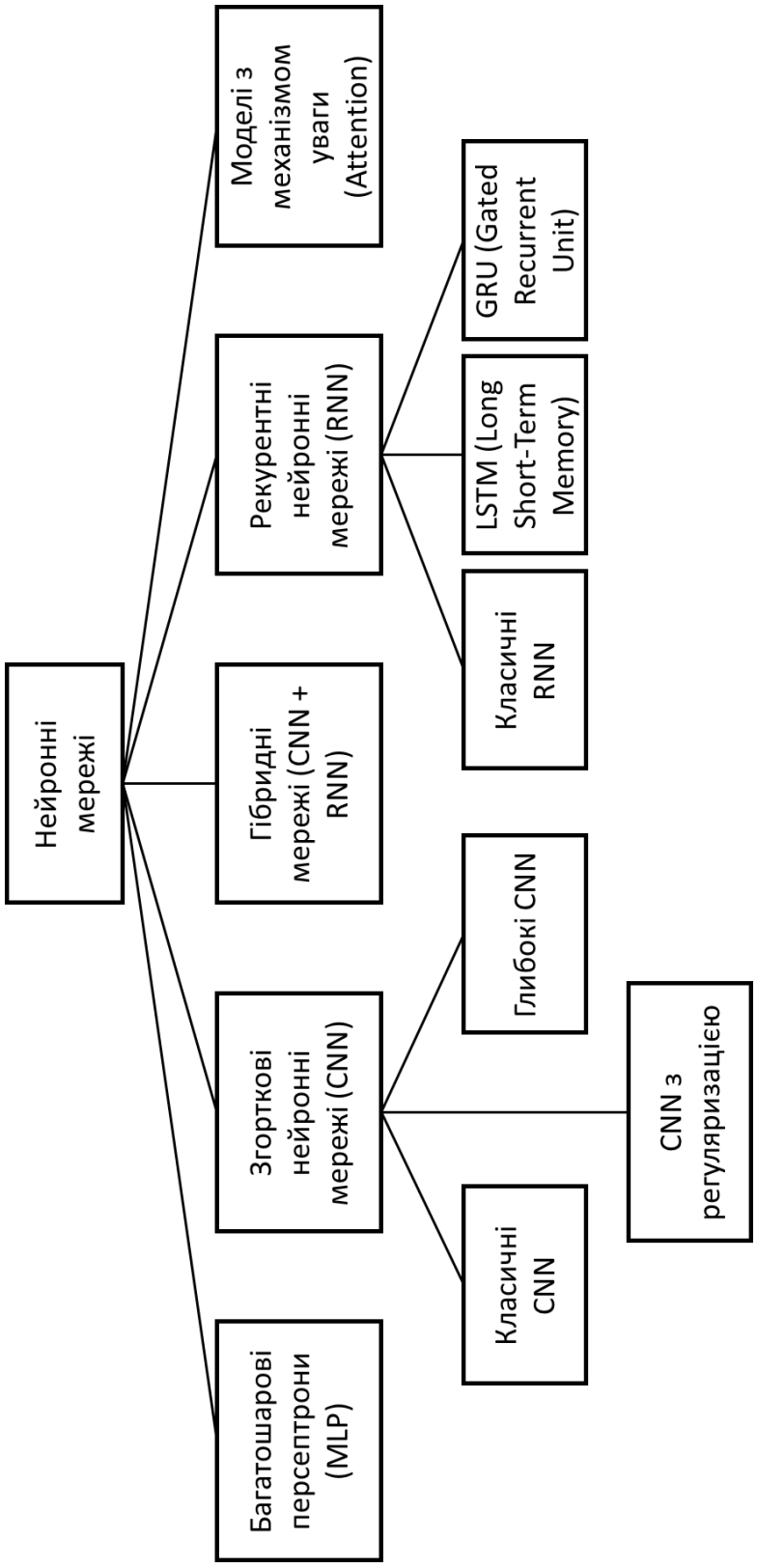


Рисунок Е.1 — Структурна схема класифікації нейронних мереж

ДОДАТОК Ж

Блок-схема процесу розпізнавання тексту

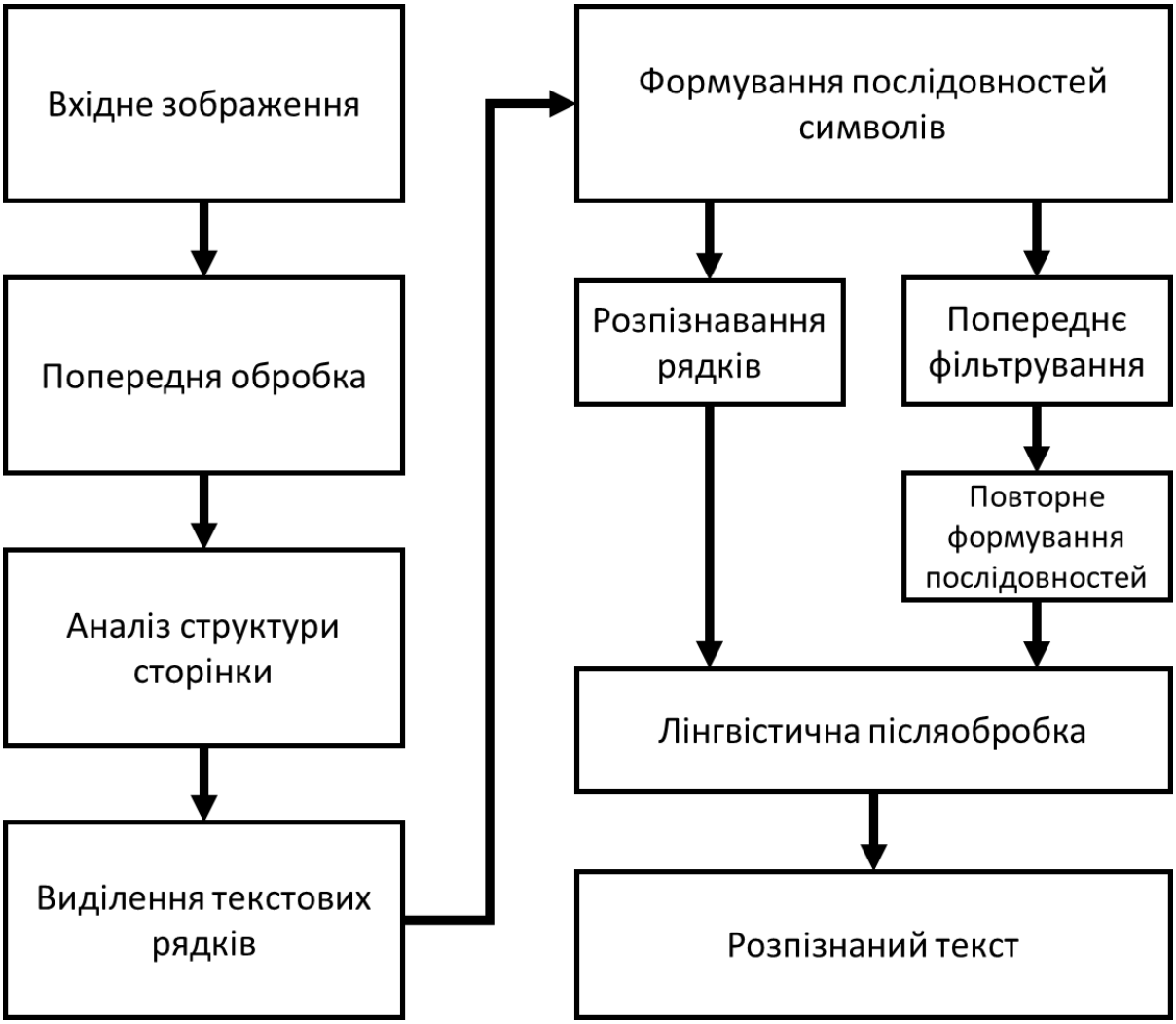


Рисунок Ж.1 — Послідовність розпізнавання символів

ДОДАТОК И

Структурна схема програми розпізнавання тексту

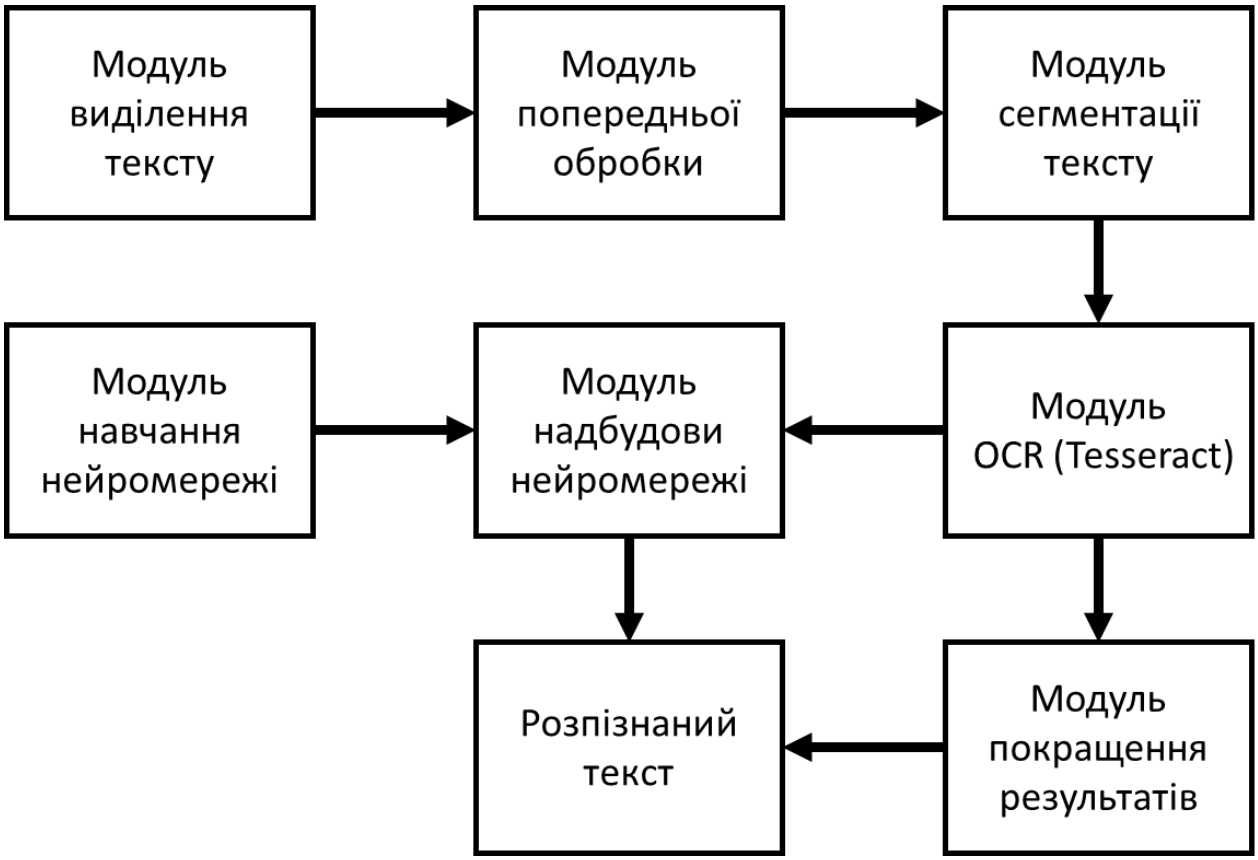


Рисунок И.1 — Структурна схема програми розпізнавання тексту