Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

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

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

ДОСЛІДЖЕННЯ МЕТОДІВ ОБРОБЛЕННЯ ТА РОЗПІЗНАВАННЯ РУКОПИСНОГО ТЕКСТУ
(тема)

Виконав:
студент 2 курсу, групи ІТКСм-20-1
Бєлінський Я.В.
(прізвище, ініціали)

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

Тип програми освітньо-професійна

Освітня програма Консолідована інформація
(повна назва освітньої програми)

Керівник доц. Творошенко І.С.
(посада, прізвище, ініціали)

Допускається до захисту

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

2021 р.

Харківський національний університет радіоелектроніки

Факультет Інформаційно-аналітичних технологій та менеджменту
(повна назва)

Кафедра Інформатики
(повна назва)

Рівень вищої освіти другий (магістерський)

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

Тип програми освітньо-професійна

Освітня програма Консолідована інформація
(повна назва освітньої програми)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«____»_____ 2021 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Бєлінському Ярославу Володимировичу
(прізвище, ім'я, по батькові)

1. Тема роботи Дослідження методів оброблення та розпізнавання рукописного тексту

затверджена наказом по університету від «_22_»_____10_____2021 року № 1575Ст.

2. Термін подання студентом роботи до екзаменаційної комісії _22_____11_____ 2021 р.

3. Вихідні дані до роботи науково-методична та науково-технічна література, матеріали конференцій, дані інтернет-мережі, бібліотека машинного навчання з відкритим кодом TensorFlow, мова програмування Python, середовище розроблення PyCharm.

4. Перелік питань, що потрібно опрацювати в роботі

1. Прокласифікувати існуючі методи оброблення та розпізнавання рукописного тексту.
2. Розробити модель оброблення та розпізнавання рукописного тексту за допомогою HMM.
3. Розробити модель оброблення та розпізнавання рукописного тексту за допомогою нейронних мереж.
4. Програмно реалізувати розроблені моделі оброблення та розпізнавання рукописного тексту у окремих застосунках.
5. Дослідити та протестувати розроблені застосунки.

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри) Актуальність дослідження, об'єкт та мета дослідження, постановка задачі дослідження, вихідні дані дослідження, етапи програмної реалізації вибраних методів, результати дослідження, перспективи подальших досліджень, апробація результатів дослідження.

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

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата
Консультант з дотримання діючих стандартів та норм	Доцент Бєлова Н.В.		

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Отримання завдання на кваліфікаційну роботу	22.10.2021	
2	Аналіз завдання, підбір літератури	23.10.21-26.10.21	
3	Аналіз літератури з досліджуваної проблеми	27.10.21-30.10.21	
4	Аналіз технічних засобів	30.10.21-02.11.21	
5	Розробка методів	02.11.21-05.11.21	
6	Програмна реалізація та тестування	06.11.21-11.11.21	
7	Оформлення пояснювальної записки	12.11.21-20.11.21	
8	Перевірка на плагіат	28.11.2021	
9	Рецензування	28.11.2021	
10	Підготовка презентації та доповіді	29.11.2021	
11	Занесення роботи в електронний архів	30.11.2021	
12	Попередній захист кваліфікаційної роботи	01.12.2021	

Дата видачі завдання 22 жовтня 2021 р.

Студент _____
(підпис)

Керівник роботи _____ доц. Творошенко І.С.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ/ABSTRACT

Пояснювальна записка до кваліфікаційної роботи: 72 с., 2 табл., 39 рис., 1 дод., 48 джерел.

ОПТИЧНЕ РОЗПІЗНАВАННЯ СИМВОЛІВ, ШТУЧНІ НЕЙРОННІ МЕРЕЖІ, ШАБЛОННИЙ МЕТОД, СТРУКТУРНИЙ МЕТОД, ОЗНАКОВИЙ МЕТОД, ПРИХОВАНА МОДЕЛЬ МАРКОВА, РОЗПІЗНАВАННЯ РУКОПИСНОГО ТЕКСТУ.

Об’єктом дослідження є зразки рукописних текстів.

Метою дослідження є програмна реалізація досліджених методів оброблення та розпізнавання рукописних текстів, а також порівняння їх точності розпізнавання на базі створеного набору зразків.

Розглянуто основні етапи оброблення зображення: перетворення зображення в градації сірого, видалення дефектів зображення і відділення тексту від фону. Застосовано такі фільтри для видалення дефектів зображення: фільтр Гауса (використовується для придушення високочастотного шуму), медіанний фільтр (використовується для придушення шуму «сіль і перець») і фільтр на основі вейвлет-перетворення.

У результаті роботи здійснено програмну реалізацію застосунків для оброблення та розпізнавання рукописного тексту.

OPTICAL CHARACTER READER, ARTIFICIAL NEURAL NETWORKS, TEMPLATE METHOD, STRUCTURAL METHOD, INDICCATIVE METHOD, HIDDEN MARKOV MODEL, HANDWRITTEN TEXT RECOGNITION.

The object of study are samples of manuscripts.

The purpose of the research is the software implementation of the studied methods of processing and recognition of handwritten texts, as well as comparing their recognition accuracy based on the created set of samples.

The main stages of image processing are considered: converting the image to grayscale, removing image defects and separating the text from the background. Various filters for removing image defects are presented: a Gaussian filter (used to suppress high-frequency noise), a median filter (used to suppress salt and pepper noise), and a wavelet transform filter.

As a result, the software implementation of applications for processing and recognition of handwritten text was implemented.

ЗМІСТ

Перелік умовних позначень, символів, одиниць, скорочень і термінів 7
Вступ 8
1 Аналіз існуючих методів оброблення та розпізнавання рукописного тексту 10
1.1 Класифікація існуючих методів оброблення рукописного тексту 10
1.2 Класифікація існуючих методів розпізнавання рукописного тексту 16
1.3 Аналіз сучасних систем розпізнавання рукописного тексту 20
1.4 Аналіз літературних джерел щодо апробації результатів оброблення та розпізнавання рукописного тексту 25
1.5 Постановка задачі дослідження 27
2 Особливості вибраних методів оброблення та розпізнавання рукописного тексту 29
2.1 Метод виділення ключових ділянок на зображенні 29
2.2 Метод сегментації на окремі складові текстових зон 33
2.3 Метод векторного розпізнавання рукописних символів 37
2.3.1 Побудова векторної моделі…. 37
2.3.2 Перетворення векторної моделі до інваріантного виду 42
2.3.3 Розпізнавання рукописних символів на основі інваріантної векторної моделі 44
2.4 Розпізнавання рукописного тексту на основі ймовірнісного підходу 46
2.4.1 Розробка лінгвістичної моделі слова…. 46
2.4.2 Адаптивна побудова дерева рішень на основі імовірнісного підходу 47
3 Реалізація та дослідження вибраних методів оброблення та розпізнавання рукописного тексту 51

3.1 Вибір інструментальних засобів для реалізації вибраних методів оброблення та розпізнавання рукописного тексту 51
3.2 Етапи програмної реалізації вибраних методів оброблення та розпізнавання рукописного тексту 53
3.2.1 Застосунок, що використовує нейронну мережу, як спосіб розпізнавання рукописного тексту….. 53
3.2.1.1 Модуль організації інтерфейсу з користувачем…… 53
3.2.1.2 Модуль сегментації…… 53
3.2.1.3 Модуль створення моделі для задачі розпізнавання рукописного тексту….. 54
3.2.1.4 Модуль розпізнавання та прийняття рішень….. 55
3.2.2 Застосунок, що використовує HMM, як спосіб розпізнавання рукописного тексту ….. 56
3.2.2.1 Модуль організації інтерфейсу з користувачем…… 56
3.2.2.2 Модуль сегментації…… 56
3.2.2.3 Модуль розпізнавання та прийняття рішень….. 57
3.3 Тестування розробленого застосунку та аналіз результатів 59
3.4 Перспективи подальшої роботи 62
Висновки 63
Перелік джерел посилання 65
Додаток А Приклади зображення слів для навчання нейронної мережі 71

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

КПК – кишеньковий персональний комп’ютер

HMM – Hidden Markov Model (прихована модель Маркова)

OCR – Optical Character Recognition (оптичне розпізнавання символів)

PDF – Portable Document Format (портативний формат документів)

CNN – Convolutional Neural Network (згорткова нейронна мережа)

RNN – Recurrent Neural Network (рекурентна нейронна мережа)

CTC – Connectionist Temporal Classification (тимчасова класифікація коннекціоністів)

CLI – Command Line Interface (інтерфейс командного рядка)

GUI – Graphical User Interface (графічний інтерфейс користувача)

HTR – Handwritten Text Recognition (розпізнавання рукописного тексту)

ВСТУП

Для банків, промислових і торгових підприємств все більш актуальною стає проблема неефективності паперового діловодства. Для переходу до автоматизованого електронного документообігу за уніфікованими і стандартизованим процесам, побудованих на безпаперових технологіях, необхідно вирішити задачу розпізнавання рукописної текстової інформації для перекладу друкованих та рукописних документів в електронну форму.

Вхідною інформацією в системах електронного документообігу можуть бути не тільки документи з друкованим текстом, але і рукописні документи (документація паспортно-візової служби, анкетування, прийом заяв від населення). Також є велика кількість успадкованих рукописних документів, що містять важливу технічну інформацію, яку бажано було б перевести в електронний вигляд.

Розпізнавання рукописного тексту на зображеннях є важливим завданням машинного навчання, так як це дозволяє організувати зручну взаємодію з даними: редагування, аналіз, пошук слів чи фраз.

В останні десятиліття, завдяки використанню сучасних досягнень комп’ютерних технологій, були розвинені нові методи оброблення зображень і розпізнавання образів, завдяки чому стало можливим створення таких промислових систем розпізнавання друкованого тексту.

Проте, створення застосунка в даній області, як і раніше залишається творчим завданням і вимагає додаткових досліджень в зв’язку зі специфічними вимогами з дозволу, швидкодії, надійності розпізнавання і обсягом пам’яті, якими характеризується кожна конкретна задача.

В останні роки розвиток методів оброблення зображень і розпізнавання образів для окремих прикладних задач дозволив створити системи оптичного розпізнавання символів (OCR-системи). Так, наприклад, програма FineReader дозволяє розпізнати друкований текст з малою кількістю помилок, які потребують людського контролю [1].

Проте, створення кожного нового застосунка в даній області, як і раніше, залишається творчим завданням і вимагає додаткових досліджень в зв’язку зі специфічними вимогами до дозволу, швидкодії, надійності розпізнавання і обсягу пам’яті, якими характеризується кожна конкретна задача.

Первинним перекладом документа в електронну форму є його сканування або фотографування, в результаті якого виходить графічний файл у вигляді фотографії або сканованого документу. Однак, такі файли, особливо з високою роздільною здатністю, займають багато місця на диску, і текст в них неможливо редагувати. У зв’язку з цим, доцільно видобувати текст з графічних файлів, що успішно робиться із застосуванням OCR.

Відомі два основних підходи до розпізнавання рукописного тексту:

– розпізнавання в режимі поточного введення символів (інтерактивний режим);

– розпізнавання раніше написаних документів (пасивний режим) [2].

Перший підхід використовується в системах реального часу, до яких відносяться системи сенсорного введення рукописних символів в КПК, комунікаторах і інших пристроях. Існує багато методів, які вирішують це завдання досить ефективно. Дані системи використовуються при введенні інформації з паперових носіїв. Вхідними даними другого підходу є зображення, отримані зі сканера або інших цифрових пристроїв.

Таким чином, завдання оброблення і розпізнавання рукописного тексту є актуальним в різних сферах діяльності, а розроблення методів розпізнавання рукописного тексту дозволить підвищити ефективність роботи таких систем.

1 АНАЛІЗ ІСНУЮЧИХ МЕТОДІВ ОБРОБЛЕННЯ ТА РОЗПІЗНАВАННЯ РУКОПИСНОГО ТЕКСТУ

1.1 Класифікація існуючих методів оброблення рукописного тексту

Розпізнавання рукописного тексту, зазвичай, складається з наступних етапів: попереднє оброблення зображення, сегментація і нормалізація, побудова ознак, класифікація та оброблення результатів.

На етапі попереднього оброблення зображення використовуються такі методи, як фільтрація, шумопоглинання та інші, їх метою є підвищити якість зображення (рис. 1.1).

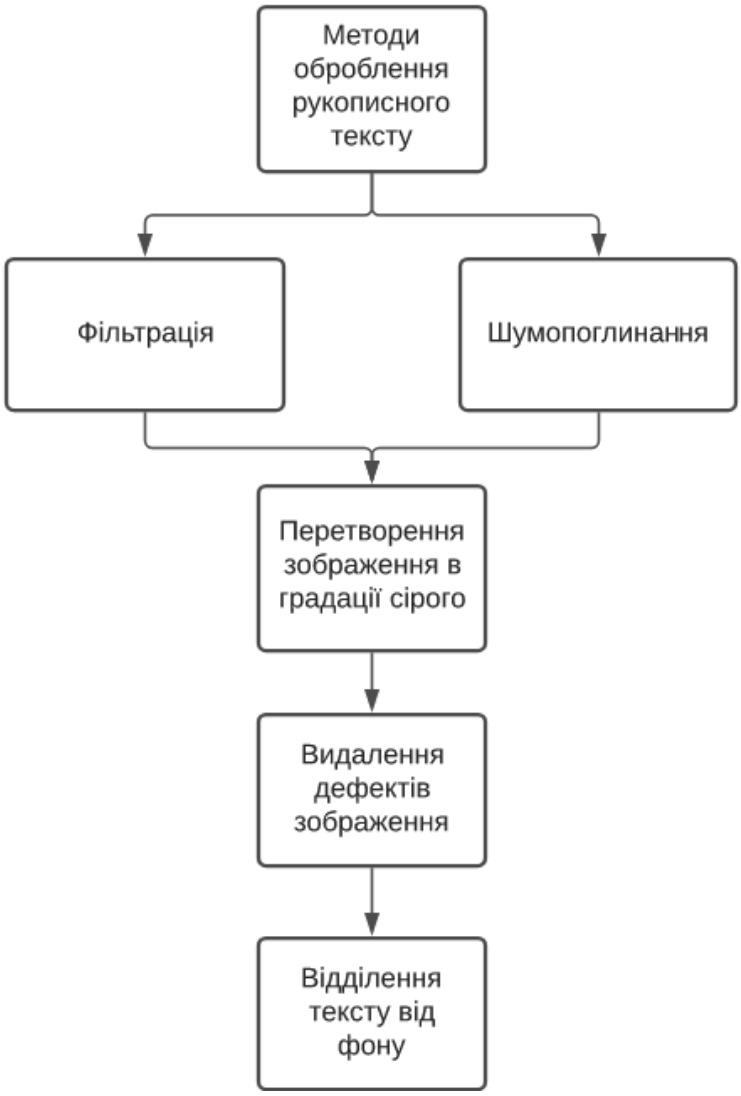


Рисунок 1.1 – Схема класифікації методів оброблення рукописного тексту

На сьогоднішній день завдання розпізнавання рукописного тексту є невирішеним. Завдання підвищення якості зображення при розпізнаванні рукописного тексту, є надзвичайно актуальним, оскільки чим краще буде якість зображення, тим зручніше буде працювати з ним на наступних етапах оброблення.

Поліпшення якості зображення включає, зазвичай, в себе перетворення зображення в градації сірого, видалення дефектів зображення і відділення тексту від фону.

Зазвичай для спрощення подальшої роботи з зображенням робиться перетворення зображення в градації сірого. Для кожного окремого пікселя обчислюється значення яскравості, яке вимірюється в діапазоні від 0 до 255. Чорний колір відповідає 0 рівню яскравості, а білий – 255 рівню яскравості. Це обчислення проводиться за допомогою такої формули [3]:

$$I = 0{,}299R + 0{,}587G + 0{,}114B,$$

де $R,\ G,\ B$ – значення червоного, зеленого і синього каналів відповідно.

Видалення дефектів здійснюється стандартними методами оброблення зображень. Найбільш часто для видалення шуму використовують фільтр Гауса для придушення високочастотного шуму і медіанний фільтр для видалення шуму «сіль і перець» (рис. 1.2) [4]. Перспективним фільтром є фільтр на основі вейвлет-перетворення.

Фільтр Гауса заснований на функціях Гауса з однією і двома змінними відповідно [3]:

$$G(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{x^2}{2\sigma^2}},$$

$$G(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}},$$

де σ – «ступінь розмиття» обробленого зображення (стандартне відхилення нормального розподілу);

x, y – відстань між пікселем (вихідною точкою) і точкою, для якої підраховується значення функції Гауса по вертикальній і горизонтальної осі відповідно.

Рисунок 1.2 – Приклади поширених типів шумів:
а) шум «сіль і перець»; б) високочастотний шум

Отже, за допомогою функції Гауса можна побудувати матрицю згортки, яка допомагає для кожного пікселя зображення розрахувати середньозважене значення сусідніх пікселів [3]:

$$r_{mn} = \sum_{i=-k}^{k} \sum_{j=-k}^{k} r_{m+i,\ n+j} G(i,j),$$

де k – розмірність матриці згортки.

Таким чином, за допомогою застосування фільтра Гауса шум буде пригнічений, оскільки всі зашумлені пікселі (яскравість яких сильно відрізняється від яскравості сусідніх пікселів) візьмуть усереднене значення, у

результаті чого контури об’єктів будуть підкреслені, що позитивно впливає на результат розпізнавання образів (в тому числі і рукописного тексту) на цифрових зображеннях.

В основі медіанного фільтру лежить поняття медіани.

Якщо множина A, що складається з чисел A_i, де $i = 1..n$, відсортовано за зростанням, то $A_{n/2}$ – є медіаною цієї множини. Медіана буде ділити відсортований набір чисел на дві частини, перша частина буде містити числа, які менші за медіану, а друга частина – більші.

Найпоширеніший спосіб реалізації даного фільтру полягає в тому, щоб упорядкувати значення яскравості пікселів за допомогою вікна з непарним радіусом, а потім замінити значення яскравості пікселів на значення медіани множини.

Фільтр на основі вейвлет-перетворення. Оскільки зображення можна представити у вигляді дискретного сигналу, то для його оброблення можна використовувати фільтри, що базуються на частотному поділі в дискретній області. Вейвлет-аналіз є досить перспективним способом аналізу даних. Сигнал можна представити таким чином [3]:

$$s(t) = f(t) + \sigma e(t),$$

де $f(t)$ – корисний сигнал;

$e(t)$ – шум;

σ – рівень шуму;

$s(t)$ – досліджуваний сигнал.

Таким чином, вейвлет-перетворення дозволяє видалити шум за 4 кроки:

– розкладання сигналу по базису вейвлетів;

– вибір порогового значення шуму для кожного з рівнів розкладання;

– порогова фільтрація коефіцієнтів деталізації;

– відновлення сингалу.

Такий спосіб фільтрації найкраще працює на гладких сигналах, тобто на таких сигналах, в розкладанні яких лише невелика кількість коефіцієнтів деталізації значно відрізняється від нуля. Підбір вейвлета і глибини розкладання залежить від властивостей об'єкта, що фільтрується сигналом. Для вибору порогу шуму застосовують критерії, які мінімізують квадратичну функцію втрат для обраної моделі шуму. Дані фільтри рідше використовуються у порівнянні з медіаною, тому що вейвлети призводять до додаткової параметризації програми і уповільнення роботи, оскільки потрібно обчислити додаткові масиви даних.

Відділення тексту від фону є окремим випадком завдання виділення об'єкта на зображенні. Завдання полягає в тому, щоб по зображенню тексту A отримати бінарне зображення B, таке, що

$$B(i,j) = \begin{cases} 1, P(i,j) \in T_A, \\ 0, P(i,j) \notin T_A, \end{cases}$$

де $P(i,j)$ – піксель (i,j);

T_A – текст на зображенні A [3].

Дане перетворення дозволяє в подальшому використовувати аналіз зв'язкових компонент, контурів, скелетів і т.д. Найбільш часто використовується методом відділення тексту від фону порогова бінаризація.

Нехай дано зображення, $I(i,j)$ – яскравість пікселя з координатами (i,j). Порогова бінаризація зображення називається попіксельним перетворення $f(i,j)$ таке, що

$$f(i,j) = \begin{cases} 1, I(i,j) \geq d, \\ 0, I(i,j) < d, \end{cases}$$

де d називається порогом бінаризації [3].

Зазвичай на гістограмі яскравості зображення тексту спостерігається два піки: високий пік в області світлих пікселів, який відповідає фону, і нижчий пік в області темних пікселів, який відповідає тексту. Таким чином, завдання пошуку порогового значення яскравості, тобто такого, щоб пікселі з яскравістю вище цього значення (фон) будуть вважатися чорними, а нижче (текст) – білими (таке «інвертування» кольору робиться з метою спрощення застосування багатьох алгоритмів в подальшому), є завданням знаходження оптимального значення між двома піками гістограми. Для вирішення цього завдання існує метод Оцу і його варіації.

У методі Оцу діапазон яскравості $[0; L]$ зображення ділиться на дві частини граничним значенням T. Суть алгоритму полягає в тому, щоб мінімізувати внутрішньокласову дисперсію, яка визначається як зважена сума дисперсій двох класів. В алгоритмі Оцу мінімізація внутрішньокласової дисперсії еквівалентна максимізації міжкласової дисперсії, яка розраховується наступним чином [3]:

$$\sigma_b^2 = \omega_1 \omega_2 (\mu_1 - \mu_2)^2,$$

де σ_b – міжкласова дисперсія;

ω_1 і ω_2 – ймовірності першого і другого класів;

μ_1і μ_2 – середнє арифметичне значення кожного з класів.

Недоліком даного методу є чутливість до нерівномірної освітленості. Для вирішення даної проблеми, зазвичай, отримують компонент освітлення шляхом низькочастотної фільтрації G зображення за допомогою фільтра Гауса.

Ще одним недоліком методу Оцу є злипання близько розташованих областей, що може вплинути на подальше оброблення і розпізнавання зображення. Тому існує метод адаптивної бінаризації, який до того ж дозволяє вирішити проблему різниці освітленості.

Отже, розглянуто основні методи оброблення зображень під час розпізнавання рукописного тексту. Встановлено, що слід детально вибирати методи, оскільки при правильному виборі вони є гарантом успішного розпізнавання тексту.

1.2 Класифікація існуючих методів розпізнавання рукописного тексту

Широко досліджуваною проблемою в даний час є розпізнавання рукописного тексту. На даний момент досягнута точність розпізнавання навіть гірша, ніж для надрукованого тексту.

Кращі показники можуть бути досягнуті тільки з використанням контекстної та граматичної інформації. Наприклад, у процесі розпізнання шукати цілі слова в словнику легше, ніж намагатися проаналізувати окремі символи з тексту. Знання граматики мови може також допомогти визначити, чи є слово дієсловом або іменником. Форми окремих рукописних символів іноді можуть не містити достатньо інформації, щоб точно (близько 100%) розпізнати весь рукописний текст [5].

Методи автоматичного розпізнавання образів та їх реалізація в системах оптичного читання текстів (OCR-системах) – одна з найбільш плідних технологій штучного інтелекту.

Під OCR розуміється автоматичне розпізнавання зображень символів друкованого або рукописного тексту за допомогою спеціальних програм, наприклад, введення в комп’ютер за допомогою сканера, і перетворення його в формат, придатний до оброблення текстовими процесорами, редакторами текстів і т.д. Іноді під OCR розуміють пристрій оптичного розпізнавання символів або автоматичного читання тексту. У даний час такі пристрої при промисловому використанні обробляють до 100 000 документів на добу. Передбачено введення документів хорошої і середньої якості – це, наприклад, бланки перепису населення, податкових декларацій і т.п.

Системи розпізнавання реалізуються у вигляді класифікаторів, які використовують різні методи, наприклад, шаблонні (растрові), ознакові, структурні і т.д. (рис. 1.3) [6].

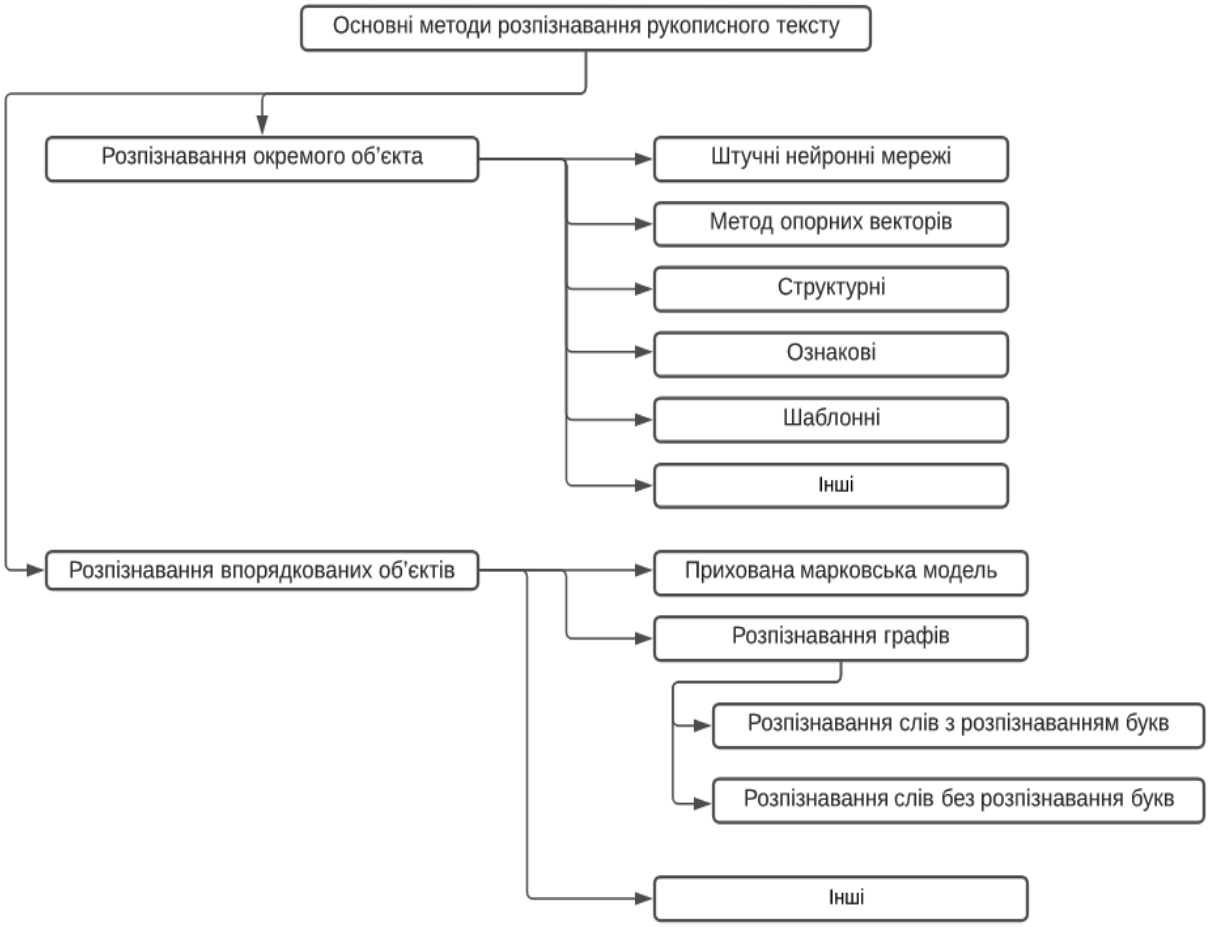


Рисунок 1.3 – Основні методи розпізнання рукописного тексту

У класифікаторі шаблонного типу за допомогою критерію порівняння визначається, який з шаблонів вибрати з бази. Критерій – мінімум точок, що відрізняють шаблон від досліджуваного зображення. До переваг шаблонного класифікатора відносяться гарне розпізнавання дефектних символів («розірваних» або «склеєних»), простота і висока швидкість розпізнавання. Недоліком є необхідність налаштування системи на типи [7] і розміри шрифтів.

Ознакові методи найбільш поширені. Аналіз в них базується на тому, що зображенню ставиться у відповідність N-мірний вектор ознак.

Розпізнавання полягає в порівнянні його з набором еталонних векторів тієї ж розмірності. Якість розпізнавання залежить від типів ознак і їх кількостей. Формування вектора проводиться під час аналізу підготовленого зображення, еталон для кожного класу отримують аналогічним способом оброблення символів з навчальної вибірки. Основними перевагами даної групи є простота реалізації, хороша узагальнююча здатність, стійкість до змін форми символів та швидкодія. Із недоліків зазначених методів варто виділити нестійкість до дефектів зображення, втрата частини інформації про символ на етапі отримання ознак, відсутність чітко сформульованих правил щодо формування ознак [8].

Структурні методи розпізнавання використовують інформацію не про написання символу, а про його топологію – взаємне розташування окремих складових частин символу. До переваг даних методів можна віднести незалежність щодо типів і розмірів шрифтів. Головною проблемою топологічних методів є ідентифікація знаків, що мають дефекти (наприклад, розрив лінії або злиття сусідніх ліній), а також невисока швидкодія.

Таким чином, під час даного дослідження розглянемо три основні підходи для вирішення задачі розпізнавання: шаблонний, ознаковий і структурний (табл. 1.1).

Таблиця 1.1 – Методи розпізнавання рукописного тексту

Назва методів	**Основні переваги**	**Основні недоліки**
Шаблонний	Висока точність розпізнавання дефектних символів	Залежність від шрифту, який зустрічається в зображенні
Ознаковий	Простота реалізації, хороша узагальнююча здатність	Нестійкість до дефектів зображення
Структурний	Незалежність щодо типів і розмірів шрифтів	Ідентифікація знаків, що мають дефекти

При використанні шаблонних методів розпізнавання символів відскановане зображення переводиться в растрове (поточне), потім проводиться його порівняння з еталонними шаблонами, які були сформовані в базі даних. Критерієм вибору шаблону є найменша кількість точок, відмінних від досліджуваного зображення. Шаблон для кожного класу отримують шляхом усереднення зображення символів навчальної вибірки. Перевагою шаблонних методів є висока точність розпізнавання дефектних символів. Основним недоліком є залежність від шрифту, який зустрічається в зображенні. Шрифт повинен бути відомий заздалегідь, інакше буде неможливо правильно розпізнати досліджуваний символ.

Наряду з іншими методами, актуальними стали методи, які не потребують попередньої сегментації, наприклад, ієрархічні приховані моделі Маркова і згорткові нейронні мережі. У зв’язку з виникненням нової хвилі популярності нейромережевих класифікаторів вони стали частіше використовуватися в дослідженнях з розпізнавання тексту. Основною перевагою використання нейромережевих технологій є хороша узагальнююча здатність, можливість використовувати контекстний аналіз і розпізнавати символи, ґрунтуючись на оточуючі символи [8].

Найпростіший алгоритм розпізнавання полягає в сегментації, розпізнаванні кожної частини і у видачі як загальної відповіді послідовності найбільш ймовірних відповідей. Однак, даний спосіб працює не ефективно. Він не виправляє помилок сегментації і адекватний тільки у розпізнаванні випадкової послідовності незалежних добре розділених об’єктів. У всіх задачах розпізнавання послідовності не випадкові, а в разі рукописного тексту – ще й погано розділені.

Для вирішення даного завдання використовуються різні стохастичні граматики і відповідні імовірнісні методи, серед яких можна вказати такі:

– прихована марківська модель (HMM): з дискретним часом, кінцевим простором станів і кінцевим простором спостережуваних станів;

– розпізнавання графів: розпізнавання слів з розпізнавачем букв і розпізнавання слів без розпізнавача букв [9].

1.3 Аналіз сучасних систем розпізнавання рукописного тексту

OCR – це новітній метод механічного перекладу, який перетворює зображення рукописного тексту в редагований текст на комп'ютері [10]. Наприклад, він може зробити звичайний PDF з відсканованого файлу за допомогою OCR або PDF на основі зображення, або перетворює рукописний текст на друкований. Технологію було розроблено ще в 1933 році, але вона постійно розвивається та удосконалюється [11]. Інструменти OCR здатні виконувати громіздку роботу щодо перетворення газет, листів, книг і будь-яких інших друкованих або рукописних матеріалів в комп'ютерні редаговані тексти.

Технологія розпізнавання OCR рукописних текстів використовується для великих баз даних, при цьому рівень точності транскрипції зростає з кожним днем, і вона вже близька до ідеалу (100%).

PDFelement Pro – інструмент для OCR-розпізнавання PDF-файлів [12]. Він може автоматично розпізнати відскановані файли PDF і відредагувати їх за допомогою вбудованих інструментів редагування (рис. 1.4). Крім цього, він підтримує кілька мов OCR. Є можливість легко редагувати PDF-тексти, зображення, посилання та інші елементи, конвертувати PDF-файли в інші формати.

Основні функції даної PDF OCR-програми [12]:

– розширена функція OCR дозволяє легко конвертувати і редагувати відскановані PDF-файли;

– редагувати тексти PDF, зображення і посилання – це так просто, як і внесення змін в Word;

– додавати підпис, пароль, водяні знаки та знаки, що намальовані від руки, в PDF-файли;

– розміщувати коментарі та зауваження, де необхідно;

– створювати PDF-файл з інших форматів;

– є можливість конвертувати PDF в такі формати, як Excel, MS Word.

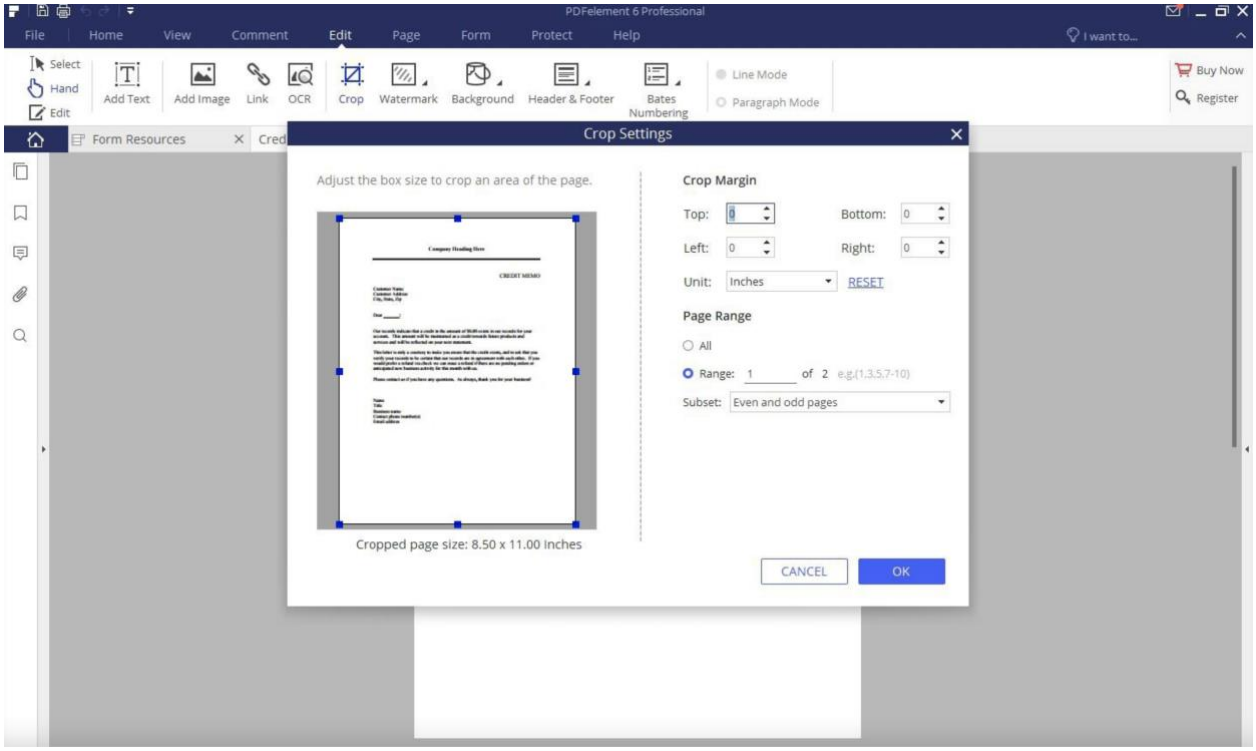


Рисунок 1.4 – PDFelement Pro

OCR Desktop – це OCR-застосунок для настільного комп’ютера, що використовує штучний інтелект і нейронні мережі для поліпшення якості роботи (рис. 1.5) [13].

Конвертер перетворення PDF-файлу в текст навчили більше, ніж чотирьом мільйонам варіантів шрифтів, перетворений текст є точним наскільки це взагалі можливо.

Він також володіє новітньою технологією OCR для вирішення будь-якої задачі в розпізнаванні почерку.

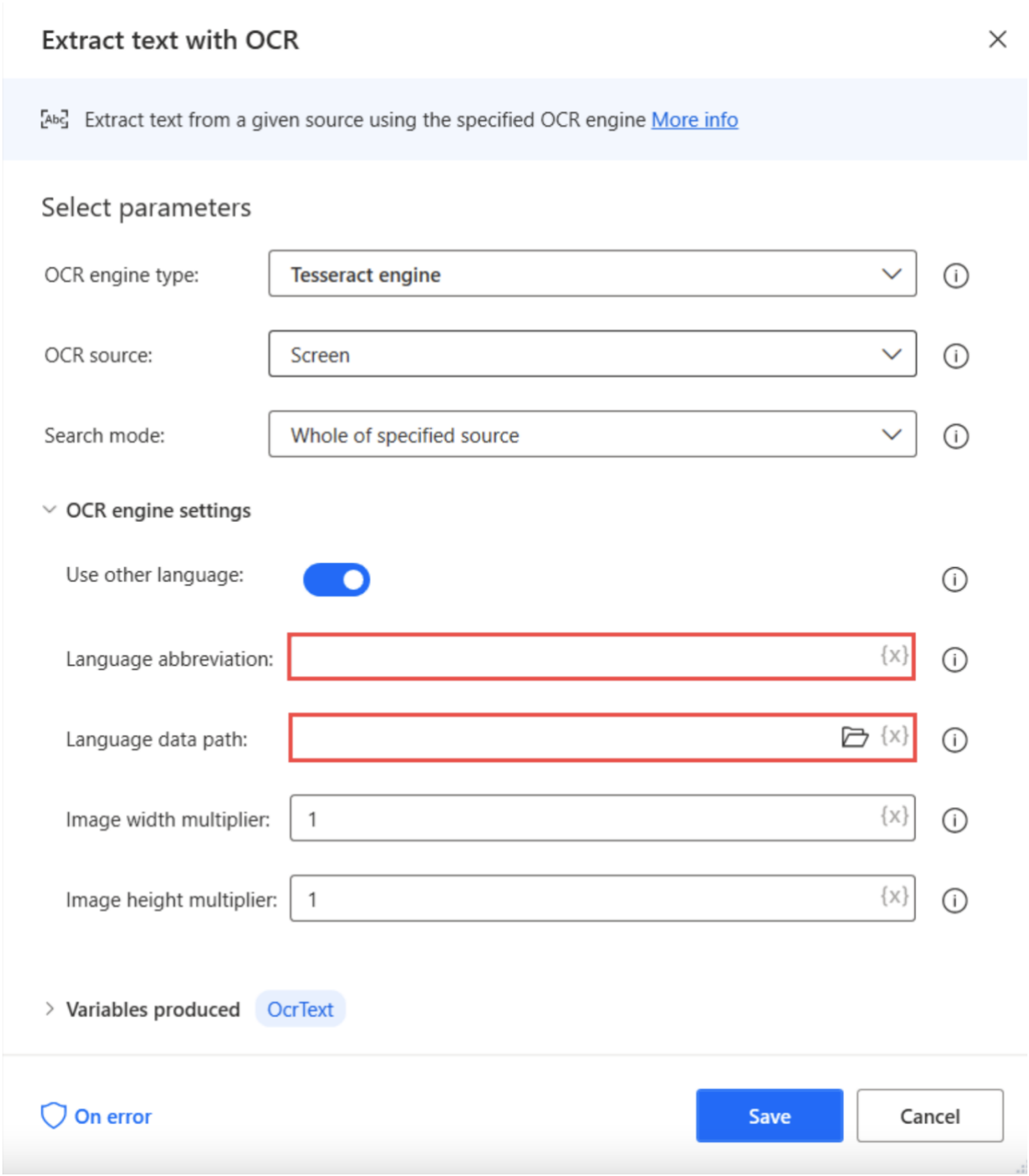


Рисунок 1.5 – OCR Desktop

SimpleOCR – одна з найпопулярніших безкоштовних програм OCR доступних в мережі (рис. 1.6) [14]. Вона досить проста, але в її арсеналі є всі основні функції сканування і конвертації, які важливі при роботі з OCR-розпізнаванням рукописних текстів. Однак, безкоштовна версія має обмежений функціонал для використання.

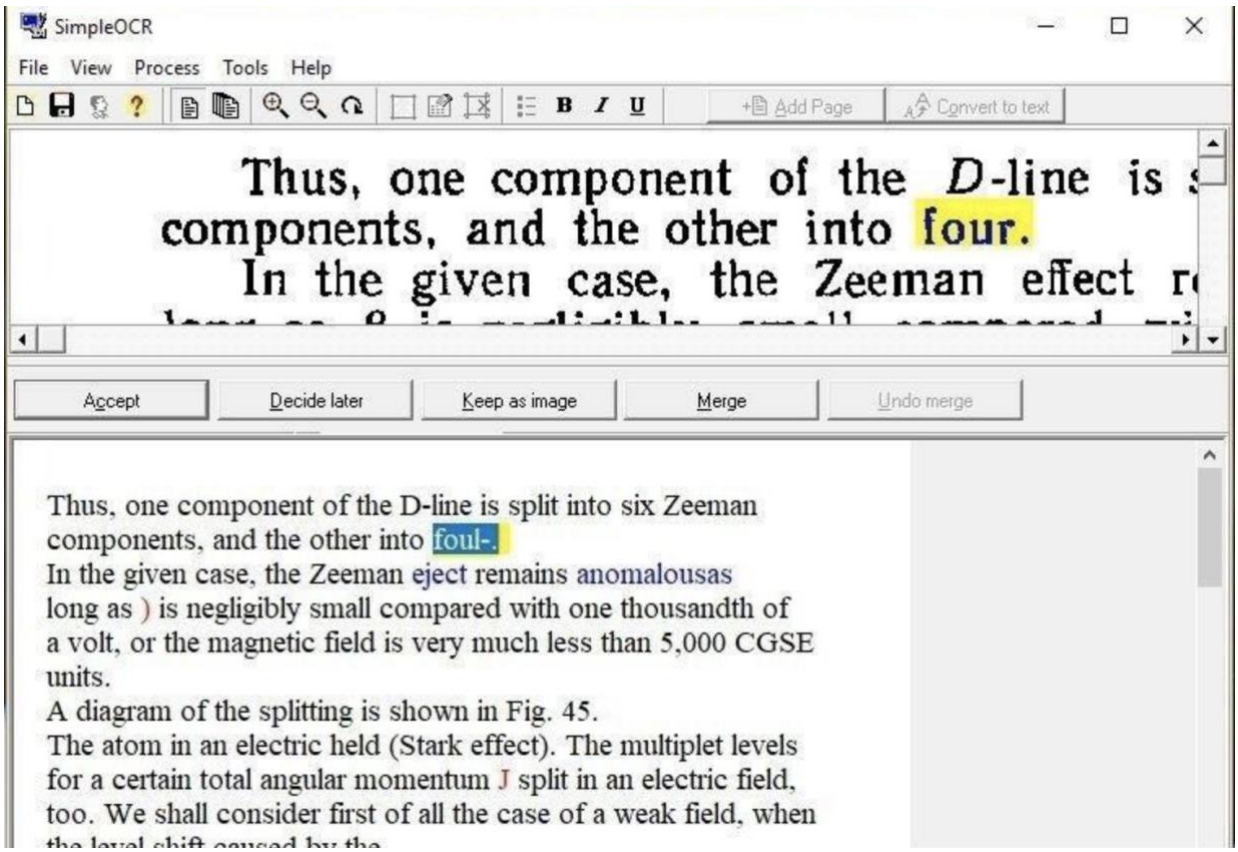


Рисунок 1.6 – SimpleOCR

Розробники TopOCR вважають, що вони створили найбільш потужну систему розпізнавання на основі нейронної мережі, що доступна на ринку, а також планують налаштувати OCR розпізнавання даних, зроблених за допомогою цифрової камери (рис. 1.7) [15].

У версію пакету «Microsoft Office» (рис. 1.8) включений модуль під назвою «Альтернативне введення», який передбачає введення рукописних текстів. Він працює для англійської, японської та китайської мов у відповідних версіях пакету.

Для використання функцій альтернативного введення потрібно вказати підключення відповідного компонента в програмі установки «Office» (Alternative User Input).

Засобами введення служать планшети з пір’яним введенням і звичайна миша. Робота здійснюється через меню «Handwriting» на мовній панелі [16].

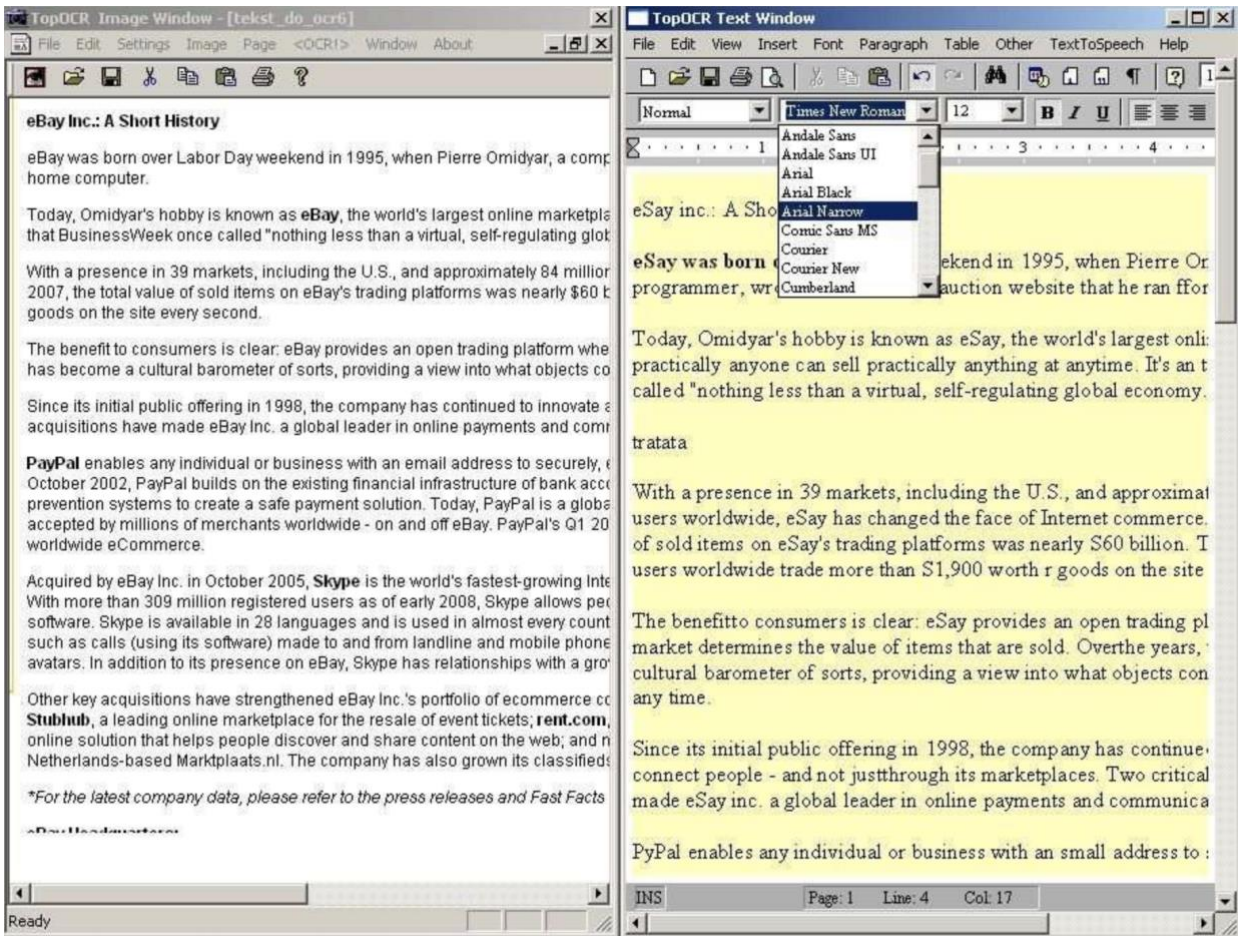


Рисунок 1.7 – TopOCR

Рисунок 1.8 – Пакет «Microsoft Office»

Перша персональна система розпізнавання «FreeStyle» [17] дозволяє вводити символи без пауз, використовуючи власний стиль, має модуль для створення макросів (часто використовуваних текстів). Пакет «FreeStyle» заснований на технології розпізнавання «OrthoGraph» фірми «ParaGraph International».

1.4 Аналіз літературних джерел щодо апробації результатів оброблення та розпізнавання рукописного тексту

У статті [6] відзначено недоліки фільтрів на основі вейвлет-перетворення. Проаналізовано основні методи відділення тексту від фону: порогова і адаптивна бінаризація, а також недоліки цих методів. У рамках методів бінаризації розглянуто метод Оцу, що вирішує завдання пошуку порогового значення яскравості. Також виявлено недоліки даного методу і способи їх усунення.

У статті [7] описано основні методи оброблення зображення, які застосовуються на етапі попереднього оброблення під час оффлайн-розпізнавання рукописного тексту. Розглянуто такі особливості предметної області, з точки зору OCR-систем: шрифтові та розмірні різноманітності символів, спотворення в зображеннях символів.

У статті [8] деталізовано принципи побудови системи оптичного розпізнавання тексту. Наводиться короткий опис основних груп методів, що використовуються під час розпізнавання тексту, виявлені їх переваги та недоліки. Відзначено недоліки фільтрів на основі вейвлет-перетворення. Розглянуто основні алгоритми відділення тексту від фону: порогова і адаптивна бінаризація.

Стаття [9] присвячена актуальному напрямку інформатики – розпізнаванню рукописного тексту. Виконано огляд систем розпізнавання, що використовують різні методи: шаблонні, ознакові, структурні. Розглянуто

переваги і недоліки цих методів. В рамках методів бінарихації розглянуто метод Оцу, що вирішує завдання пошуку порогового значення яскравості. Також відзначені недоліки даного методу і способи їх усунення.

У статті [10] розглянуті існуючі етапи розпізнавання рукописного тексту. На етапі попереднього оброблення зображення виконується підвищення якості зображення за рахунок фільтрації, шумопоглинання та інших методів, що мають на меті підвищити якість зображення. Використовується порогова бінаризація, яка дозволяє чітко розділити текст і фон, спрощує в подальшому застосування багатьох методів, а також позбавляє від деяких шумів на зображенні.

Стаття [11] присвячена використанню бібліотеки OpenCV в рамках виконання завдання розпізнавання рукописного тексту. Реалізація даного завдання, зазвичай, складається з множини етапів. Етапи попереднього оброблення зображення і сегментації є одними з них. Бібліотека OpenCV є досить потужним інструментом, який дозволяє вирішити завдання цих двох етапів з добрим результатом.

Стаття [12] містить інформацію про бібліотеку OpenCV, а саме про множину методів різних типів: розпізнавання об’єктів, усунення спотворень, виявлення подібності і форми об’єктів, відстеження переміщення об’єкта, розпізнавання рухів і жестів. Для попереднього оброблення зображень існує багато методів, у тому числі реалізації фільтра Гауса і медіанного фільтра, метод перетворення зображення в градації сірого, а також метод порогової бінаризації.

Стаття [14] описує питання підтримки морфологічних операцій у бібліотеці OpenCV, за допомогою яких можна виділити слова на зображенні. У даній статті докладно продемонстровано, яким чином можна об’єднати кілька методів, реалізованих в бібліотеці OpenCV, для вирішення завдань попереднього оброблення зображення і виділення на ньому слів в рамках завдання розпізнавання рукописного тексту, а також яких результатів можна досягти.

У статті [18] наводяться характерні особливості розпізнавання рукописного тексту: спотворення у зображеннях символів, розриви образів символів, злипання сусідніх символів, перекоси при скануванні, включення в зображеннях, поєднання фрагментів тексту на різних мовах, велика різноманітність класів символів, які можуть бути розпізнані лише при наявності додаткової контекстної інформації.

У рамках статті [19] розглянуто етапи оброблення тексту. На етапі попереднього оброблення відбувається підвищення якості зображення для подальшої його сегментації. Етап сегментації передбачає виділення тексту на зображенні та його поділ на складові частини. Зазвичай, текст обробляється ієрархічно: спочатку виділяються окремі рядки, потім окремі слова, символи або частини символів.

У ході аналізу літературних джерел щодо апробації результатів оброблення та розпізнавання рукописного тексту виявлено, що незважаючи на те, що завданням розпізнавання рукописних символів дослідники почали займатися ще з 70-х років XX століття (Ковалевський В.А., Рибак В.І., Фукунага К. та ін.), до цих пір є як теоретичні, так і практичні проблеми, пов’язані з великим різноманіттям написання окремих рукописних символів та тексту.

1.5 Постановка задачі дослідження

Таким чином, завдання оброблення і розпізнавання рукописного тексту є актуальним у різних сферах людської діяльності, а розроблення методів розпізнавання рукописного тексту дозволить підвищити ефективність роботи інформаційних систем. Розпізнавання рукописного тексту на зображеннях є важливим завданням машинного навчання, як дозволяє організувати зручну взаємодію з даними: редагування, аналіз, пошук слів чи фраз. Необхідно проаналізувати застосунки, що використовують сучасні методи оброблення рукописного тексту, щоб застосувати вибрані методи до реальної задачі.

Об’єктом дослідження є зразки рукописних текстів.

Метою дослідження є програмна реалізація досліджених методів оброблення та розпізнавання рукописних текстів, а також порівняння їх точності розпізнавання на базі створеного набору зразків.

Для досягнення мети необхідно вирішити такі завдання:

– класифікувати існуючі методи оброблення та розпізнавання рукописного тексту;

– здійснити аналіз сучасних систем розпізнавання рукописного тексту;

– розробити модель оброблення та розпізнавання рукописного тексту за допомогою HMM;

– розробити модель оброблення та розпізнавання рукописного тексту за допомогою нейронних мереж;

– програмно реалізувати розроблені моделі оброблення та розпізнавання рукописного тексту у окремих застосунках;

– виконати тестування розроблених застосунків;

– порівняти точність розпізнавання досліджених методів на базі створеного набору зразків.

2 ОСОБЛИВОСТІ ВИБРАНИХ МЕТОДІВ ОБРОБЛЕННЯ ТА РОЗПІЗНАВАННЯ РУКОПИСНОГО ТЕКСТУ

2.1 Метод виділення ключових ділянок на зображенні

При розпізнаванні рукописних символів на зображеннях важливу роль займає виявлення зон, що містять рукописні знаки. Для виявлення таких зон часто використовують метод гістограм. Зображення послідовно сканують, підсумовуючи при цьому кількість пікселів у рядку, на основі отриманих результатів будують гістограму. Потім проводиться аналіз гістограми на максимуми та мінімуми. Максимальне значення показує ймовірне розташування ключової ділянки, що містить символи. Слід зазначити, що даний підхід ефективний тоді, коли рядки розташовані горизонтально [20].

Якщо рядки розташовані не горизонтально, необхідно багаторазове сканування в різних напрямках та вибір такого напрямку, який забезпечує максимально виражені максимуми і мінімуми на отриманій гістограмі. Даний факт накладає суттєві обчислювальні обмеження і вимагає великих ресурсів машинного часу і пам’яті. Необхідно запропонувати модифікований метод, заснований на виявленні текстових зон за допомогою морфологічного оброблення з подальшим виявленням пов’язаних областей, крім того, такий метод дозволить визначити орієнтацію вхідного зображення, що істотно спрощує процедуру розпізнавання та збільшує ефективність системи розпізнавання [21].

У загальному випадку модель визначення текстових зон на зображенні можна виразити такою формулою:

$$TZ = < \{On_{TZ}\}, \{Om_{TZ}\}, \{Oc_{TZ}\}, \{Os_{TZ}\} >,$$

де $\{On_{TZ}\}$ – оператор попередньої нормалізації текстових зон;

$\{Om_{TZ}\}$ – оператор морфологічного оброблення;

$\{Oc_{TZ}\}$ – оператор категоризації символів;

$\{Os_{TZ}\}$ – оператор знаходження спеціальних символів.

Модифікований метод виявлення можна розділити на наступні етапи:

– попередня обробка (усунення шумів, бінаризація);

– морфологічна обробка (операції розширення і стиснення);

– виявлення пов’язаних областей і побудова текстових зон;

– визначення кута повороту текстових зон щодо горизонтального напрямку [22].

На етапі попередньої обробки для усунення перешкод на зображенні застосовуються різні методи фільтрації. Найбільш простими є фільтри, що згладжують: лінійний та медіанний. Оскільки зображення з рукописними символами найчастіше є двокольоровим зображенням, то доцільно перетворити його у бінарне методом порогової бінаризації. Як поріг можна використовувати середнє значення яскравості пікселів зображення. Слід зазначити, що існують більш складні схеми вибору порогу. Однак, оскільки в даному випадку потрібна лише контурна інформація, їх застосування є недоцільним [23]. При пороговій бінаризації привласнення значення вихідного елементу виконується за формулою:

$$Q(i,j) = \begin{cases} 0, \text{якщо } A(i,j) < P, \\ 1, \text{якщо } A(i,j) \geq P, \end{cases}$$

де $A(i,j)$ – значення яскравості елемента вихідного зображення;

$Q(i,j)$ – значення бінарного зображення;

P – значення порогу.

На етапі морфологічної обробки здійснюється послідовне застосування операції розширення і стиснення. У морфологічних алгоритмах беруть участь цифрові зображення, задані функціями $f(x,y)$ та $b(x,y)$, де функція $f(x,y)$ – вихідне зображення, а $b(x,y)$ – зображення примітиву [24].

Тоді операція розширення f по b визначається як:

$$(f \oplus b)(s,t) = max\{f(s-x,t-y) + b(x,y)\}\big|(s-x,t-y) \in D_f(x,y) \in D_b),$$

де D_f і D_b – області визначень зображень f і b відповідно;

s і t – зрушення координат по осях X і Y.

Аналогічним чином визначається операція стиснення f по b:

$$(f \oplus b)(s,t) = min\{f(s+x,t+y) - b(x,y)\}\big|(s+x,t+y) \in D_f(x,y) \in D_b),$$

де D_f і D_b – області визначень зображень f і b відповідно;

s і t – зрушення координат по осях X і Y.

Як примітиви в операції розширення використовуються маски апертурою 3×5, 3×7 і вище (залежно від середньої висоти рукописних символів), у результаті цього контури символів, що близько знаходяться, будуть пов’язані в загальний контур текстової зони (рис. 2.1 а). Далі застосовується операція стиснення для згладжування зовнішніх країв пов’язаних областей. Як примітив використовується маска апертурою 3×3.

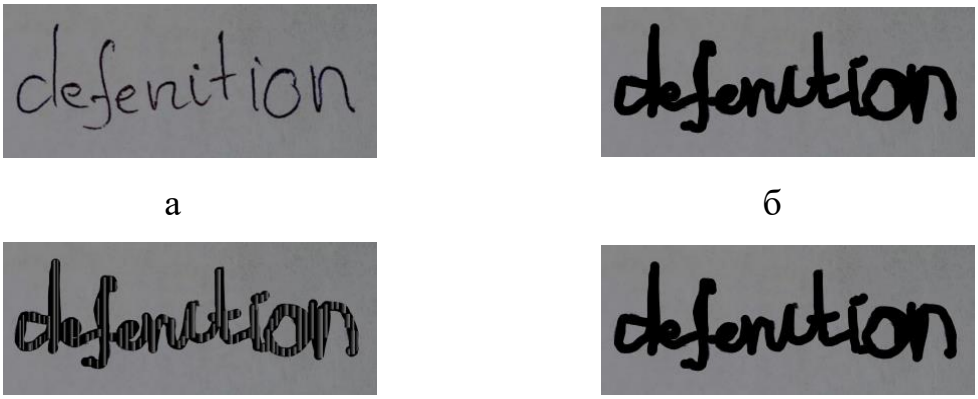
Дані операції можуть здійснюватися послідовно декілька разів для більш ефективного злиття в загальні області (вибираються емпіричним шляхом для відповідних примітивів операцій) (рис. 2.1 б). Дослідження показує, що для маски апертурою 3×5 необхідно в середньому виконати 3 операції розширення і стиснення, а для маски апертурою 3×7 досить 1-2 операції розширення і стиснення [25].

Після цього здійснюється маркування пов’язаних областей з урахуванням оточуючих маркерів (рис. 2.1 в). Як оточуючий маркер використовується маркер вищого пікселя та пікселя зліва.

Якщо вищий піксель позначений маркером, то для поточного пікселя при скануванні зображення встановлюється аналогічний маркер [26].

В іншому випадку поточний піксель позначається маркером, яким володіє піксель зліва. Якщо цей піксель не має маркера, то поточний піксель маркується наступним маркером [27].

На підставі отриманих маркерів будується таблиця зв'язності маркованих областей і здійснюється зв'язування цих областей в загальну текстову зону з позначкою даної зони індексом (рис. 2.1 г).



а б в г

Рисунок 2.1 – Побудова загальних текстових зон: а – вихідне зображення; б – виділення зображень зон; в – маркування областей; г – зв'язування областей в текстові зони

За середніми кутах нахилу текстових зон щодо горизонтальної лінії обчислюється кут повороту всього зображення для забезпечення більш якісної сегментації текстових зон на окремі символи [28].

Обчислюються координати центрів мас зображень текстових зон і далеких віддалених точок, а також кути повороту зображень текстових зон:

$$Alfa_i = arctan((y_{im} - y_{ic})/(x_{im} - x_{ic})),$$

де $Alfa_i$ – кут повороту i-ї текстової зони;

x_{im}, y_{im} – координати далекої віддаленої точки i-ї текстової зони;

x_{ic}, y_{ic} – координати точки центру мас i-ї текстової зони [29].

Відповідно кут повороту всього зображення обчислюється знаходженням середньоарифметичного кутів текстових зон:

$$Alfa = \frac{\sum_{i=1}^{n} Alfa_i}{n},$$

де $Alfa$ – кут повороту зображення;

$Alfa_i$ – кут повороту i-ї текстової зони;

n – загальне число текстових зон [30].

Після цього зображення повертається на обчислений кут, знайдені зони виділяються прямокутною областю. Відбувається накладення цих областей на початкове зображення і здійснюється сегментація виділених текстових зон (рис. 2.2).

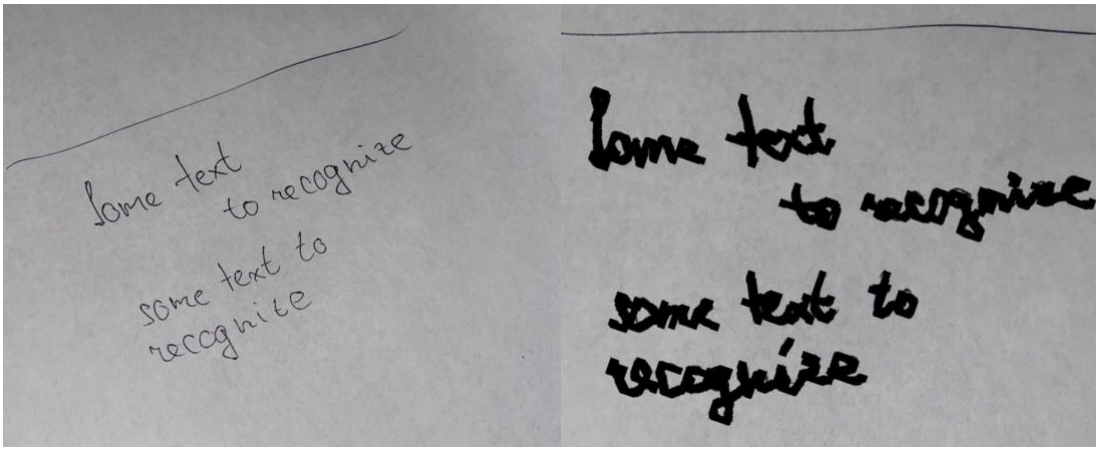


Рисунок 2.2 – Приклад сегментації і нормалізації документа

2.2 Метод сегментації на окремі складові текстових зон

Наступним етапом у розпізнаванні рукописного тексту є поділ слів на окремі складові (символи). Тут діють оператори посилення центральної частини символу Oc_{SC}, визначення верхньої частини символу Ou_{SC},

визначення нижньої частини символу Od_{SC}, пошуку та аналізу прийменників Oa_{SC}, адаптивного знаходження символів в слові Os_{SC} [31].

Модель визначення окремих символів на зображенні слова виражається кортежем множин:

$$SC =< \{Oc_{SC}\}, \{Ou_{SC}\}, \{Od_{SC}\}, \{Oa_{SC}\}, \{Os_{SC}\} >.$$

У цьому випадку, зображення (рис. 2.3 а) піддається морфологічній обробці з метою виділення можливого ядра кожного символу в текстовій зоні [32]. Для цього будується гістограма розподілу щільності пікселів по рядках (рис. 2.3 б).

Нехай $IM(i, j)$ – зображення групи символів, HG – висота текстової зони, WG – ширина текстової зони [33].

Масив середніх значень яскравості піксельних рядків у зображенні текстової зони обчислюється так:

$$SG\_ROM(i) = sum_j(IM(i, j))/WG,$$

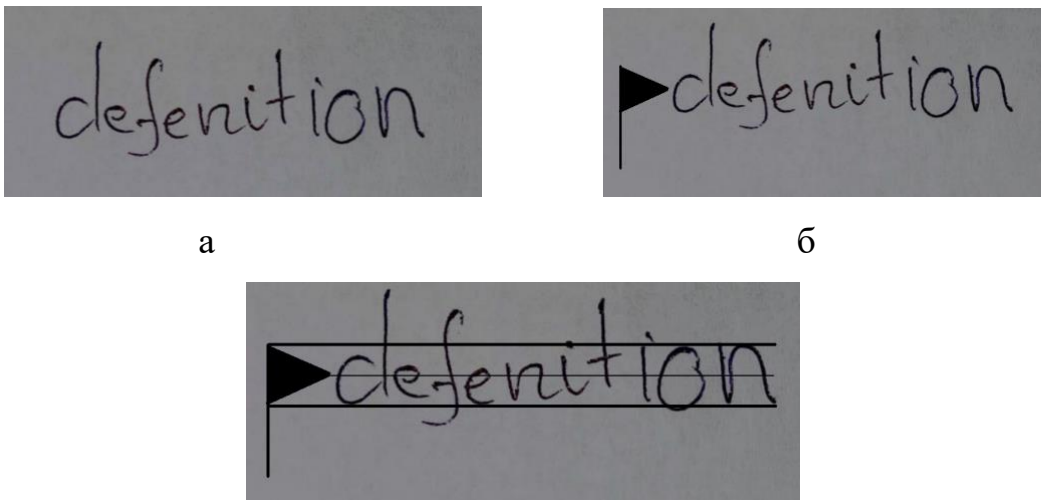
де $i = 1,2, \ldots, HG$;

$j = 1,2, \ldots, WG$.

Далі в масиві $SG\_ROM(i)$ знаходяться елементи, значення яких максимальні.

Робиться припущення про те, що це значення відповідає базовій лінії зображень символів (рис. 2.3 в). Також по обидва боки від базової лінії шукається границя текстової зони, і зображення буде скорочуватися по цих межах.

Зображення верхньої та нижньої частин зони щодо базової лінії піддаються морфологічної процедурі розширення з примітивами розмірністю 3×3 (рис. 2.4) [34].



а б

в

Рисунок 2.3 – Побудова гістограми та пошук базової лінії:
а – вихідне зображення; б – гістограма; в – базова лінія

0	1	0
0	1	0
0	0	0

а

0	0	0
0	1	0
0	1	0

б

Рисунок 2.4 – Примітиви морфологічної операції розширення:
а – для верхньої частини зони; б – для нижньої частини зони

У результаті такої процедури виходить розширене зображення групи символів, в яких внутрішні області символів найбільш насичені (ядра) [35].

При цьому міжсимвольні відстані залишаються практично незмінними (рис. 2.5).

а б

Рисунок 2.5 – Посилення ядра символів: а – вихідне зображення;
б – зображення після оброблення

Серед інформативних зон шукаються найменші зони (поодинокі символи) [36]. Обчислюється середнє значення розміру чарунку для декількох типів символів (малі, великі та малі з додатковими елементами – «в», «б», «д», «у» і т. д.). На решту групи символів накладається отриманий чарунок. Припустимо, що написання тексту йде зліва направо [37]. У цьому випадку для визначення границі символу на правій межі виділяють чарунок по гістограмі, шукаючи локальний мінімум щільності точок зображення текстових символів (рис. 2.6).

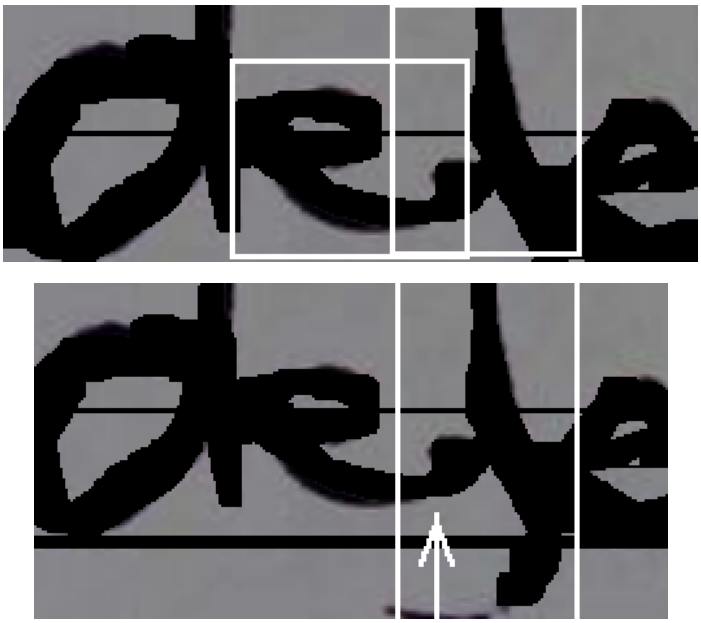


Рисунок 2.6 – Адаптивне виділення символів

Обчислюється масив середніх значень яскравості піксельних стовпців у зображенні групи символів:

$$SG\_COL(j) = sum_i(IM(i,j))/HG,$$

де $i = 1,2,\ldots,HG$;

$j = 1,2,\ldots,WG$.

У масиві $SG\_COL(j)$ знаходяться номери стовпців, які входять в поточну область на правій межі виділеного чарунку, для яких значення елементів

масиву мінімальні. Робиться припущення про те, що ці стовпці представляють собою місце розташування пов’язуючого символи елемента і відбувається корекція виділеного чарунку до цього місця розташування, а також відбувається зафарбовування кольором фону кордону виділеного чарунку (рис. 2.7). Таким чином, виходять нові параметри виділеного чарунку, які заносяться в таблицю для подальших операцій [38].

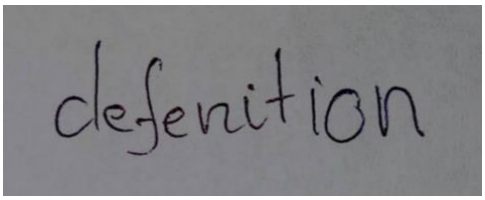


Рисунок 2.7 – Результат сегментації текстової зони на окремі символи

Наступні параметри виділеного чарунку (довжина, ширина) вибираються як середні між усіма параметрами чарунків, занесених в таблицю. У результаті відбувається сегментація зображень груп символів на окремі символи і виходить набір сегментованих зображень. Отримані окремі символи піддаються процедурі векторизації зовнішнього контуру і класифікації [39].

2.3 Метод векторного розпізнавання рукописних символів

2.3.1 Побудова векторної моделі

У задачах розпізнавання зображень ключовим моментом є виділення деяких характерних ознак зображення. Зокрема, при розпізнаванні рукописних символів такою характерною ознакою може бути зображення зовнішнього контуру. У цьому випадку використання векторного підходу для опису зовнішнього контуру зображення рукописних символів є найбільш ефективним, оскільки дозволяє отримати інваріантність по відношенню до перетворень афінної групи.

Етап переходу до нормалізованих образів окремих символів пов’язаний з виділенням зовнішнього контуру символу, параметризацією векторного подання контуру і нормалізацією векторного подання. Якщо завдання виділення зовнішнього контуру об’єкта і параметризація його векторного подання є добре виконаним, то нормалізація векторного подання у контексті поставленого завдання вимагає додаткових зусиль [40].

Особливістю розпізнавання рукописних символів є велика різноманітність контурів.

Пропонується інваріантність до масштабу та різних стилів написання символів забезпечувати приведенням суми довжин векторів до одиниці та вибором напрямків векторів відповідно до стандартних напрямків, а інваріантом вибору початкової точки обходу контуру вважати вектор з мінімальною довжиною.

При наявності декількох векторів з мінімальною довжиною використовуються спеціальні правила вибору початкової точки обходу, що реалізуються на етапі навчання системи.

Таким чином, модель нормалізації образу символу можна представити у вигляді:

$$VD = < \{Oo_{VD}\}, \{Ov_{VD}\}, \{On_{VD}\} >,$$

де $\{Oo_{VD}\}$ – оператор виділення зовнішнього контуру символу;

$\{Ov_{VD}\}$ – оператор параметризації векторного подання контуру;

$\{On_{VD}\}$ – оператор нормалізації векторного подання.

Для побудови векторної моделі символу і її класифікації спочатку необхідно знайти опорні точки зовнішнього контуру символу.

Опорною точкою є така точка, де лінія, що описує зовнішній контур зображення символу, має вигин. Для знаходження таких точок пропонується використовувати модифікований фільтр Робертса:

$$A' = |A - D| + |B - C| \text{ або } A' = \sqrt{(A - D)^2 + (B - C)^2},$$

причому, якщо $A' > 255$, то $A' = 127$ (будь-яке значення відмінне від 255), де A' – нове значення яскравості пікселя вікна;

A, B, C, D – початкові значення яскравості пікселів вікна.

Результат роботи фільтра зображений на рисунку 2.8.

а б

Рисунок 2.8 – Модифікований фільтр Робертса:
а – вихідне зображення; б – зображення після фільтрації

Після накладення даного фільтра на бінарне зображення здійснюється його сканування на предмет належності кожної точки зображення до класу опорних або не опорних точок [41].

Модифікація фільтра дозволяє отримати на виході зображення в трьох градаціях яскравості пікселів. Опорна точка при цьому буде мати максимальне значення яскравості. Після знаходження будь-якої з опорних точок здійснюється обхід зображення по зовнішньому контуру з послідовним визначенням всіх наступних опорних точок.

Для обходу зображення застосовується хвильовий алгоритм (рис. 2.9).

Алгоритм працює з двовимірним масивом, що є матрицею зображення, отриманою після попередньої обробки і містить інверсні значення яскравості пікселів зображення. У процесі роботи алгоритму здійснюється поширення в 4-х напрямках від попереднього стану.

Функціонування алгоритму – це спрямоване поширення напрямку вздовж передбачуваного контуру.

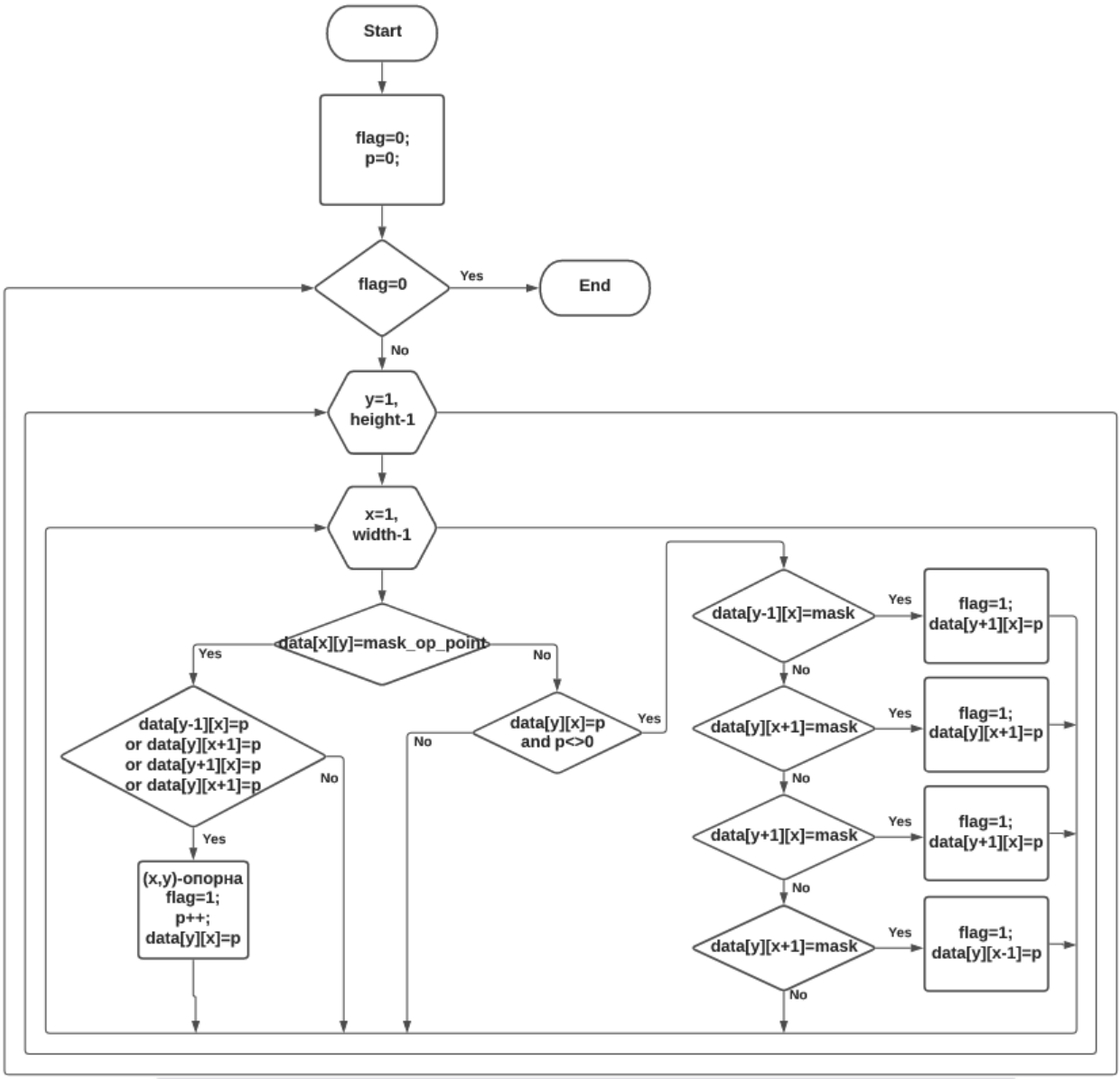


Рисунок 2.9 –Хвильовий алгоритм

Даний алгоритм здійснює пошук зв’язності опорних точок від першої знайденої опорної точки (при скануванні зображення зліва – направо та зверху – вниз) до останньої опорної точки в даному контурі (рис. 2.10).

Рисунок 2.10 – Пошук опорних точок

Після знаходження опорних точок і занесення їх в таблицю будується векторна модель, що є набором векторів і послідовність їх зв’язку між собою:

$$Model = \{V_1, V_2, \ldots, V_i, V_{i+1} \ldots V_n\}.$$

Початок кожного наступного вектора V_{i+1} знаходиться в кінці попереднього V_i, утворюючи замкнутий контур [42].

Кожен вектор характеризується парою координат: кут відносно горизонтальної лінії $(alfa)$ і довжина вектора (len). Визначення координат вектора $V_i = (Len_i, Alfa_i)$ здійснюється шляхом переходу від абсолютних координат до відносних координат:

$$Len_i = \sqrt{x_i^2 + y_i^2}, Alfa_i = ArcTan\frac{y_i}{x_i},$$

де x_i, y_i – відносні координати вектора, які знаходяться за формулою:

$$x_i = x_{j+1} - x_j, y_i = y_{j+1} - y_j,$$

де x_i, y_i – координати поточної опорної точки;

x_{i+1}, y_{i+1} – координати наступної опорної точки.

2.3.2 Перетворення векторної моделі до інваріантного виду

Після побудови векторна модель піддається процедурам ущільнення і нормалізації.

Під ущільненням розуміється видалення несуттєвих векторів, що з'єднують точки, що знаходяться поруч. Дана процедура здійснюється за наступною схемою:

Крок 1. z = l.

Крок 2. Вибирається V_i вектор моделі.

Крок 3. Якщо довжина V_i менше деякого порогового значення EPS, то даний вектор видаляється з моделі.

Крок 4. $i = i + 1$.

Крок 5. Якщо $i < N$, то перехід на Крок 2 (здійснюємо перебір всіх векторів моделі).

Після ущільнення виконується процедура нормалізації, яка полягає в нормалізації довжин векторів і кутів векторів по напрямками [43].

Нормалізація довжини відбувається наступним чином: сума довжин векторів у моделі проводиться до одиничної довжини, а всі інші довжини векторів відповідно до цього масштабуються:

$$Len_i = Len_i/K,\ K = \sum_{i=1}^{n} Len_i,$$

де K – коефіцієнт масштабування.

Нормалізація кутів за напрямками здійснюється наступним чином: обчислюється найближчий напрямок вектора до поточного вектору в моделі, проводиться нормалізація кута до даного напрямку (кут вектора вибирається рівним куту напрямку). Як напрямки вибираються одиничні вектори:

$$E_i = (1, E\_Alfa_i),\ E\_Alfa_i = \frac{i \times 2 \times \pi}{Z},$$

де Z – кількість напрямків.

Таким чином, відповідно до запропонованої побудови векторної моделі у залежності від вибору початкової опорної точки можна отримати N варіантів векторної моделі (N – кількість векторів у моделі):

$$Model\_1 = \{V_1, V_2 \ldots, V_i, \ldots, V_n\},$$

$$Model\_2 = \{V_n, V_1, V_2 \ldots, V_i, \ldots, V_{n-1}\},$$

$$Model\_3 = \{V_{n-1}, V_n, V_1, V_2 \ldots, V_i, \ldots, V_{n-2}\},$$

$$Model\_n = \{V_2, V_3 \ldots, V_i, \ldots, V_n, V_1\}.$$

Послідовний перебір всіх отриманих моделей вимагає додаткових ресурсів часу та пам’яті в процесі розпізнавання. Доцільним є побудова векторної моделі інваріантної до вибору початкового вектора:

Крок 1. Визначається вектор мінімальної довжини моделі V_k.

Крок 2. Якщо вектор один, то здійснюється побудова інваріантної моделі, яка є набором векторів. Як початковий вектор вибирається знайдений мінімальний вектор:

$$Model = \{V_k, V_{k+1} \ldots, V_i, \ldots, V_{k-1}\}.$$

Крок 3. Якщо векторів мінімальної довжини декілька, то здійснюється пошук «найкращого» вектора, здійснюється побудова інваріантної моделі. Як початковий вектор вибирається знайдений «найкращий» вектор V_i:

$$Model = \{V_t, V_{t+1} \ldots, V_i, \ldots, V_{t-1}\}.$$

Алгоритм пошуку «найкращого» вектора моделі полягає в наступному:

Крок 1. Вибирається перший знайдений вектор мінімальної довжини.

Крок 2. Визначається сумарна довжина шляху до подальших векторів мінімальної довжини.

Крок 3. Вибирається наступний вектор мінімальної довжини.

Крок 4. Якщо не всі мінімальні вектора оброблені, то перехід до Кроку 2.

Крок 5. Вибирається вектор, сумарна довжина шляху від якого мінімальна. Цей вектор вважається «найкращим» [44].

2.3.3 Розпізнавання рукописних символів на основі інваріантної векторної моделі

Різні реалізації рукописних знаків одного класу відрізняються один від одного перенесенням, масштабом, нахилом, пропорціями і т.д.

Для забезпечення інваріантності опису до цих перетворень розглядаються послідовності елементарних напрямків, кожен з яких характеризує взаємне розташування двох сусідніх точок зображення.

Всього вибрано N можливих елементарних напрямків z, а зображення описується у вигляді:

$$X_l = (x_1, x_2, \ldots, x_i \ldots, x_l),$$

де $x_i = \{z\}$;

$z = 1 \ldots N$ – i-й елемент зображення, що є одним з векторів моделі;

l – довжина зображення.

Послідовності, що складаються з елементів, напрямок і порядок слідування яких зберігаються постійними для всіх реалізацій знаків одного класу, називаються штрихами.

Змінюючи в допустимих межах довжину окремих штрихів додаванням або викиданням деякого числа елементів, а також виробляючи нормалізацію,

можна отримати декілька зображень одного класу. У результаті генеруються ідеалізовані рукописні знаки, що називаються еталонами даного класу:

$$E_l^j(e_1^j, e_2^j, \ldots, e_i^j, \ldots, e_l^j),$$

де e_l^j – i-й елемент еталона, визначений на певному напрямку z.

Вплив шуму, спричиненого, наприклад, коливанням кінчика пера або дискретністю сітківки, враховується завданням розподілу $P(X_l/E_l^j)$, визначеного на множині елементарних напрямків.

Для побудови алгоритму розпізнавання використовується метод максимальної правдоподібності. Вирішальне правило визначається виразом

$$V_j^0 = arg\ \max_j \max_l P(X_l/E_l^j).$$

Максимум шукається по всіх класах та по всіх можливих стандартах довжиною l. За допомогою цього правила шукається еталон довжиною l, найближчий до зображення X_l.

Для реалізації вирішального правила замість величини $P(X_l/E_l^j)$ можна використовувати монотонно спадаючу функцію від цієї величини, наприклад:

$$g(X_l/E_l^j) = \sum_{i=1}^{n} g(x_i, e_i^j),$$

де $g(X_l/E_l^j)$ – функція близькості елементів вхідного зображення x_i, і еталона e_i^j, що визначається цілим додатнім числом, пропорційним абсолютній величині кута між двома порівнюваними елементарними напрямками, а також різницею довжин елементів у векторній моделі еталона та даного вхідного зображення.

Таким чином, вирішальне правило може бути виражене в такий спосіб:

$$V_j^0 = arg\ \max_j \max_l) \sum_{i=1}^{n=l} g(x_i, e_i^j).$$

Серед усіх класів V шукається такий клас V_j^0, еталон довжиною l якого дає мінімальне значення функції відмінності (або максимальне значення функції подібності).

Здійснюється визначення оцінки такого вигляду:

$$V_j^0 = arg\,\min_j \sum_{i=0}^{n} (|Alfa_i, Len_i| - E\_Alfa_i^j, E\_Len_i^j|)^2,$$

де $Alfa_i$, Len_i – координати i-го вектора у вхідній моделі;

E_Alfa_i, E_Len_i – координати i-го вектора у j-й еталонній моделі.

2.4 Розпізнавання рукописного тексту на основі ймовірнісного підходу

2.4.1 Розробка лінгвістичної моделі слова

Модель розпізнавання рукописного тексту включає оператор побудови лінгвістичної моделі слова Ol_{RHT}, оператор розпізнавання рукописних символів Osr_{RHT} і оператор підключення тематичних словників Ots_{RHT}:

$$RHT = < \{Ol_{RHT}\}, \{Osr_{RHT}\}, \{Ots_{RHT}\} >.$$

Для підвищення достовірності розпізнавання слова повністю пропонується побудова імовірнісної лінгвістичної моделі слова з застосуванням тематичних словників. На основі такої моделі можна побудувати таблицю ймовірностей і тим самим здійснювати підстроювання системи розпізнавання одиночних символів [45].

Нехай $A = \{a_1, a_2, \ldots, a_n\}$ – алфавіт мови, що містить букви і цифри (для української мови це 33 букви і 10 арабських цифр).

Тоді слово можна записати наступним чином: $w = a_i, a_s \ldots a_k$. Можна побудувати словник слів $S = \{w_1, w_2, \ldots, w_i, w_{wl}\}$, де w_i – i-е слово.

Таким чином, можна використовувати інформацію про слово у системі розпізнавання шляхом вибору з всіх еталонів символів найбільш ймовірних символів з урахуванням їх входження в кожне слово.

Нехай є набір еталонів векторних моделей символів $E = \{e_1, e_2, \ldots, e_{cl}\}$, що упорядкований по зростанню символьних найменувань класів моделей («0», «1», ..., «9», «а», «б», «в», ..., «я»).

Упорядкований в алфавітному порядку словник слів $S = \{w_1, w_2, \ldots, w_i, w_{wl}\}$, де $w_j = a_{1j}a_{2j}a_{3j}\ldots a_{kj}$ – j-е слово.

Позначимо через $r = r_1 r_2 r_3 \ldots r_i$ – частину розпізнаного слова, що складається з i символів r_i.

Нехай N – кількість слів w_i зі словника S, в які входить поточна частина розпізнаного слова r. Тоді ймовірність кожного можливого чергового еталонного символу e^q q-го класу в передбачуваному слові дорівнює:

$$Pv^q(a_{i+1j}) = 1/q_k + sum_j(a_{i+1j})/N | a_{i+1j} = e^q,$$

де $q = 1 \ldots q_k$;

q_k – кількість класів моделей.

Таким чином, виходить імовірнісна лінгвістична модель слова, яку можна використовувати в алгоритмі розпізнавання, що дозволяє серед усього розмаїття еталонних векторних моделей вибирати ті, які скоріш за все описують слово, що розпізнається [43].

2.4.2 Адаптивна побудова дерева рішень на основі імовірнісного підходу

Для побудови алгоритму розпізнавання використовується метод максимальної правдоподібності. Вирішальне правило визначається виразом:

$$V_j = argmax_j max_i P(X_l, E_i^j).$$

Максимум шукається по всіх класах і по всіх можливих стандартах. За допомогою цього правила знаходиться еталон довжиною l, найближчий до зображення X_l. Міра близькості зображення і еталонів обчислюється за формулою:

$$D_j = sum_i(|Alfaa_i Len_i| - E\_Alfa_i^j, E\_Len_i^j|)^2,$$

де D_j – міра близькості вхідної моделі символу з j-ю еталонною моделлю;

$Alfa_i, Len_i$ – координати i-го вектора у вхідній моделі;

E_Alfa_i, E_Len_i – координати i-го вектора j-ї еталонної моделі.

Досліджено декілька різних варіантів метрик.

Метрика Левенштейна:

$$D(i,j) = \begin{cases} 0, & i = 0, j = 0, \\ i, & j = 0, i > 0, \\ j, & i = 0, j > 0, \\ \min\{D(i,j-1) + 1, & \\ \quad D(i-1,j) + 1, & \\ D(i-1,j-1) + m(S_1[i], S_2[j])\}, & \\ j > 0, i > 0. & \end{cases}$$

Манхеттенська метрика:

$$D(p,q) = \|p - q\|_1 = \sum_{i=1}^{n} |p_i - q_i|.$$

Відстань Хеммінга:

$$d_{i,j} = \sum_{k=1}^{p} |x_{ik} - x_{jk}|.$$

Дослідження показують, що найбільша точність розпізнавання досягається при використанні Манхеттенської метрики (рис. 2.11). У системі розпізнавання пропонується використовувати саме її.

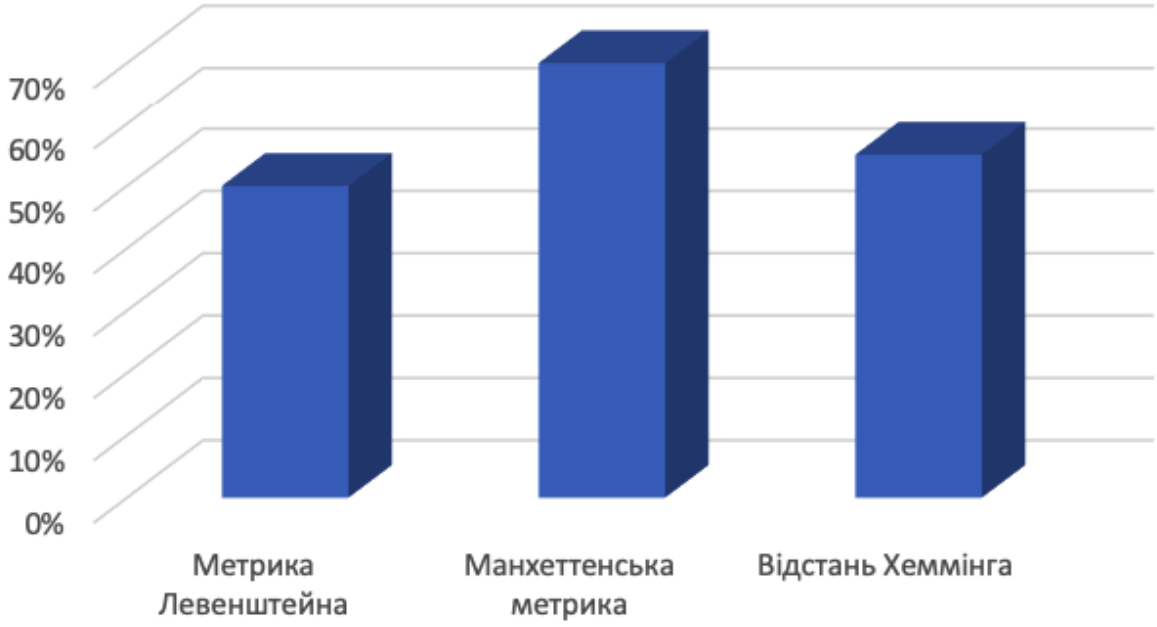


Рисунок 2.11 – Точність розпізнавання для різних метрик

Імовірність того, що вхідна модель відноситься до певного класу, може бути обчислена за такою формулою:

$$Pe_j = (1 - D_j / sum_j(D_j)) / sum_j(D_j).$$

Остаточне значення ймовірності належності вхідної моделі j-й еталонній моделі з урахуванням ймовірнісної моделі слова буде дорівнювати:

$$P(X_l E_i^j) = Pv^i Pe_j.$$

Якщо значення ймовірності $P(X_l E_i^j)$ нижче деякого порогового значення, то передбачається, що в тексті є слово, якого немає в словнику.

У цьому випадку ймовірність чергового еталонного символу Pv^i приймається рівною одиниці, слово розпізнається побуквенно без урахування моделі слова і заноситься в поточний додатковий словник.

Таким чином, необхідно мати певний набір словників та в залежності від тематики тексту здійснювати вибір того чи іншого словника, що дозволить врахувати особливість тексту в цілому.

Можна використовувати такі електронні словники: тлумачний словник Ожегова або Даля, орфографічний словник Лопатіна тощо.

3 РЕАЛІЗАЦІЯ ТА ДОСЛІДЖЕННЯ ВИБРАНИХ МЕТОДІВ ОБРОБЛЕННЯ ТА РОЗПІЗНАВАННЯ РУКОПИСНОГО ТЕКСТУ

3.1 Вибір інструментальних засобів для реалізації вибраних методів оброблення та розпізнавання рукописного тексту

Для написання програми для розпізнавання рукописного тесту обрано мову програмування Python, яка є популярною і потужною інтерпретованою мовою. Це інструмент, який можна використовувати як для досліджень та чисельних розрахунків, так і для розроблення виробничих систем. У Python є багато модулів та бібліотек, які забезпечують декілька способів виконання кожного завдання.

Дана мова програмування підтримує множину парадигм: структурна, об’єктно-орієнтована, функціональна, імперативна і аспектно-орієнтована.

У Python присутня динамічна типізація, автоматичне керування пам’яттю, повна інтроспекція, механізм оброблення виключень, підтримка багатопотокових обчислень та зручні високорівневі структури даних. Програмний код на Python організовується у функції та класи, які можуть об’єднуватися в модулі, а вони, у свою чергу, можуть бути об’єднані в пакети.

Головною бібліотекою для реалізації машинного навчання обрано TensorFlow. TensorFlow – це відкрита бібліотека, що створена Google, вона використовується при розробленні систем, що засновані на технології машинного навчання. Ця бібліотека включає в себе реалізацію множини потужних алгоритмів, розрахованих на вирішення поширених завдань машинного навчання, серед яких можна відзначити розпізнавання образів та прийняття рішень.

Проєкт TensorFlow був переведений Google в розряд відкритих ще у 2015 році, його попередником був проєкт DistBelief, роки досвіду, накопичені у ході роботи з яким, відобразилися на TensorFlow.

Розробники бібліотеки TensorFlow прагнули до того, щоб вона була гнучкою та ефективною. Її можна користуватися в різних обчислювальних середовищах: від тих, які формуються мобільними пристроями, до середовищ, представлених величезними кластерами.

Бібліотека дозволяє швидко готувати до реальної роботи навчені моделі, що усуває необхідність у створенні особливих реалізацій моделей. Ця бібліотека, завдяки спільним зусиллям всіх тих, хто працює над нею, підходить для вирішення завдань різних масштабів. Від тих, які виникають перед самостійним розробником, до тих, які займаються стартапами, і навіть перед великими компаніями на кшталт Google.

З того моменту, як ця бібліотека стала відкритою, вона стала однією з найцікавіших бібліотек машинного навчання. Її все частіше і частіше використовують під час проведення досліджень, при розробленні реальних програм, при навчанні.

TensorFlow постійно поліпшується, її постійно постачають чимось новим, оптимізують. Крім того, зростає і спільнота, сформована навколо цієї бібліотеки.

3.2 Етапи програмної реалізації вибраних методів оброблення та розпізнавання рукописного тексту

3.2.1 Застосунок, що використовує нейронну мережу, як спосіб розпізнавання рукописного тексту

3.2.1.1 Модуль організації інтерфейсу з користувачем

Модуль організації інтерфейсу с користувачем здійснює координацію роботи системи в цілому і забезпечує механізми взаємодії з користувачем. Подача вхідних даних здійснюється за допомогою CLI або через інтерактивне середовище програми PyCharm за допомогою аргументів, які можна подати, як вхідні параметри (рис. 3.1).

```python
def main():
    """Main function."""
    parser = argparse.ArgumentParser()

    parser.add_argument('--mode', choices=['train', 'validate', 'infer'], default='infer')
    parser.add_argument('--decoder', choices=['bestpath', 'beamsearch', 'wordbeamsearch'], default='bestpath')
    parser.add_argument('--batch_size', help='Batch size.', type=int, default=100)
    parser.add_argument('--data_dir', help='Directory containing IAM dataset.', type=Path, required=False)
    parser.add_argument('--fast', help='Load samples from LMDB.', action='store_true')
    parser.add_argument('--line_mode', help='Train to read text lines instead of single words.', action='store_true')
    parser.add_argument('--img_file', help='Image used for inference.', type=Path, default='../data/sentences.png')
    parser.add_argument('--early_stopping', help='Early stopping epochs.', type=int, default=25)
    parser.add_argument('--dump', help='Dump output of NN to CSV file(s).', action='store_true')
    args = parser.parse_args()
```

Рисунок 3.1 – Вхідні аргументи застосунка

3.2.1.2 Модуль сегментації

Попередньо оброблене зображення надходить у модуль сегментації зображення на текстові зони, де здійснюється процедура морфологічного розширення і стиснення, після чого здійснюється маркування і виявлення текстових зон, далі здійснюється нормалізація всього документа за допомогою повороту зображення на обчислений середній кут.

На кожному зображенні текстової зони здійснюється процедура виділення ядра символів за допомогою умовної процедури морфологічного розширення до центральної лінії, після цього здійснюється сегментація текстової зони.

У місці поділу символів відбувається зафарбовування кольором фону кордону чарунка, параметри чарунку адаптивно підлаштовуються.

Результат зберігається в масиві зображень символів, що виділяє чарунок, та переміщається до необробленої частини текстової зони, процедура повторюється.

Клас попереднього оброблення зображення (рис. 3.2) використовується у обох застосунках.

```python
class Preprocessor:
    def __init__(self,
                 img_size: Tuple[int, int],
                 padding: int = 0,
                 dynamic_width: bool = False,
                 data_augmentation: bool = False,
                 line_mode: bool = False) -> None:...

    @staticmethod
    def _truncate_label(text: str, max_text_len: int) -> str:...

    def _simulate_text_line(self, batch: Batch) -> Batch:...

    def process_img(self, img: np.ndarray) -> np.ndarray:...

    def process_batch(self, batch: Batch) -> Batch:...

def main():...


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

Рисунок 3.2 – Опис класу попереднього оброблення зображення

3.2.1.3 Модуль створення моделі для задачі розпізнавання рукописного тексту

Система розпізнавання рукописного тексту (HTR) реалізована за допомогою TensorFlow і навчена на автономному наборі даних IAM HTR [46].

Модель отримує зображення окремих слів або текстових рядків (декілька слів) як вхідні дані та виводить розпізнаний текст. 3/4 слів із набору перевірки правильно розпізнаються, тобто приблизно 75%.

Клас моделі для задачі розпізнавання рукописного тексту (рис. 3.3) складається з 5 шарів CNN, 2 шарів RNN, рівня втрат та декодування CTC.

```python
class Model:
    """Minimalistic TF model for HTR."""

    def __init__(self,
                 char_list: List[str],
                 decoder_type: str = DecoderType.BestPath,
                 must_restore: bool = False,
                 dump: bool = False) -> None:...

    def setup_cnn(self) -> None:...

    def setup_rnn(self) -> None:...

    def setup_ctc(self) -> None:...

    def setup_tf(self) -> Tuple[tf.compat.v1.Session, tf.compat.v1.train.Saver]:...

    def to_sparse(self, texts: List[str]) -> Tuple[List[List[int]], List[int], List[int]]:...

    def decoder_output_to_text(self, ctc_output: tuple, batch_size: int) -> List[str]:...

    def train_batch(self, batch: Batch) -> float:...

    @staticmethod
    def dump_nn_output(rnn_output: np.ndarray) -> None:...

    def infer_batch(self, batch: Batch, calc_probability: bool = False, probability_of_gt: bool = False):...

    def save(self) -> None:...
```

Рисунок 3.3 – Опис класу моделі для задачі розпізнавання рукописного тексту

3.2.1.4 Модуль розпізнавання та прийняття рішень

Натренована модель приймає, як вхідне значення, відредаговану картинку слова або речення та знаходить найкращий збіг (рис. 3.4).

Рисунок 3.4 – Вихідні дані застосунку

Як вихідні дані користувач отримає масив символів та значення точності розпізнавання.

3.2.2 Застосунок, що використовує HMM, як спосіб розпізнавання рукописного тексту

3.2.2.1 Модуль організації інтерфейсу з користувачем

Модуль організації інтерфейсу з користувачем здійснює координацію роботи системи в цілому і забезпечує механізми взаємодії з користувачем. Подача вхідних даних здійснюється за допомогою вводу символів через GUI застосунку (рис. 3.5).

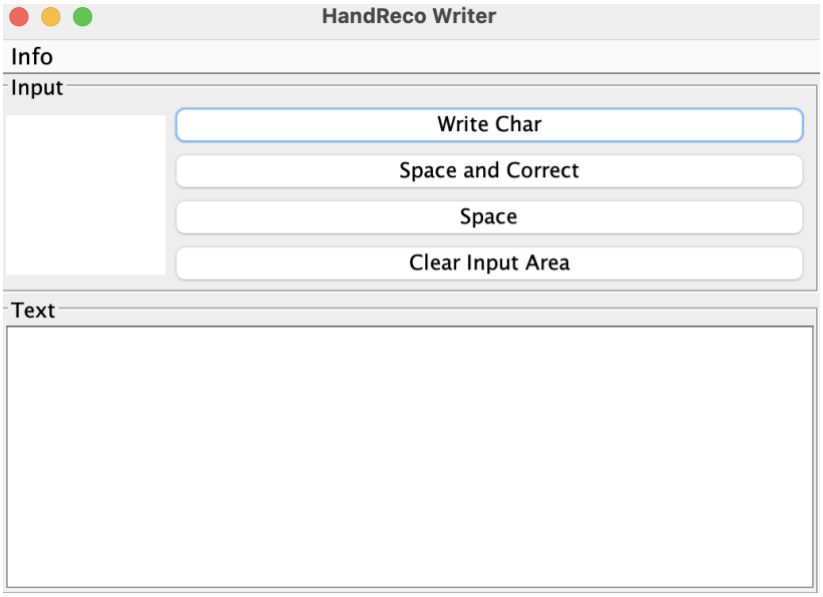


Рисунок 3.5 – Інтерфейс GUI застосунку

3.2.2.2 Модуль сегментації

Модуль сегментації зображення застосовує принципи спрощенні принципи роботи аналогічного модуля з попереднього застосунку для розпізнавання рукописного тесту за допомогою нейронної мережі (рис. 3.6).

```
class WordClassifier(object):

    def __init__(self,
                 words_with_examples=None,
                 nr_of_hmms_to_try=3,
                 fraction_of_examples_for_test=0.1,
                 train_with_examples=True,
                 initialisation_method=SpecializedHMM.InitMethod.count_based,
                 alphabet=get_example_alphabet(),
                 from_string_string=None):...

    def train(self,train_with_examples=True):...


    def create_hmm_for_word(self,
                            word,
                            training_examples,
                            test_examples,
                            nr_of_hmms_to_try,
                            train_with_examples):...


    def classify(self,string):...

    def test(self,test_examples):...

    def to_string(self):...
```

Рисунок 3.6 – Опис методу попереднього оброблення зображення

3.2.2.3 Модуль розпізнавання та прийняття рішень

Запропоновано метод об’єднання та спільної оптимізації розпізнавання та нормалізації зображення в алгоритмах оптичного розпізнавання символів на основі псевдодвовимірних HMMs.

Реалізація класу, що класифікує символи (рис. 3.7), та реалізація класу, що класифікує слова (рис. 3.8), не потребують великої обчислювальної потужності.

```
class CharacterClassifier(WordClassifier):
    """..."""


    def __init__(self,
                 characters_with_examples=None,
                 nr_of_hmms_to_try=3,
                 fraction_of_examples_for_test=0.1,
                 train_with_examples=True,
                 initialisation_method=SpecializedHMM.InitMethod.count_based,
                 feature_extractor=None,
                 from_string_string=None):...

    def classify_character_string(self,string):...

    def classify_image(self,buffered_image):...

    def test(self,test_examples):...

    def to_string(self):...
```

Рисунок 3.7 – Опис класу класифікації символу

```
import ...

def create_word_classifier(word_list, save_to_file_path):
    training_examples = generate_examples_for_words(words=word_list,
                                                    number_of_examples=800,
                                                    poelap=0.03,
                                                    poelenl=0.7,
                                                    powlap=0.1,
                                                    polmap=0.03)
    classifier = WordClassifier(training_examples,
                                nr_of_hmms_to_try=1,
                                fraction_of_examples_for_test=0,
                                train_with_examples=True,
                                initialisation_method=SpecializedHMM.InitMethod.count_based)
    classifier_string = classifier.to_string()
    file = open(save_to_file_path,'w')
    file.write(classifier_string)
    file.close()

if __name__ == '__main__':
    word_list = ["dog","cat","pig","love","hate",
                 "scala","python","summer","winter","night",
                 "daydream","nightmare","animal","happiness","sadness",
                 "tennis","feminism","fascism","socialism","capitalism"]
    create_word_classifier(word_list, "word_classifier.dat")
```

Рисунок 3.8 – Опис класу класифікації слова

Цей метод можна комбінувати з попереднім методом спільної сегментації та розпізнавання зв’язкового тексту.

Можна використовувати у модулях вищого рівня в інтелектуальній системі розпізнавання документів, як допоміжний засіб у процесі розпізнавання. Обчислювальна вартість цього методу невисока.

Вихідними даними застосунку, що використовує HMM, є слова, які збережені у масиві символів та виведені у вікно вихідних даних на GUI застосунку (рис. 3.9).

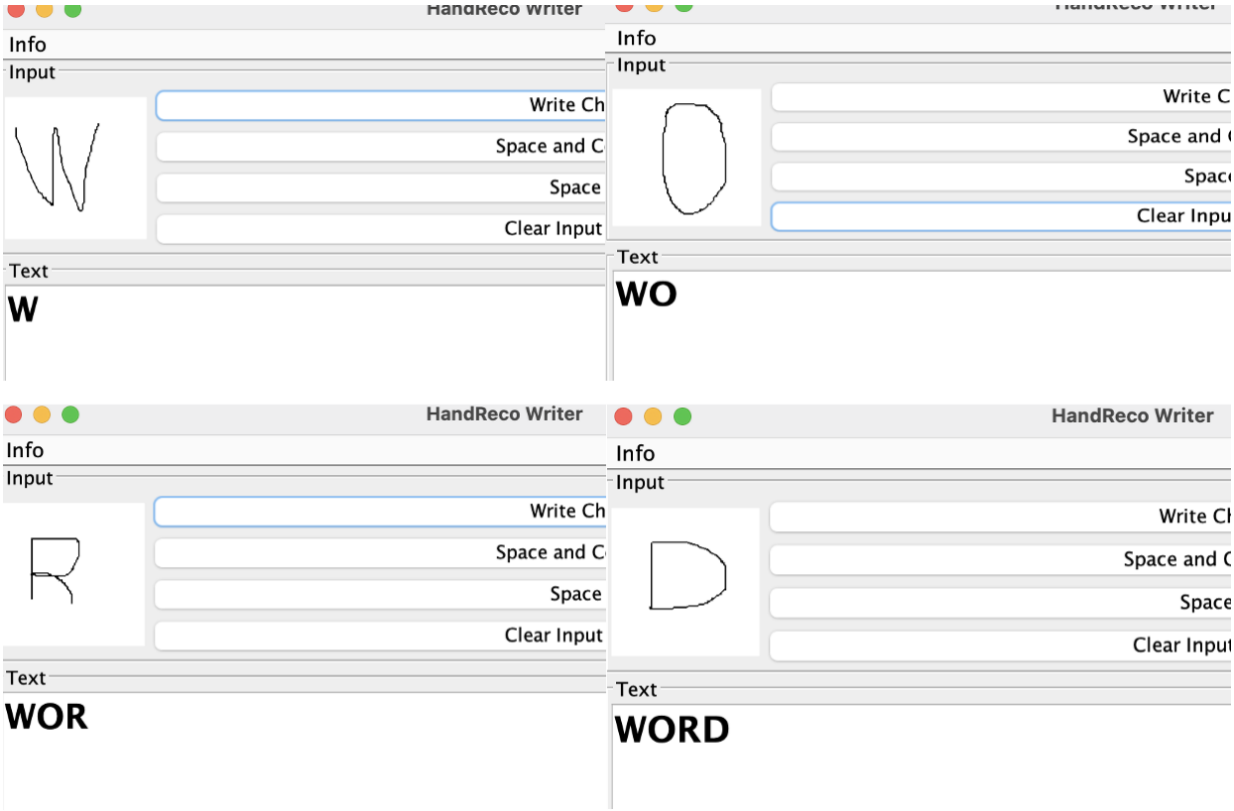


Рисунок 3.9 – Вихідні дані застосунку, що використовує HMM

3.3 Тестування розробленого застосунку та аналіз результатів

Для порівняльного тестування двох застосунків було створено тестові дані (додаток А), ураховуючи різні почерки та розмір слів та речень.

Застосовані приклади тестових (рис. 3.10) мають різний почерк та розмір символів.

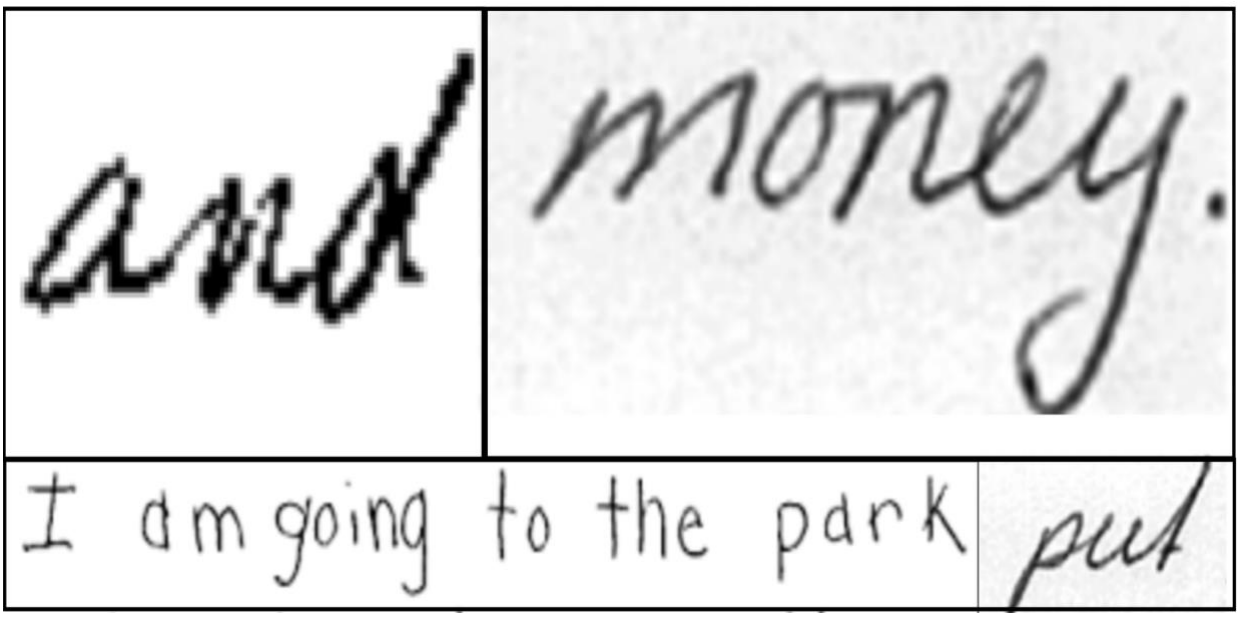


Рисунок 3.10 – Приклади тестових даних

Після проведення порівняльного тестування двох застосунків дані занесені в таблицю (табл. 3.1).

Таблиця 3.1 – Результати тестування застосунків

Тип застосунку	**Почерк 1 типу**	**Почерк 2 типу**	**Почерк 3 типу**	**Середнє значення точності розпізнавання**
Застосунок, що використовує натреновану нейронну мережу	77%	80%	71%	76%
Застосунок, що використовує HMM	49%	45%	37%	43%

Проаналізуємо результати роботи застосунків з введеними нечіткими або не існуючими у наборі даних фрагментів рукописного тексту (рис. 3.11, рис. 3.12).

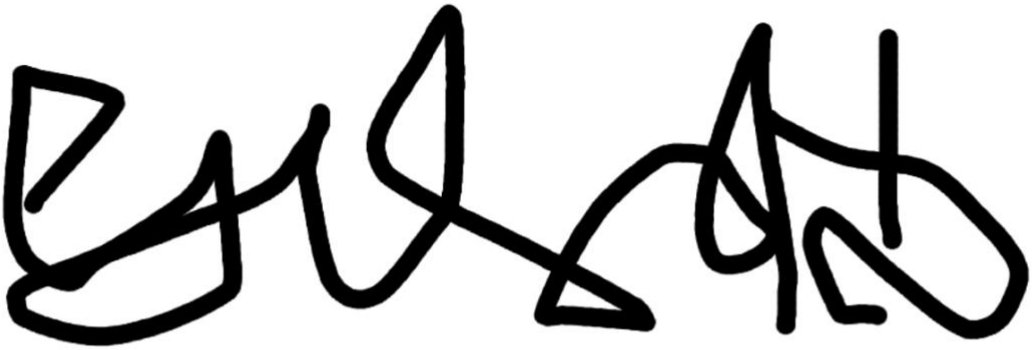


Рисунок 3.11 – Результати роботи застосунку на базі нейронної мережі

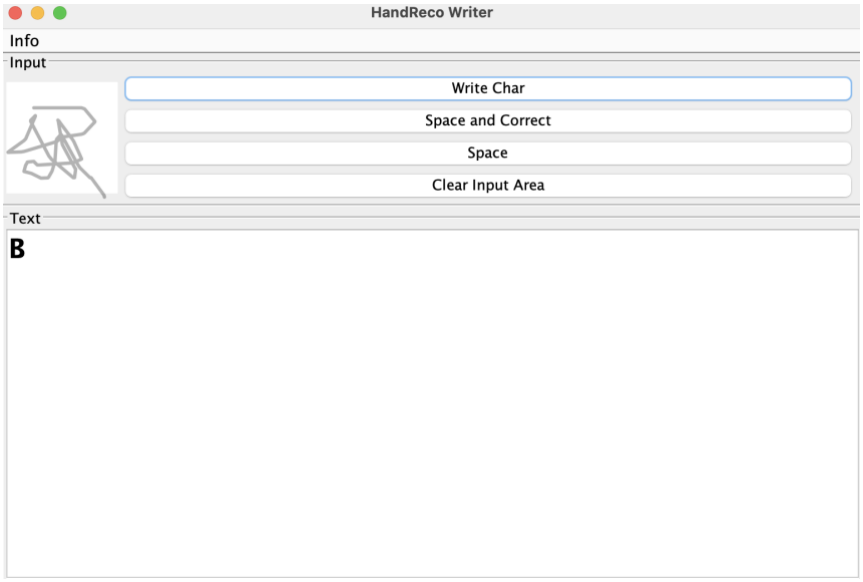


Рисунок 3.12 – Результати роботи застосунку, що використовує HMM

Результати роботи застосунку, що використовує нейронну мережу, як і результати роботи застосунку, що використовує HMM не розпізнають введені нечіткі або не існуючі у наборі даних фрагменти рукописного тексту, результат роботи з нейронною мережею визначає точність розпізнавання близькою до 0 (рис. 3.11).

Проаналізувавши результати роботи, можна впевнено сказати, що нейронні мережі впоралися із задачею розпізнавання рукописного тексту набагато краще, ніж застосунок на основі HMM.

Точність розпізнавання окремих символів вище щонайменше у 1,5 рази у порівняння з результатами HMM. Для коректної роботи моделі нейронної мережі було затрачено близько 5 годин на її навчання, значення часу може бути навіть більшим, якщо тестовий набір даних для навчання буде більший, ніж застосований у даній роботі.

3.4 Перспективи подальшої роботи

Подальші напрямки роботи пов’язані з дослідженням комбінацій методів сегментації та розпізнавання. Так, зокрема, закладений принцип послідовного розпізнавання дозволяє здійснювати попереднє розпізнавання на етапі сегментації і, таким чином, дозволяє враховувати найбільш гнучку специфіку окремих рукописних символів (наприклад, букви «ш» та «щ»).

Такий підхід дозволить застосувати різні параметри чарунку, що виділяється, для символів і скоротити кількість помилкових сегментацій.

На етапі подальшого розпізнавання доцільно застосувати більш складні граматики, побудовані на прихованих марківських моделях, що враховують сукупність всіх символів у слові, наприклад, на основі алгоритму Вітербі.

Облік результату розпізнавання на етапі сегментації символів дозволить скоротити кількість порівнянь на етапі наступного розпізнавання.

Крім того, доцільним є автоматичний вибір відповідного тематичного словника на основі слів, що розпізнаються.

Подальше застосування складніших алгоритмів, заснованих на розумінні сенсу тексту природною мовою, дозволить здійснити корекцію не тільки слів, написаних з помилками, а й словосполучень та речень загалом.

ВИСНОВКИ

У рамках кваліфікаційної роботи розроблено і реалізовано застосунки для оброблення та розпізнавання рукописного тексту із застосуванням нейронних мереж та HMM.

Виконано усі поставлені задачі для досягнення мети кваліфікаційної роботи, а саме:

– прокласифіковано існуючі методи оброблення та розпізнавання рукописного тексту. Виявлено переваги і недоліки цих методів. На основі отриманої інформації щодо існуючих методів, обрано конкретні методи для реалізації прикладних застосунків оброблення та розпізнавання рукописного тексту. Результати роботи дали уявлення про сучасну ситуацію у вирішенні проблеми оброблення рукописного тексту;

– проаналізовано сучасні системи розпізнавання рукописного тексту та нові OCR системи. Результати аналізу показали, що більшість сучасних OCR систем використовують нейронні мережі, як спосіб розпізнавання рукописного тексту;

– розроблено модель оброблення та розпізнавання рукописного тексту за допомогою HMM, яка слугує підґрунтям для програмної реалізації застосунку;

– розробити модель оброблення та розпізнавання рукописного тексту за допомогою нейронних мереж, яка слугує підґрунтям для програмної реалізації застосунку;

– програмно реалізовано розроблені моделі оброблення та розпізнавання рукописного тексту у окремих застосунках. Реалізовано GUI та CLI для введення і виведення даних. Розроблено дві програмні платформи, одна на базі HMM алгоритму, а друга – нейронної мережі;

– виконано тестування розроблених застосунків на різних наборах даних, використано різні почерки та розміри символів, введені нечіткі або не

існуючі у наборі даних фрагменти рукописного тексту. Отримані результати експериментальних досліджень показали, що точність розпізнавання рукописного тексту найвища у випадку використання нейронних мереж;

– визначено перспективу подальшої роботи, що дозволить підвищити точність розпізнавання рукописного тексту на базі створеного набору зразків.

Подальші напрямки роботи пов’язані з дослідженням комбінацій методів сегментації та розпізнавання. Подальше застосування складніших алгоритмів, заснованих на розумінні сенсу тексту природною мовою, дозволить здійснити корекцію не тільки слів, написаних з помилками, а й словосполучень та речень загалом.

Результати роботи апробовано у вигляді 2 тез доповідей під час Міжнародних науково-практичних конференцій «PROBLEMS OF MODERM SCIENCE AND PRACTICE» [47] та «TRENDS IN SCIENCE AND PRACTICE OF TODAY» [48].

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Adankon, M.M., & Cheriet, M. (2009). Model selection for the LS-SVM. Application to handwriting recognition. *Pattern Recognition*, *42*(12), 3264-3270.
2. Deng, L. (2012). The mnist database of handwritten digit images for machine learning research [best of the web]. *IEEE Signal Processing Magazine*, *29*(6), 141-142.
3. Жиляков, Е.Г., & Черноморец, А.А. (2009). Вариационные алгоритмы анализа и обработки изображений на основе частотных представлений. Белгород: Изд-во ООО ГиК, 146.
4. Zhang, W., Deng, L., Yang, L., Yang, P., Diao, D., Wang, P., & Wang, Z.L. (2020). Multilanguage-handwriting self-powered recognition based on triboelectric nanogenerator enabled machine learning. *Nano Energy*, *77*, 105174.
5. Pham, V., Bluche, T., Kermorvant, C., & Louradour, J. (2014, September). Dropout improves recurrent neural networks for handwriting recognition. In *2014 14th international conference on frontiers in handwriting recognition* (pp. 285-290). IEEE.
6. Фу, К. (1977). Структурные методы в распознавании образов.
7. C. Bahlmann, B. Haasdonk. Online handwriting recognition with support vector machines. URL: https://ieeexplore.ieee.org/abstract/document/1030883 (дата звернення 04.09.2021).
8. Мерков, А.Б. (2004). Основные методы, применяемые для распознавания рукописного текста.
9. Ott, F., Wehbi, M., Hamann, T., Barth, J., Eskofier, B., & Mutschler, C. (2020). The OnHW Dataset: Online Handwriting Recognition from IMU-Enhanced Ballpoint Pens with Machine Learning. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, *4*(3), 1-20.

10. Kienzle, W., & Chellapilla, K. (2006, June). Personalized handwriting recognition via biased regularization. In *Proceedings of the 23rd international conference on Machine learning* (pp. 457-464).

11. Python + OpenCV + Keras. URL: https://habr.com/ru/post/466565/ (дата звернення 04.09.2021).

12. Гонсалес, Р., & Вудс, Р. (2019). *Цифровая обработка изображений*. Litres.

13. Распознавание рукописных цифр на Python + GUI. URL: https://pythonru.com/primery/raspoznavanie-rukopisnyh-cifr-na-p-ython-gui (дата звернення 15.09.2021).

14. Как работает распознавание рукописного текста. URL: https://vc.ru/ml/96273-kak-rabotaet-raspoznavanie-rukopisnogo-teksta (дата звернення 15.10.2021).

15. Машинное зрение на Python. Обучаем нейросеть распознавать цифры. URL: https://medium.com/@enduranceprog/machine-vision-digits-94eb258c6ff8 (дата звернення 20.10.2021).

16. OCR-конвейер для обработки документов. URL: https://habr.com/ru/company/arcadia/blog/505950/ (дата звернення 20.09.2021).

17. ТОП-4 программ для OCR распознавания рукописного текста. URL: https://pdf.iskysoft.com/ru/ocr-pdf/handwriting-ocr.html (дата звернення 11.10.2021).

18. Распознавание рукописного текста. URL: http://idr.in.ua/info/rukopisniy-tekst.html (дата звернення 15.09.2021).

19. ОСНОВНЫЕ МЕТОДЫ ОБРАБОТКИ ИЗОБРАЖЕНИЙ ПРИ ОФФЛАЙН-РАСПОЗНАВАНИИ РУКОПИСНОГО ТЕКСТА. URL: https://science-engineering.ru/ru/article/view?id=1184 (дата звернення 15.09.2021).

20. Yousef Ibrahim Daradkeh, and Iryna Tvoroshenko (2020) Application of an Improved Formal Model of the Hybrid Development of Ontologies in Complex Information Systems, *Applied Sciences*, 10(19). p. 6777.

21. Daradkeh, Y.I., Gorokhovatskyi, V., Tvoroshenko, I., Gadetska, S., and Al-Dhaifallah, M. (2021) Methods of Classification of Images on the Basis of the Values of Statistical Distributions for the Composition of Structural Description Components, *IEEE Access*, 9, pp. 92964-92973.

22. Кучеренко, Є.І., Творошенко, І.С., Анопрієнко, Т.В. (2016) Моделювання та оцінювання станів складних об'єктів із застосуванням формальної логіки. *Системи обробки інформації*, (2), 76-82.

23. Gorokhovatskyi V.O., Tvoroshenko I.S., and Peredrii O.O. (2020) Image classification method modification based on model of logic processing of bit description weights vector, *Telecommunications and Radio Engineering*, 79(1), pp. 59-69.

24. Gorokhovatskyi V., and Tvoroshenko I. (2020) Image Classification Based on the Kohonen Network and the Data Space Modification, In *CEUR Workshop Proceedings: Computer Modeling and Intelligent Systems* (CMIS-2020), 2608, pp. 1013-1026.

25. Gorokhovatskyi V.O., Tvoroshenko I.S., and Vlasenko N.V. (2020) Using fuzzy clustering in structural methods of image classification, *Telecommunications and Radio Engineering*, 79(9), pp. 781-791.

26. Kobylin O., Gorokhovatskyi V., Tvoroshenko I., and Peredrii O. (2020) The application of non-parametric statistics methods in image classifiers based on structural description components, *Telecommunications and Radio Engineering*, 79(10), pp. 855-863.

27. Tvoroshenko I., and Tkachenko D. (2020) Mechanisms of image classification based on descriptors of local features, *Abstracts of IV International Scientific and Practical Conference «Integration of scientific bases into practice» (October 12-16, 2020)*. Stockholm, Sweden, pp. 443-448.

28. Tvoroshenko I., and Dziubenko M. (2020) Modern methods of analysis of the movement scheme using video detection of vehicles, *Abstracts of V International Scientific and Practical Conference «Study of modern problems of civilization» (October 19-23, 2020)*. Oslo, Norway, pp. 422-428.

29. Tvoroshenko, I. (2019). Development of models of spatial analysis of status of interactive processes of complex systems.

30. M. Ayaz Ahmad, Irina Tvoroshenko, Jalal Hasan Baker, Liubov Kochura, Vyacheslav Lyashenko (2020) Interactive Geoinformation Three- Dimensional Model of a Landscape Park Using Geoinformatics Tools, *International Journal on Advanced Science, Engineering and Information Technology*, 10(5), pp. 2005-2013.

31. Gorokhovatskyi, V., Rusakova, N., and Tvoroshenko, I. (2020) The application of image analysis methods and predicate logic in applied problems of magnetic monitoring, *Telecommunications and Radio Engineering*, 79(20), pp. 1801-1811.

32. Daradkeh, Y.I., Tvoroshenko, I., Gorokhovatskyi, V., Latiff, L.A., and Ahmad, N. (2021) Development of Effective Methods for Structural Image Recognition Using the Principles of Data Granulation and Apparatus of Fuzzy Logic, *IEEE Access*, 9, pp. 13417-13428.

33. Кобилін О.А., Творошенко І.С. (2021). Методи цифрової обробки зображень: навч. посібник. *Харків: ХНУРЕ*.

34. Творошенко І.С. (2021). Технології прийняття рішень в інформаційних системах: навч. посібник. *Харків: ХНУРЕ*.

35. Гороховатський В.О., Творошенко І.С. (2021). Методи інтелектуального аналізу та оброблення даних: навч. посібник.

36. Кучеренко Е.И., Филатов В.А., Творошенко И.С., Байдан Р.Н. (2005) Интеллектуальные технологии в задачах принятия решений технологических комплексов на основе нечеткой интервальной логики, *Восточно-Европейский журнал передовых технологий*, № 2. С. 92-96.

37. Кучеренко, Е.И., Творошенко, И.С. (2003) Процессы принятия решений в сложных системах на основе нечетких интервальных представлений. *Вісник Національного технічного університету «ХПІ». Тематичний випуск: Системний аналіз, управління та інформаційні технології. – Х.: НТУ «ХПІ»*, 1(7), 79-86.

38. Кучеренко, Е.И., Корниловский, А.В., Творошенко, И.С. (2010) О методах настройки функций принадлежности в нечетких системах. *Системы управления, навигации и связи*, (1), 13.

39. Творошенко, І.С., Мгеброва, В.Р., & Білий, В.В. (2016). Практичні аспекти створення вихідної інформації для проведення геоінформаційного аналізу у сфері управління нерухомістю.

40. Кучеренко, Е.И., Творошенко, И.С. (2010) Прикладные аспекты моделирования нечетких процессов в сложных системах. *Збірник наукових праць Харківського університету Повітряних сил*, (1), 127-131.

41. Бодянский, Е.В., Кучеренко, Е.И., Творошенко, И.С. (2004). О синтезе нечетких алгоритмов на основе композиции фрагментов правил и моделей. *АСУ и приборы автоматики*, (128), 19-28.

42. Творошенко, И.С., Дехтярь, А.П. (2005, June) Информационные технологии в задачах компьютерной диагностики с использованием интеллектуальных систем. In *Клиническая информатика и Телемедицина. Компьютерная Медицина–2005: материалы междунар. научн.-технич. конф., Харьков* (p. 138).

43. Творошенко, І.С., & Табашник, В.А. (2018). Розробка просторової моделі геоінформаційної підтримки людей з обмеженими можливостями, що пересуваються на інвалідних колясках, у місті Харків. *Збірник наукових праць Харківського національного університету Повітряних Сил*, (1), 122-128.

44. I. Tvoroshenko (2020). Information technologies for decision-making on the conditions of spatially distributed objects, in Abstracts of I International Scientific and Practical Conference. Problems and perspectives of modern science and practice, Austria. pp. 45-50.

45. M. Ayaz Ahmad, Volodymyr Gorokhovatskyi, Iryna Tvoroshenko, Nataliia Vlasenko, Syed Khalid Mustafa (2021) The Research of Image Classification Methods Based on the Introducing Cluster Representation Parameters for the Structural Description, *International Journal of Engineering Trends and Technology*, 69(10), pp. 186-192.

46. IAM Database. URL: https://fki.tic.heia-fr.ch/databases/iam-handwriting-database (дата звернення 29.09.2021).

47. Tvoroshenko I., and Bielinskyi Y. (2021) On the features of methods of processing and recognition of handwritten text, *Abstracts of I International scientific-practical conference «Problems of modern science and practice» (September 21-24, 2021). Boston, USA*, pp. 410-415.

48. Tvoroshenko I., and Bielinskyi Y. (2021) Features of methods of issuation of key areas and vector recognition of manuscript symbols on the image, *Abstracts of V International scientific-practical conference «Trends in science and practice of today» (October 19-22, 2021). Ankara, Turkey*, pp. 412-417.