Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

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

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

ДОСЛІДЖЕННЯ ТА РЕАЛІЗАЦІЯ МЕТОДУ РОЗПІЗНАВАННЯ КИРИЛИЦІ У ВІДЕОПОТОЦІ
(тема)

Виконав:
студент 2 курсу, групи ІТКСм-20-1
Тимофєєв О.П.
(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. Вечірська І.Д.
(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 Тимофєєву Олексію Павловичу
(прізвище, ім'я, по батькові)

1. Тема роботи Дослідження та реалізація методу розпізнавання кирилиці у відеопотоці

затверджена наказом по університету від «_22_» жовтня 2021 року № 1575Ст.

2. Термін подання студентом роботи до екзаменаційної комісії _26_ листопада 2021 р.

3. Вихідні дані до роботи *статистичні моделі обробки зображень у відеопотоці, методи оптичного розпізнавання тексту, середа розробки Android Studio, бібліотека машинного навчання ML Kit, мова програмування Kotlin.*

4. Перелік питань, що потрібно опрацювати в роботі

1. Огляд методів розпізнавання кирилиці.
2. Алгоритм пошуку та обробки зображення у відеопотоці.
3. Математичні моделі класифікації текстових областей зображення.
4. Алгоритм комбінування результатів розпізнавання з декількох зображень.

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри) актуальність проблеми обробки зображень у відеопотоці, математичне моделювання методу розпізнавання кирилиці у відеопотоці, аналіз методів оптичного розпізнавання тексту, програмна реалізація прототипу, тестові зображення.

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	23.10.21-25.10.21	
3	Аналіз літератури з досліджуваної проблеми	25.10.21-31.10.21	
4	Аналіз технічних засобів	1.11.21-2.11.21	
5	Розробка методу	3.11.21-10.11.21	
6	Програмна реалізація	10.11.21-23.11.21	
7	Оформлення пояснювальної записки	23.11.21-02.12.21	
8	Перевірка на плагіат	03.12.2021	
9	Рецензування	04.12.2021	
10	Підготовка презентації та доповіді	05.12.2021	
11	Занесення роботи в електронний архів	06.12.2021	
12	Попередній захист кваліфікаційної роботи	07.12.2021	

Дата видачі завдання 22 жовтня 2021 р.

Студент _____
(підпис)

Керівник роботи _____ доц. Вечірська І.Д.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ/ABSTRACT

Пояснювальна записка до кваліфікаційної роботи: 71 с., 1 табл., 22 рис., 40 джерел.

ДЕТЕКТУВАННЯ ЗОБРАЖЕННЯ У ВІДНОПОТОЦІ, РОЗПІЗНАВАННЯ КИРИЛИЦІ, СЕГМЕНТАЦІЯ ТЕКСТУ, МЕТОД ХАФА, ПОПЕРЕДНЯ ОБРОБКА ЗОБРАЖЕННЯ, КОМБІНУВАННЯ РЕЗУЛЬТАТІВ РОЗПІЗНАВАННЯ.

Об'єктом дослідження є алгоритми розпізнавання символів українського алфавіту, а предметом дослідження - система оптичного розпізнавання кирилиці у відеопотоці.

Метою дослідження є аналіз та реалізація методу підвищеної точності для оптичного розпізнавання кирилиці у відеопотоці за рахунок комбінування методів оптичного розпізнавання тексту, попередньої обробки зображення та аналізу отриманих результатів.

Використано методи розпізнавання тексту та захвату зображення у відеопотоці. Проведено дослідження методів пошуку текстової інформації на зображенні.

Розроблено математичну модель методу розпізнавання кирилиці у відеопотоці. Використано метод Хафа, загальну відстань Левенштейна.

У результаті роботи здійснена програмна реалізація методу розпізнавання кирилиці у відеопотоці.

IMAGE DETECTION IN STREAM FLOW, CYRILLIC RECOGNITION, TEXT SEGMENTATION, HUFA METHOD, PRELIMINARY IMAGE PROCESSING, KOMBANBANT

The object of the research are algorithms for recognizing the symbols of the Ukrainian alphabet, and the subject of the research is the system of optical recognition of the Cyrillic alphabet in the video stream.

The aim of the study is to analyze and implement a method of high accuracy for optical recognition of Cyrillic in the video stream by combining methods of optical text recognition, image pre-processing and analysis of the results.

Methods of text recognition and image capture in the video stream are used. The research of methods of search of the textual information on the image is carried out.

A mathematical model of the method of Cyrillic recognition in a video stream has been developed. The Huff method, the total Levenstein distance, is used.

As a result of work the software realization of a method of recognition of Cyrillic in a video stream is carried out.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

OCR – оптичне розпізнавання символів

ШНМ – штучна нейронна мережа

БД – база даних

БЗ – база знань

ОС – операційна система

API – Application Programming Interface

PDF – Portable Document Format

JPEG – Joint Photographic Experts Group

GIF – Graphics Interchange Format

TIFF – Tagged Image File Format

BMP – Bitmap Picture

HTML – HyperText Markup Language

BDD – Behavior-Driven Development

TDD – Test-Driven Development

ВСТУП

Однією з актуальних проблем на даний момент є неефективна робота різних державних організацій, таких як «Пенсійний Фонд», міські поліклініки, різні організації бухгалтерського обліку, оскільки все робиться в ручному режимі. Кожен бланк заповнюється власноручно, і якщо потрібно занести якусь інформацію (наприклад, паспортні дані) в базу даних, то дана задача виконується оператором власноруч.

Автоматизація процесу переведення інформації з паперових документів в електронний вигляд зараз актуальна практично у всіх державних і комерційних організаціях. Також рішення даної проблеми є важливим аспектом безпеки і контролю, адже найменша помилка при введенні даних може дорогого коштувати.

Уникнути цього можна, використовуючи програмне забезпечення, що дозволяє автоматизувати даний процес. Його можна застосовувати в різних сферах роботи, що стосуються обробки документів. Ілюстрацією вище викладеного служить практична діяльність компаній, які фіксують персональні дані громадян: комунальні служби, паспортні столи, бухгалтерія та інші. Оскільки усі державні структури використовуюсь саме кириличні символи в документації, дуже гостро стоїть питання розпізнавання кирилиці. Адже більшість закордонних систем для розпізнавання тексту дуже погано розпізнають кириличні символи, або взагалі працюють лише з символами латинського алфавіту.

Актуальність дослідження полягає у необхідності поглибити знання в галузі розпізнавання кирилиці у відеопотоці та удосконаленні алгоритмів перенесення тексту з друкованих носіїв у електронний варіант в режимі реального часу.

1 ОГЛЯД ОСНОВНИХ МЕТОДІВ ОПТИЧНОГО РОЗПІЗНАВАННЯ ТЕКСТУ

У цьому розділі розглядаються основні поняття в сфері оптичного розпізнавання символів, методи розпізнавання друкованих літер, програмні засоби для розпізнавання і додатки, здатні розпізнавати друкований текст.

1.1 Оптичне розпізнавання символів

Оптичне розпізнавання символів – це механічний або електронний переказ зображень рукописного, машинописного або друкованого тексту в послідовність кодів, що використовуються для подання в текстовому редакторі. Оптичне розпізнавання має широкі межі застосування: для конвертації книг і документів в електронний вигляд, для автоматизації систем обліку в бізнесі або для публікації тексту на веб-сторінці. Над перетвореним в електронний вигляд за допомогою оптичного розпізнавання текстом можна здійснювати множинні маніпуляції: редагувати, здійснювати пошук слова або фрази, зберігати його в більш компактній формі, демонструвати або роздруковувати матеріал, не втрачаючи якості, аналізувати інформацію, а також застосовувати до тексту електронний переказ, форматування або перетворення в мова за допомогою відповідного програмного забезпечення. Оптичне розпізнавання тексту є прогресуючою і активно досліджуваної проблемою нерозривно пов’язаною з розпізнаванням образів, мови і комп’ютерним зором [1].

Перші системи оптичного розпізнавання тексту потребували постійної калібрування для роботи з кожним конкретним шрифтом; в ранніх версіях для розпізнавання було необхідно зображення кожного символу. Програма одночасно могла працювати тільки з одним шрифтом. Не так давно широкого поширення набули «інтелектуальні» системи, які з високим ступенем

точності здатні розпізнавати більшість відомих накреслень і шрифтів. Наступним етапом стала поява систем, здатних не тільки досить з великим ступенем точності розпізнавати текст, а й відновлювати початкове форматування, включаючи колонки, таблиці та інші найпростіші графічні компоненти.

Сучасні технології дозволяють автоматизувати перетворення великої кількості паперових документів у цифровий вигляд, використовуючи сканер та подальшу цифрову обробку.

1.2 Процес розпізнавання символів

Найбільш швидким і простим способом перекладу документів є сканування паперових варіантів за допомогою сканерів або фотографування. Після обробки документа сканером виходить графічний образ (графічне зображення документа). Але графічний образ ще не є текстовим документом. Людський мозок влаштований так, що людині досить просто поглянути на аркуш паперу з текстом, щоб зрозуміти, що на ньому написано. З точки зору персонального комп'ютера, документ після сканування перетворюється в набір різнокольорових точок, а зовсім не в текстовий документ. Більш кращим, в порівнянні з графічними файлами, є текстове представлення інформації. Цей варіант дозволяє істотно прискорити і спростити процес подальшої обробки документа, скоротити витрати на зберігання і передачу отриманої інформації, а також дозволяє реалізувати всі можливі варіанти подальшого застосування і аналізу електронних документів. Але Отже, найбільший інтерес з точки зору подальшого застосування представляє саме переклад паперових носіїв в текстовий електронний документ.

Завдання розпізнавання даних з зображення можна умовно розділити на дві підзадачі: попередня обробка і власне розпізнавання даних.

Дана робота присвячена дослідженню та реалізації методу розпізнавання кирилиці у відеопотоці на прикладі застосунку для розпізнавання даних з паспортів громадян. У загальному випадку розпізнавання реалізується в три етапи:

– попередня обробка зображення;
– розпізнавання символів;
– подальший орфографічний аналіз.

Попередня обробка зображення полягає у виділенні ділянок з цікавою для нас інформацією і обробці отриманого зображення різними фільтрами з метою поліпшення якості. Під поліпшенням якості зображення передбачається: зменшення шумів на зображенні, підвищення контрастності, визначення меж символів, різні фільтрації і обертання окремих областей інтересу.

На етапі розпізнавання виділяються символи, які потім розпізнаються за допомогою одного з існуючих методів розпізнавання. Однією з основних проблем є досить низька якість картинки що надходить на вхід, саме тому велика увага приділяється попередньому етапу обробки вихідного відеопотоку та виділенню найбільш якісного кадру, адже від якості зображення безпосередньо залежить точність роботи модуля розпізнавання.

Однією з переваг запропонованого рішення є те, що розроблений алгоритм розпізнавання текстової інформації може бути дуже швидко адаптований під роботу з будь-яким типом документів (паспорт, водійське посвідчення, поліси та інші). Все що буде вимагатися: достатня кількість початкових шаблонів, з прикладами даного документа, і невелике коректування модулів з метою врахування особливостей документа, якщо такі є.

Шаблонні і структурні алгоритми розпізнавання [2, 3], що застосовуються в багатьох сучасних системах, малоефективні при досить великій кількості шумів на зображенні, а також через відсутність підтримки символів кирилиці. Тому була зроблена спроба створення подібного

програми шляхом комбінування власних методів попередньої обробки з використанням методів OCR.

На вхід системи розпізнавання надходить відеопотік, з якого фіксується найбільш якісний кадр, для подальшої сегментації та розпізнавання текстової інформації. Для якісної роботи алгоритму розпізнавання бажано, щоб зображення яке подається на вхід системи було як більш високої якості. Якщо зображення має дефекти, шуми, має низьку контрастність, погане дозвіл, то це ускладнить і суттєво знизить точність алгоритму розпізнавання. Загальний процес обробки зображення представлений на рисунку 1.1.

Рисунок 1.1 – Загальний процес обробки зображення

Враховуючи перераховані вище недоліки вхідних зображень, перед безпосереднім розпізнаванням графічного файлу алгоритмами розпізнавання, необхідно провести якнайретельнішу попередню обробку вхідного відеопотоку, спрямовану на поліпшення якості зображення. Ця процедура включає в себе ротацію зображення, перетворення на використовуваний системою формат, подальшу фільтрацію зображення від шумів, локальне підвищення різкості та контрастності необхідних ділянок. Підготовлене зображення надсилається на вхід модуля сегментації областей інтересу. Завданням даного модуля є знаходження та визначення основних структурних одиниць тексту – рядків та слів. Виділення фрагментів найвищих рівнів, таких як рядки та слова, може бути здійснено на основі

аналізу проміжків між темними областями на бінаризованому зображенні. Однак такий підхід не може бути успішно застосований для виділення окремих букв через особливості написання або різних спотворень. Зображення сусідніх літер можуть об'єднуватися в одну компоненту зв'язаності, як показано на рисунку 1.2, в даній ситуації велика ймовірність того, що «нн» буде прийнято за «м», або за «ии» [4].

Рисунок 1.2 – Об’єднання декількох букв в одну компоненту зв’язності

У багатьох випадках для вирішення задачі сегментації окремих букв використовуються евристичні алгоритми підвищеної складності, що витрачають надто великі обчислювальні потужності. Для ухвалення рішення про проходження кордону конкретної літери на даному етапі обробки системі розпізнавання буде недостатньо інформації. У розробленому алгоритмі завданням модуля сегментації лише на рівні букв буде знаходити лише можливі межі символів усередині кожної літери, а остаточне рішення про поділ слова приймається на останньому етапі роботи модуля розпізнавання, з урахуванням ідентифікації окремих фрагментів зображення як букв. Додатковою перевагою такого підходу є можливість роботи з накресленнями літер, що складаються з кількох компонентів пов'язаності завдяки розгляду таких випадків.

Внаслідок роботи модуля сегментації ми отримуємо дерево сегментації. Дерево сегментації – це особлива структура даних, організація якої відображає структуру тексту на сторінці. Найвищому рівню відповідає сам вихідний об’єкт – документ. Він містить у свою чергу масив об'єктів, що описують рядки. Кожен рядок, у свою чергу, включає свій набір об'єктів-слів. Слова є листям одержуваного нами на виході алгоритму дерева. Інформація

про можливі місця поділу слова на літери зберігається у самому слові, окремі об’єкти для літер не виділяються. У кожному об’єкті дерева зберігається інформація про область, яку займає відповідний об’єкт на зображенні.

1.3 Огляд існуючих систем розпізнавання тексту

На даний момент вже існують готові програмні рішення для розпізнавання тексту. В даному розділі будуть розглянуті найбільш поширені рішення в галузі розпізнавання тексту.

1.3.1 Abbyy FineReader

FineReader – комерційний продукт для оптичного розпізнавання символів, розроблений компанією ABBYY і безумовний лідер у сфері оптичного розпізнавання символів. В основі FineReader є запатентована технологія оптичного розпізнавання символів ABBYY OCR, яку ліцензують такі світові гіганти як Fujitsu, Panasonic, Xerox та Samsung. Програма дозволяє переводити зображення друкованих документів у безліч електронних форматів, що редагуються. Починаючи з версії 12, вона підтримує розпізнавання тексту 190 мовами, має вбудовану перевірку орфографії для 48 з них, здатна зберігати вихідне форматування документа, має великий набір функцій для попередньої обробки зображення, дозволяє налаштувати безліч параметрів. Але основною її перевагою є майже бездоганна точність розпізнавання. За деякими даними, після ретельного навчання система може розпізнавати рукописний текст, однак її потрібно буде навчати під почерком кожного конкретного користувача. На даний момент програма налічує понад 20 мільйонів користувачів у всьому світі [5].

1.3.2 CuneiForm

CuneiForm – система оптичного розпізнавання текстів компанії CognitiveTechnologies. На даний момент CuneiForm позиціонується як шрифтонезалежна система перетворення електронних копій паперових документів та графічних файлів у редагований вигляд з можливістю збереження структури та гарнітури шрифтів оригінального документа в автоматичному або напівавтоматичному режимі. Система включає дві програми для одиночної і пакетної обробки електронних документів [6].

1.3.3 OCRopus

OCRopus – система оптичного розпізнавання символів, спочатку спрямовану перетворення в електронний вигляд великого обсягу документів з урахуванням власного розпізнавального ядра, починаючи з версії 0.4. Це програмний пакет для розпізнавання тексту, що розвивається за принципами OpenSource, що розповсюджується під ліцензією ApacheLicense 2.0. Програма дозволяє підключати додаткові модулі для аналізу макету вмісту, попереднього покращення зображення. За задумом розробників, за допомогою OCRopus стане можливим визначати текстовий вміст на графічних файлах рядково і переводити його у звичайний текстовий формат для подальшого редагування. Крім друкованого тексту програма зможе розпізнавати і рукописні матеріали. OCRopus в даний час доступна лише для GNU/Linux. В даний час OCRopus використовує тільки інтерфейс командного рядка, приймаючи вказівки на вхідні зображення з текстом, та виводячи дані у форматі hOCR (відкритий формат на основі HTML), TXT [7].

1.3.4 FreeOCR

FreeOCR – безкоштовна програма для розпізнавання відсканованого тексту. Вона працює не тільки с файлами зображень, але і з pdf файлами і безпосередньо зі сканером. Для сканування необхідно тільки мати підключений сканер. Основним плюсом програми є її повна автоматизація і відсутність будь-яких налаштувань. Підтримуються формати JPEG, GIF, TIFF BMP і PDF. Також, існує ліміт на 10 зображень У годину. Стверджується, що система здатна розпізнавати більшість східно-європейських мов, в тому числі українську.

1.4 Недоліки існуючих рішень

Наведені систем оптичного розпізнавання тексту є найбільш поширеними та мають досить великий спектр використання. При аналізі існуючих рішень були виділені наступні недоліки:

– багато з систем мають не зовсім інтуїтивно зрозумілий інтерфейс і занадто велику кількість функцій, що може виявитися недоліком;

– не всі мають можливість попередньої обробки зображення з метою поліпшення якості;

– відсутність можливості розпізнавання тексту у відеопоці в режимі реального часу;

– довгий час обробки тексту та відсутність можливості використання на малопродуктивних комп’ютерах та телефонах.

Більшість систем OCR показують найкращі результати при обробці чорного тексту світлому фоні. Що стосується текстових документів – це критично. Але якщо застосовувати таку систему для обробки нестандартних зображень: паспортів, посвідчень водія, номерних знаків автомобілів, то ми

зіткнемося з неможливістю отримати коректні дані без попередньої обробки зображення.

Багато проблем, описаних вище, можна усунути на етапі попередньої обробки зображення.

Метод оптичного розпізнавання тексту у відеопотоці повинен:

- виділяти на цифровому зображенні текстові області;
- виділяти в них окремі рядки, потім окремі символи;

– розпізнавати ці символи і бути нечутливою (стійкою) по відношенню до способу верстки, відстані між рядками та іншим параметрам друку.

1.5 Методи розпізнавання символів

Розпізнавання структурованих (друкованих) символів різних зображень забезпечує вирішення низки наукових та прикладних завдань під час ідентифікації об'єктів різної природи. Сучасні методи розпізнавання символів використовуються вирішення як типових завдань, наприклад, розпізнавання тексту, і спеціалізованих завдань, орієнтованих розпізнавання символьної інформації, нанесеної на поверхню різних об'єктів. Розглянемо найвідоміші та найпоширеніші методи розпізнавання символів.

1.5.1 Структурні методи

Алгоритм роботи даних методів полягає у розбитті вихідного образу на складові, що описуються як стабільні ідеальні елементи.

Одні з методів даного напрямку використовують для розпізнавання топологічний опис зображень. Іншими словами, стандарт містить інформацію про взаємне становище окремих складових елементів знаку. При цьому стає неважливим розмір літери, що розпізнається, і навіть шрифт, яким

вона надрукована. Можливі зображення, що становлять той чи інший клас, можна уявити як результат гомеоморфних перетворень деякого еталонного зображення, яке відповідає цьому класу [2].

Завдання розпізнавання може бути зведено до встановлення гомеоморфності пред'явленого зображення з одним з еталонних. Його можна виявити за допомогою топологічних інваріантів – таких властивостей зображення, які не змінюються за його гомеоморфних перетворень. Інваріантом, що дозволяє дати чисельний опис зображень, є, наприклад, кількість ліній, що сходяться в точці. Відповідний опис виходить обходом у порядку контурів зображення з одночасною фіксацією індексів точок.

Встановлення гомеоморфності (власне розпізнавання) – зводиться до порівняння описів пред'явленого зображення та еталонних зображень класів. Важлива перевага топологічного опису – його нечутливість до сильних деформацій зображення, що включає всі перетворення подібності, якщо пов'язувати з кожним зображенням певну характерну точку, з якої починається обхід. Навчання топологічного коду полягає у веденні набору еталонних описів.

Інші структурні методи реалізують алгоритм подієвого розпізнавання. Подієвий метод спирається на топологічну структуру об'єкта, що складається з ліній і не змінюється при малих деформаціях образу. Лінією називається частина образу, у кожному перерізі якого є лише один інтервал. Лінії, що огрублені на деякій сітці, визначають події. Подієве уявлення є не лише формальним набором ознак, а й адекватним топологічним описом. Навчання методу зводиться до складання списку еталонів досить великий послідовності образів.

Основною проблемою структурних методів розпізнавання є ідентифікація знаків, що мають дефекти.

Перевагами методу є здатність розпізнавання спотворених символів та швидкодія при малому алфавіті.

1.5.2 Шаблонні методи

Алгоритм роботи шаблонних методів спирається на зіставлення вхідного графічного зображення, ідеального шаблону. Першим етапом роботи шаблонного методу є перетворення отриманого зображення на растрове. У процесі розпізнавання перебираються шаблони і обчислюється відстань від образу до шаблону. Клас, шаблони якого знаходяться на мінімальній відстані від вхідного образу є результатом розпізнавання.

Дані методи поділяються на два класи шрифтозалежні та шрифтонезавізні. Шрифтонезалежні методи використовують заздалегідь певні шаблони, і універсальні для всіх типів шрифтів. Однак за такого підходу знижується вірогідність правильного розпізнавання. Шрифтозалежні алгоритми розраховані тільки на один тип шрифту, це підвищує якість їх роботи, але вони зовсім не працездатні при використанні інших шрифтів. Були запропоновані методи розпізнавання великих обсягів тексту, коли частина символів гарантовано розпізнається шрифтонезалежними методами, а потім на підставі розпізнаних символів будуються шаблони для шрифтозалежних алгоритмів [8].

При великій кількості друкованої продукції в процесі навчання неможливо охопити всі шрифти та їх модифікації.

До переваг цього алгоритму відносяться:

- простота реалізації;
- надійна робота в умовах відсутності перешкод;
- висока точність розпізнавання дефектних символів;
- швидкість при малому алфавіті.

До недоліків можна віднести:

- залежність від шаблонів та складність підбору оптимальних шаблонів;
- неможливість розпізнати шрифт, що відрізняється від закладеного в систему;

– повільна робота при великій кількості перешкод;
– чутливість до обертання, шумів та спотворень.

1.5.3 Ознакові методи

Дані методи базуються на тому, що зображенню ставиться у відповідність N-вимірний вектор ознак. Розпізнавання полягає у порівнянні його з набором еталонних векторів тієї ж розмірності. Прийняття рішення про належність образу тому чи іншому класу, виходячи з аналізу обчислених ознак, має низку суворих математичних рішень у межах імовірнісного підходу. Тип і кількість ознак значною мірою визначають якість розпізнавання. Формування вектору відбувається під час аналізу зображення. Цю процедуру називають вилученням ознак. Еталон для кожного класу одержують шляхом аналогічної обробки символів навчальної вибірки.

До переваг методу можна віднести:

– простота реалізації;
– висока узагальнююча здатність;
– стійкість до зміни форми символів;
– висока швидкодія.

До недоліків методу відносяться:

– нестійкість до різних дефектів зображення;
– втрата інформації про символ на етапі отримання ознак.

1.5.4 Методи засновані на використанні нейронних мереж

Ідея цих методів – моделювання роботи мозку людини. На вхід заздалегідь навченої нейронної мережі надходить вектор, який є уявленням вхідного образу (пікселі, частотні характеристики, вейвлети). На виході

нейрон, який відповідає класу розпізнаного символу, видає максимальне значення функції активації. Або на вихід надходить безліч ключових характеристик зображення, які потім обробляються іншими системами. Навчання нейронних мереж відбувається на багатьох навчальних прикладах. Причому можливе навчання з учителем (персептрон) чи самоорганізація (мережа Кохонена) [4].

На рисунку 1.3 [4], як приклад схематично показана двошарова нейронна мережа, що включає 35 входів (кожний символ – матриця 7×5, відповідно, вектор, що описує матрицю, складається з 35 елементів), 26 виходів (кількість літер) і 10 нейронів прихованого шару. Як функцію активації в цій мережі використовується сигмоїдна функція, вихід якої представлений в діапазоні від 0 до 1, що потім зручно перевести в булеву алгебру.

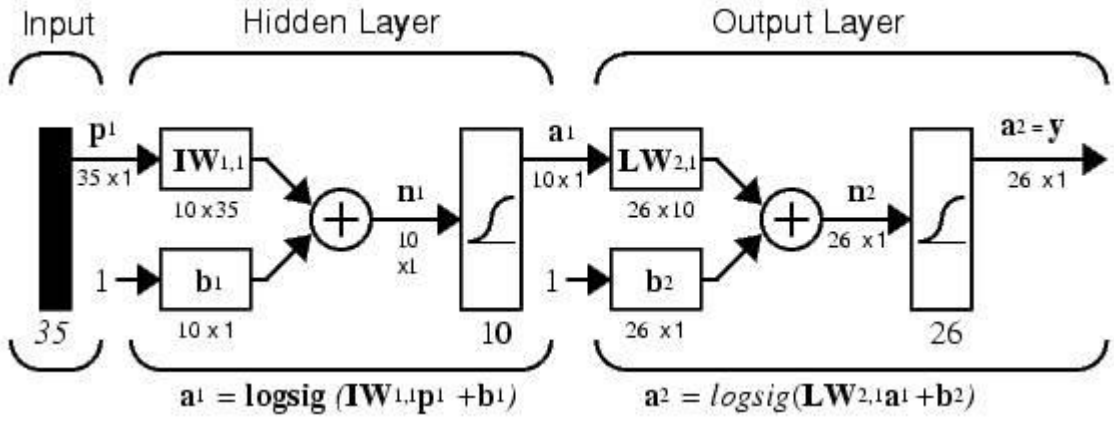


Рисунок 1.3 – Приклад двошарової нейронної мережі

Перевагами методу є:

- здатність до узагальнення;
- висока швидкість роботи.

Недоліки:

- чутливість до обертання та спотворення символів;
- складність підбору навчальної вибірки та алгоритму навчання.

1.6 Особливості розпізнавання кирилиці

Поточна ситуація на ринку така, що сервіси для розпізнавання тексту мають здебільше закордонне походження. Це означає, що вони акцентовані на латиницю і погано визначають кириличні символи.

На сьогоднішній день латиниця та кирилиця вважаються основними формами писемності на землі. Алфавіти дуже схожі між собою, мають ряд однакових літер, але в кирилиці є необхідні нам літери, для позначення шиплячих звуків «Ш» і «Щ». Саме на цих літерах та особливостях граматичних особливостях написання різних слів, потрібно сконцентрувати основну уваги під час навчання та тестування методу розпізнавання кирилиці у відеопотоці. Наприклад, українській мові досить часто зустрічаються слова з дубльованою «нн», яка при поганій якості зображення може сприйматися системой як буква «м».

Під час навчання методу розпізнавання кирилиці у відеопотоці повинна використовуватися вибірка кириличних символів (рис. 1.4) та типові слова, які демонструють ті чи інші граматичні конструкції. Увагу потрібно сконцентрувати на наступних символах: «Й», «Ш», «Щ», «Ь», «Ґ», «Ї», «Ф», «Ц».

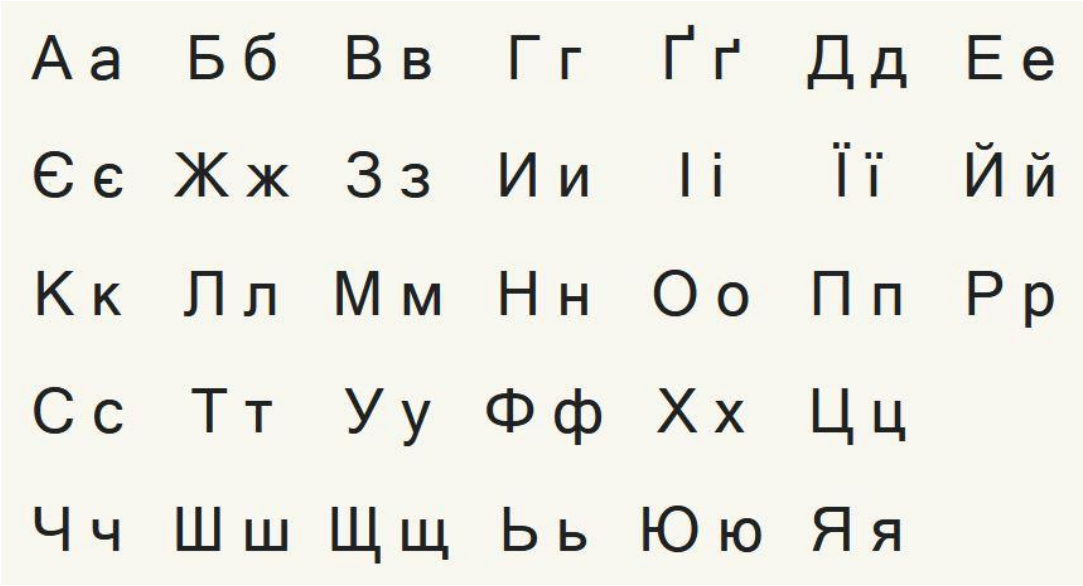


Рисунок 1.4 – Символи кирилиці

1.7 Постановка задачі дослідження

Розглянувши найбільш поширені системи розпізнавання тексту та основні алгоритми виявлення тексту на фото чи у відеопотоці, для подальшої обробки, можна висунути ряд вимог для реалізації методу розпізнавання кирилиці у відеопотоці.

Оскільки розробляємий метод націлений перш за все на мобільний сегмент електронних пристроїв, адже повинен розпізнавати текстову інформацію у відеопотоці в режимі реального часу, найважливішим є швидкість роботи алгоритму обробки зображення та розпізнавання тексту. Також дуже важливим є точність розпізнавання символів кирилиці та можливість розпізнавання тексту за умови великої кількості шумів в кадрі.

Основною задачею є створення швидкого, точного та універсального методу розпізнавання кирилиці у відеопотоці, завдяки глибокому дослідженню існуючих методів захвату кадру, обробки зображення та розпізнавання тексту.

Метою даної роботі є дослідження та реалізація методу підвищеної точності для оптичного розпізнавання кирилиці у відеопотоці за рахунок комбінування методів оптичного розпізнавання тексту, попередньої обробки зображення та аналізу отриманих результатів.

Об’єктом дослідження в даному випадку є алгоритми розпізнавання символів, а предметом дослідження – система оптичного розпізнавання символів з зображень.

Для досягнення мети необхідно вирішити такі завдання:

– провести поглиблений аналіз існуючих методів детектування та захвату зображення у відеопотоці;

– провести дослідження методів розпізнавання тексту на зображенні;

– реалізувати метод розпізнавання кирилиці у відеопотоці;

– реалізувати прототип системи розпізнавання кирилиці з використанням створеного методу.

2 МОДЕЛЮВАННЯ МЕТОДУ РОЗПІЗНАВАННЯ КИРИЛИЦІ У ВІДЕОПОТОЦІ

2.1 Загальна модель роботы методів розпізнавання тексту

При первинному розгляді методів розпізнавання текстів можна виділити загальний алгоритм їх роботи, що складається з наступних етапів:

Етап 1. Пошук області, що містить текст, його локалізація;

Етап 2. Попереднє покращення якості локалізованої області, її бінаризація;

Етап 3. Виявлення структури знайденого блоку тексту, визначення порядку читання;

Етап 4. Сегментація тексту на рядки/слова та символи;

Етап 5. Отримання ознакового опису кожного символу;

Етап 6. Розпізнавання окремих символів;

Етап 7. Словникова перевірка.

Першим етапом є пошук областей тексту, в результаті якого на вихідному зображенні виділяються текстові області. Для вирішення цього завдання застосовуються згорткові нейронні мережі [9], детектор Віоли-Джонса, методи класифікації на основі аналізу текстурних ознак [10], ознак Томура, HOG-дескрипторів. Варто відмітити що сучасні OCR-системи погано справляються із завданням детектування тексту на довільних зображеннях, оскільки спочатку розроблялися для роботи із зображеннями, що містять переважно однорідний структурований текст, що рідко зустрічається на фотографіях та відеозаписах у реальних варіантах використання.

Алгоритми покращення якості припускають попереднє згладжування зображення, визначення типів присутніх шумів та їх усунення.

Сегментація полягає у розбиття тексту на рядки, слова та символи. Пошук рядків, як правило, ґрунтується на періодичності та регулярності текстових областей та здійснюється на основі методу Хафа, методу

зв’язкових компонентів [11], аналізу горизонтальних, вертикальних та діагональних гістограм.

Алгоритми отримання ознакового опису символів поділяються на 2 класи:

– аналізуючи вихідне зображення символу в якості ознаки;

– аналізуючи обчислювані ознаки, такі як довжина хорди [12], апроксимовані контури, кістяки символів.

Для зниження розмірності простору ознак часто застосовується метод головних компонентів або лінійний дискримінантний аналіз.

Для розпізнавання символів реалізуються різні класифікатори, засновані на нейронних мережах [13] і найпоширеніших нині згорткових нейронних мережах [1, 9].

Словникова перевірка здійснюється на основі стандартних чи динамічно створених мовних словників, *N*-грам, реалізованих у вигляді списків, дерев чи графів [10, 12].

Сучасні алгоритми спрямовані на вирішення наступних завдань: пошук та локалізація тексту, визначення структури тексту, визначення кута нахилу текстових рядків.

Завдання визначення структури документа, як правило, націлена на виділення однорідних блоків, таких як текст, зображення, графік, таблиця та інші.

Алгоритми класифікації блоків на текст та не текст дозволяють визначити, чи є виділений блок текстом, зображенням або графіком, таблицею та ін.

Оскільки в даній роботі розглядається розпізнавання кирилиці саме у відеопотоці, також потрібно розглянути та проаналізувати задачу ідентифікації і захвату зображення у відеопотоці. Тобто алгоритм розпізнавання кирилиці у відеопотоці має наступну послідовність дій, забражених на рисунку 2.1.

Рисунок 2.1 – Загальний алгоритм роботи методу розпізнавання кирилиці у відеопотоці

Далі детальніше проаналізуємо виділені етапи роботу методу.

2.2 Визначення об'єктів першого плану

Методи визначення структури документа виділяють області, що містять об'єкти першого плану (текст, зображення або графік), від фону на

вхідному зображенні. Усі методи визначення об'єктів першого плану можна розділити на 2 категорії: методи, що здійснюють аналіз знизу-вгору та зверху-вниз [14].

У методах, що здійснюють аналіз зверху-вниз, вихідне зображення ітеративно розбивається окремі блоки. У методах, що здійснюють аналіз знизу-вгору спочатку виділяються найменші можливі компоненти, які згодом групуються в блоки. Існують також і гібридні підходи, що комбінують обидва ці підходи.

Методи сегментації сторінки можуть базуватися на аналізі проекцій [15], методі зв'язкових компонентів [16], аналізі пробільних прямокутників [17], аналіз контурів [18] або текстури.

2.2.1 Алгоритм аналізу проекцій

Одним із перших було запропоновано алгоритм аналізу проекцій. Його найбільш швидка версія працює з RLE-поданням вихідного бінарного зображення і складається з наступних етапів [18]:

Етап 1. Отримання RLE-подання зображення.

Етап 2. Видалення білих RLE-послідовностей, довжина яких менша за поріг T1.

Етап 3. Поворот вихідного зображення на 90°. Повторення кроків 1, 2 порогом T2.

Етап 4. Застосування логічної операції "І" над зображеннями, отриманими на кроці 2 та 3.

Етап 5. Видалення білих RLE-послідовностей, довжина яких менша за T3 на зображенні, отриманому на кроці 4 (T3 < T1).

Етап 6. Застосування методу зв’язкових компонентів.

Етап 7. Зіставлення площі та висоти кожної знайденої області з константами для визначення текстових та нетекстових областей.

Перевагою алгоритму є його простота та висока швидкість роботи за рахунок попереднього стиснення алгоритмом RLE. Складність алгоритму є лінійною O(N). Одним із недоліків є необхідність вирішення питання про спосіб його бінаризації. Крім того, якщо малюнок або області тексту розташовані досить близько, то висока ймовірність їхнього злиття.

2.2.2 Рекурсивний алгоритм X-Y cut

Рекурсивний алгоритм X-Y cut, вперше описаний в 1984 році [19], відноситься до алгоритмів, що працюють зверху-вниз. В результаті його роботи сегментована сторінка подається у вигляді дерева. На кожному етапі сторінка рекурсивно розбивається на дві або більше прямокутні області, що формують вузли дерева, згідно з наступним алгоритмом:

Крок 1. видалення дрібного шуму із зображення.

Крок 2. виділення зв’язкових компонентів.

Крок 3. обчислення середньої висоти та ширини символу.

Крок 4. пошук можливості поділу блоку вертикальним або горизонтальним розрізом (можливість визначається на основі аналізу побудованих горизонтальних та вертикальних гістограм блоку за чорними та білими пікселями).

Крок 5. поки є можливість поділу, повторити крок 4 для кожного з отриманих блоків.

Найбільш складними етапами даного алгоритму є пошук зв'язкових компонентів і рекурсивне поділ блоків. Реалізація двопрохідного методу зв'язкових компонентів має складність O(N). Рекурсивний поділ на блоки аналогічний до побудови бінарного дерева зі складністю O(N). Дані етапи виконуються послідовно, отже, загальна складність алгоритму дорівнює O(N). Основним недоліком даного алгоритму є неможливість здійснення

сегментації у разі, якщо ні одна з розділових смуг не поділяє сторінку повністю по горизонталі чи вертикалі.

2.2.3 Алгоритм пошуку максимальних білих прямокутників

У цьому алгоритмі спочатку передбачається наявність блокової структури тексту, де текстові області розділені прямокутними перегородками. Алгоритм складається з двох основних етапів: пошук можливих білих прямокутників та вибір максимальних у тому числі. Алгоритм пошуку білих багатокутників на зображенні документа вперше був запропонований у [20] та адаптований для зображень під нахилом у роботі [21].

На вхід алгоритму подається набір чорних прямокутників, що містять зв'язкові компоненти символів документа, попередньо підданого високочастотної фільтрації. Чорні прямокутники сортуються за рядками та стовпцями, після чого обчислюється попарна відстань між сусідніми них. Найбільший білий прямокутник, обмежений знайденими зв'язковими компонентами, визначається кількістю зв'язкових чорних компонент, що стикаються з його довгою стороною [22].

Далі з усіх знайдених білих прямокутників визначаються максимальні. Білий прямокутник вважається максимальним, якщо він не може бути більше розширено, тобто його межі стикаються зі знайденими з'язковими компонентами або досягають краю документа.

Нехай *Q(r)* – ознака якості для поточного прямокутника *r*, його значення має задовольняти наступній умові: якщо $r1 \subseteq r2$, то $Q(r1) \geq Q(r2)$ [24]. У якості подібних ознак можна взяти площу, периметр або суму квадратів висоти та ширини прямокутника.

У роботі для оцінки якості було запропоновано таку формулу (2.1):

$$Q(r) = \sqrt{area(r)W(|\log_2(height(r)/width(r)|))}\,, \qquad (2.1)$$

де W – оцінка пропорційності.

У роботі пропонується така функція для оцінки пропорційності (2.2):

$$W(x) = \begin{cases} 0,5, & x < 3, \\ 1,5, & 3 \le x < 5, \\ 1, & \textit{інакше.} \end{cases} \qquad (2.2)$$

Таким чином, завдання пошуку максимального прямокутника зводиться до пошуку прямокутника, який не перекриває жодні інші прямокутники, функція якості якого є максимальною. Алгоритм виконуэться за наступною схемою:

Крок 1. У чергу заноситься вихідне зображення.

Крок 2. Здійснюється пошук максимальних прямокутників усередині сторінки.

Крок 3. Створюється черга з пріоритетами прямокутників (першими зберігаються прямокутники з найбільшим пріоритетом).

Крок 4. Далі у циклі:

Крок 4.1. Береться перший прямокутник із черги.

Крок 4.2. Якщо він порожній (не містить вкладених прямокутників), то один із максимальних білих прямокутників знайдений.

Крок 4.3. Якщо ні, то із зв'язкових областей у досліджуваному прямокутнику вибирається один "опорний" прямокутник, розташований ближче до центру прямокутника-батька.

Крок 5. У разі перетину опорного прямокутника з іншими обмежується його висота та ширина.

Крок 6. Прямокутники, з якими він перетинається, додаються до черги.

Умовою виходу з циклу може бути або достатня кількість знайдених прямокутників, або якість знайдених прямокутників.

Решта зв’язкових областей на зображенні надалі класифікуються на текстові та нетекстові.

До недоліків алгоритму відноситься неможливість його роботи із зображеннями, що містять маленькі або нерегулярні блокові роздільники, а також складність алгоритму *O(N2)*. Додаткову складність додає пошук прямокутників на зображеннях під нахилом.

2.3 Класифікація текстових та не текстових областей зображення

Виділені на попередньому етапі області можуть являти собою текст, горизонтальні та вертикальні лінії, малюнок або графік. Важливим чинником класифікації областей є вибір ознакового простору. Усі існуючі методи класифікації можна розділити на 2 категорії:

– методи, що ґрунтуються на статистичних ознаках;

– методи, засновані на спектральних ознаках (перетворення Фур’є, вейвлет-аналіз, фільтри Габора).

Для класифікації сегментованих блоків часто застосовуються лінійні класифікатори [23], дерева рішень [24] чи нейронні мережі [7, 12].

2.3.1 Алгоритми засновані на аналізі статичних ознак

Найпростіший варіант алгоритму, заснований на аналізі статистичних ознак [25], складається з трьох основних кроків:

Крок 1. Пошук рядків на основі аналізу координат знайдених зв’язкових компонентів.

Крок 2. Визначення щільності символів у рядку (якщо d > 80%, то це строка):

$$d = \sum w_{ch} / l_{str}, \tag{2.3}$$

де w_{ch} – ширина поточного символу рядка;

l_{str} – довжина всього рядка, підсумовування ведеться по всіх символах, що входять до поточного рядка.

Крок 3. Якщо кількість рядків в області перевищує 50%, то область класифікується як текстова.

Передбачається, що блок може являти собою рядок тексту, горизонтальний або вертикальну лінію, малюнок чи графік. Класифікація здійснюється на основі 10 числових ознак, розрахованих для кожного блоку:

– висота $(\triangle y)$;

– довжина $(\triangle x)$;

– площа $(\triangle y \times \triangle x)$;

– ексцентриситет $(\triangle y, / \triangle x)$;

– загальна кількість чорних пікселів на зображенні із зменшеним дозволом;

– загальна кількість чорних пікселів у сегментованому блоці;

– число переходів від білого до чорного на зображенні зі зменшеною роздільною здатністю;

– відсоток чорних пікселів на всьому зображенні із зменшеним дозволом;

– відсоток чорних пікселів у сегментованому блоці;

– середня довжина рядків на всьому зображенні з зменшеним дозволом.

Одним із основних недоліків описаного методу є неможливість правильно класифікувати області тексту, що містять більше одного рядка [26].

За додаткового аналізу текстурних ознак області можна успішно аналізувати блоки будь-якого розміру (що включають кілька рядків). Підхід,

запропонований в [27], передбачає побудову двох додаткових матриць: BW (оцінка переходу чорно-білий), BWB (оцінка переходу чорно-білий) на основі RLE-зображення. Чорно-білий перехід (BW) – це набір пікселів, що складається з безлічі чорних пікселів, за якими слідує безліч білих (рис. 2.2). Довжина переходу дорівнює сумарній кількості чорних та білих пікселів у ньому [28].

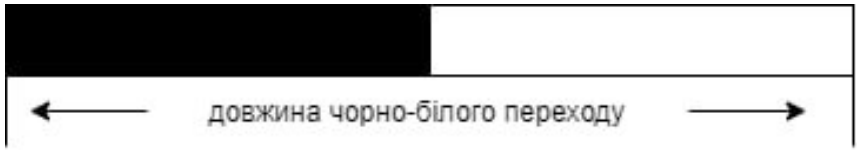


Рисунок 2.2 – Приклад переходу чорний-білий

Виділяють 9 категорій різних переходів (відповідно до процентного співвідношення в них чорних і білих пікселів). Елемент матриці BW(*i, j*) визначає кількість переходів, що потрапляють до категорії *i*, довжина яких дорівнює *j*.

Матриця BWB служить для оцінки розподілу білі проміжки між чорними. Довжина переходу розраховується як довжина білого проміжку у переході. Усі знайдені переходи квантуються на 3 категорії, виходячи із відсоткового співвідношення чорних проміжків. Елемент матриці BWB(i, j) визначає кількість переходів, які у категорію i, довжина білої частини якого дорівнює j (рис. 2.3).

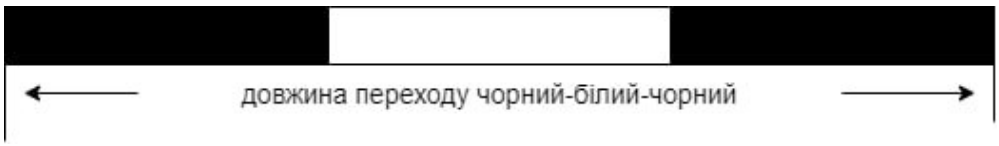


Рисунок 2.3 – Приклад переходу чорний-білий-чорний

З таблиці BW розраховуються ознаки F_1 та F_2 :

$$F_1 = \sum_{i=1}^{Nc}\sum_{j=1}^{Nr}\left[\frac{BW(i,j)}{j^2}\right] / \sum_{i=1}^{Nc}\sum_{j=1}^{Nr} BW(i,j), \quad (2.4)$$

$$F_2 = \sum_{i=1}^{Nc}\sum_{j=1}^{Nr}\left[j^2 \cdot BW(i,j)\right] / \sum_{i=1}^{Nc}\sum_{j=1}^{Nr} BW(i,j), \quad (2.5)$$

З таблиці BWB розраховується ознака F_3:

$$F_3 = \sum_{j=1}^{Nr} j^2 \left[\sum_{i=1}^{Nc} p`(i,j)\right] / \sum_{i=T_1}^{Nr}\sum_{i=1}^{Nc} p`(i,j), \quad (2.6)$$

де

$$\begin{cases} BWB(i,j), \ BWB(i,j) > T_2, \\ 0, \ BWB(i,j) < T_2. \end{cases} \quad (2.7)$$

Nc – кількість різних переходів (стовпців матриць BW та BWB відповідно);
Nr – кількість різних довжин переходів (рядок матриць BW та BWB відповідно).

Поріг T_1 відсікає короткі переходи, т.к. наявність довгих переходів необхідно визначення графічних блоків. Поріг T_2 необхідний видалення невеликих (випадкових) значень BWB(i,j) довгих переходів. Експериментально було отримано значення, $T_1 = 50$, $T_2 = 50$. Отриманий тривимірний вектор (F_1, F_2, F_3) подається на вхід лінійного класифікатора з метою класифікації блоку на 5 класів: текстова область з маленькими, середніми або великими символами, графіком або напівтоновим зображенням [28].

Схожий метод, що полягає у розбитті бінарного зображення на квадратні області (10×10, 20×20 в залежності від розміру вихідного зображення), був запропонований [29]. У кожному вікні розраховуються такі

параметри: співвідношення чорних та білих пікселів, середня довжина чорних проміжків, крос-кореляція між послідовними вертикальними та горизонтальними рядками, розташованими на рівних проміжках друг від друга. Складність алгоритму дорівнює *O(N)*.

Як ознаки можна аналізувати гістограми, оцінюючи частку світлих пікселів в області [30]. Метод успішно працює для відділення графіків та тексту від зображень, однак документ може містити графіки з великою площею білого тла, що створює труднощі при розподілі графіка та тексту. Додатково як ознаку можна враховувати горизонтальні та вертикальні проекції блоку [31].

2.3.2 Алгоритми, засновані на спектральних ознаках

Альтернативний підхід ґрунтується на визначенні спектральних ознак сегментованої галузі [25]. Спектральні ознаки описуються через розкладання зображення за набором базисів. До них відносяться косинусне розкладання, синусне розкладання вейвлет-перетворення Хаара, Добеші, фільтри Габора. У процесі отримання ознак вибирається одне з розкладів зображення як двовимірної функції, коефіцієнти розкладання використовуються як ознаки, що характеризують текстуру.

Спочатку класифікація виконується для кожного пікселя зображення в одну з таких категорій: текст, малюнок, напівтонове зображення, фон, на основі аналізу текстурних ознак околиці пікселя за допомогою спеціальних фільтрів. Далі пікселі групуються відповідно до їх просторового розташування. Остаточне рішення приймається з допомогою деякого класифікатора, наприклад, у роботі [26] використовується нейронна мережа. Основними недоліками даного підходу є неможливість правильно визначити категорію у разі зливаються і символів, що перекриваються, а також обчислювальна складність алгоритму.

При застосуванні кратномасштабних ортогональних вейвлет-пакетів, підібраних згідно з умовою максимізації міжкласових відмінностей, вихідне зображення піддається вейвлет-перетворенню. Вектор ознак формується на основі розрахунку друго і третього центрального моменту за допомогою ковзного вікна у кожному діапазоні частот. Класифікатор заснований на нейронній мережі, навченій методом зворотного розповсюдження помилки. До переваг даного методу можна віднести незалежність від структури документа та хороші результати роботи з перекриваються областями [25].

Класифікація областей на основі фільтра Габора наведена у статті [26]. Вихідне зображення піддається фільтрації Габора з різними характеристиками (у роботі розглядається банк з 8 вузькосмугових фільтрів), після чого для кожного отриманого зображення розраховується локальна енергія для кожного пікселя на його околиці. Кластеризація здійснюється для кожного окремого пікселя виходячи із значень його енергії та просторового розташування. Результати кластеризації досить точні, проте алгоритм є ресурсомістким.

Основний недолік методів, заснованих на аналізі спектральних ознак – це висока обчислювальна складність, проте їх перевагою є досить точне та інтуїтивно зрозуміле опис текстури області, так як вони, в першу чергу чергу, виявляють періодичність текстури, що є репрезентативним ознакою текстових областей, навіть із зображеннями з невеликим відсотком текстових областей [32].

2.4 Визначення кута нахилу рядків тексту

Після того, як текстові блоки були знайдені, здійснення сегментації необхідно визначити їх спрямованість. Нормалізація, що є вирівнювання рядків уздовж горизонталі, часто наводить зниження якості зображення. Тому краще обмежитися визначенням кута нахилу рядків тексту та

виконувати сегментацію з його врахуванням. Для вирішення даної задачі найчастіше використовуються алгоритми аналізу проекцій, перетворення Хафа, метод зв'язкових компонент, оцінка міжрядкової кореляції [25].

Аналіз горизонтальних та вертикальних проекцій рядків [27, 19] є одним із найпростіших способів виявлення кута нахилу, однак він вимагає побудови та оцінки проекцій для кожного можливого кута нахилу зображення, що є витратним по часу.

Більш успішно для вирішення цього завдання застосовується метод Хафа, що складається з наступних кроків:

Крок 1. Вибір підмножини опорних точок, що визначають центри чи межі зв'язкових компонентів.

Крок 2. Застосування перетворення Хафа для вибраного підмножини опорних точок.

Крок 3. Кут нахилу, який зустрічається найчастіше і є передбачуваним кутом нахилу рядків.

Крок 4. Опорні точки, що формують між собою знайдені прямі, утворюють рядки.

Основною перевагою методу Хафа є його точність, що залежить від частоти дискретизації. Оскільки аналізу піддаються лише опорні точки, метод має гарну швидкість роботи. Проте його точність залежить від ступеня дискретизації осередків накопичення. Складність методу Хафа дорівнює $O(N^2)$.

В розробляємому методі розпізнавання кирилиці у відеопотоці, пошук кута буде виконуватися згідно наступного алгоритму ($Y_{ср}$ – середнє відхилення зв’язкових компонентів, що належать рядку,від її центру):

Крок 1. Застосування методу зв’язкових компонентів.

Крок 2. Фільтрування знайдених зв’язкових компонентів (мета – залишити компоненти, що представляють рядки, прибрати дрібні деталі), залишаються лише компоненти висотою між 20 і 95 %.

Крок 3. Сортування зв’язкових компонентів за x-координатом лівого кордону компоненти.

Крок 4. Початкове завдання рядків:

Крок 4.1. $Y_{cp} \to 0$;

Крок 4.2. Для кожної компоненти знайти поточну рядок, що має найбільшу площуперекриття із поточною компонентою;

Крок 4.3. Якщо такого рядка немає, то:

Крок 4.3.1. Створити новий рядок, прикріпити до ній компоненту, задати верхній та нижній кордон рядка згідно меж компоненти.

Крок 4.4. Інакше:

Крок 4.4.1. Додати компонент до рядка, змінити координати рядка.

Крок 4.5. Крок 4.1.

Крок 5. Інтерполяція рядків проводиться за допомогою методу найменших квадратів.

Даний алгоритм має більшу точність і швидкістю роботи в порівнянні з методом Хафа [27]. Його складність *O(N)* обумовлена застосуванням методу зв’язкових компонентів та методу найменших квадратів.

У роботі [27] запропоновано алгоритм уточнення висот рядків на основі аналізу гістограми висот зв’язкових компонентів. Гістограма висот знайдених зв’язкових компонентів згладжується фільтром Гауса, що значно знижує кількість піків. В результаті повторного застосування фільтра Гауса в гістограмі виділяються 3 піки, які відповідають висот знаків розділових знаків, малих і великих букв відповідно. Таким чином, висоти текстових рядків можуть бути уточнені на основі крайнього правого піку гістограми.

Сегментація блоку на рядки базується на методі зв'язкових компонентів, виявленні базової (середньої) лінії та висоти малих символів (x-height). Для символів, що мають однакове написання в рядковому та великому вигляді, ця інформація є визначальною при розпізнаванні.

2.5 Математична модель розпізнавання кирилиці у відеопотоці

Запропонуємо нову математичну модель системи оптичного розпізнавання кирилиці у відеопотоці з модулем комбінування результатів розпізнавання об’єкта на одиночних зображеннях та модулем зупинки. При використанні мобільних пристроїв виникає можливість розглядати відеопотік як цифрове уявлення об’єкта, що дозволяє вирішувати завдання, недоступні у разі аналізу одиночної. фотографії. Зовнішні умови зйомки можуть призвести до того, що об’єкт на одиночному кадрі спотворено. Прикладом такого спотворення є відблиск, що виявляється на глянсовій поверхні (рис. 2.4).

Рисунок 2.4 – Фрагмент кадру з бліком на глянцевій поверхні

Оскільки у відеопотоці геометричне положення об’єкта, що знімається, може змінюватися, блік також змінює своє становище, що дозволяє отримати інформацію про об’єкт, що приховується, на іншому кадрі відеопотоку.

Пропонована модель системи розпізнавання об’єкта у відеопотоці представлена на рисунку 2.5. Відеопотік представляється як послідовність $I_t(x) \in I$ захоплюючих зображень об’єкта x у дискретні моменти часу t. Час, потрібний для отримання оновленого результату розпізнавання після введення чергового образу $I_t(x)$, у загальному випадку, є функцією від

зображення та внутрішнього стану системи. Результат, що враховує зображення $I_t(x)$, може бути доступний тільки в момент часу $T(t) \geq t$. Нехай у момент часу t_0 відбувається захоплення зображення $I_{t_0}(x)$. Результат розпізнавання одиночного кадру $\hat{f}(I_{t_0}(x))$ стає доступним на момент часу $t_1 \geq t_0$ і реєструється в модулі пам'яті системи. Після цього відбувається комбінування результатів розпізнавання зображень об'єкта, накопичених на поточний момент, та момент часу $t_2 = T(t_0) \geq t_1$ відбувається висновок поточного інтегрованого результату розпізнавання R_{t_2}, що враховує інформацію, яка міститься у зображеннях $I_0(x), I_{T^1(0)}(x), I_{T^2(0)}(x), ..., I_{t_0}(x)$. Якість результату характеризується близькістю результату R_{t_2} до справжнього значення $\nu(x)$ об'єкта x, згідно з деякою метрикою. Після виведення результату відбувається захоплення чергового зображення $I_{t_2}(x)$ та процес триває.

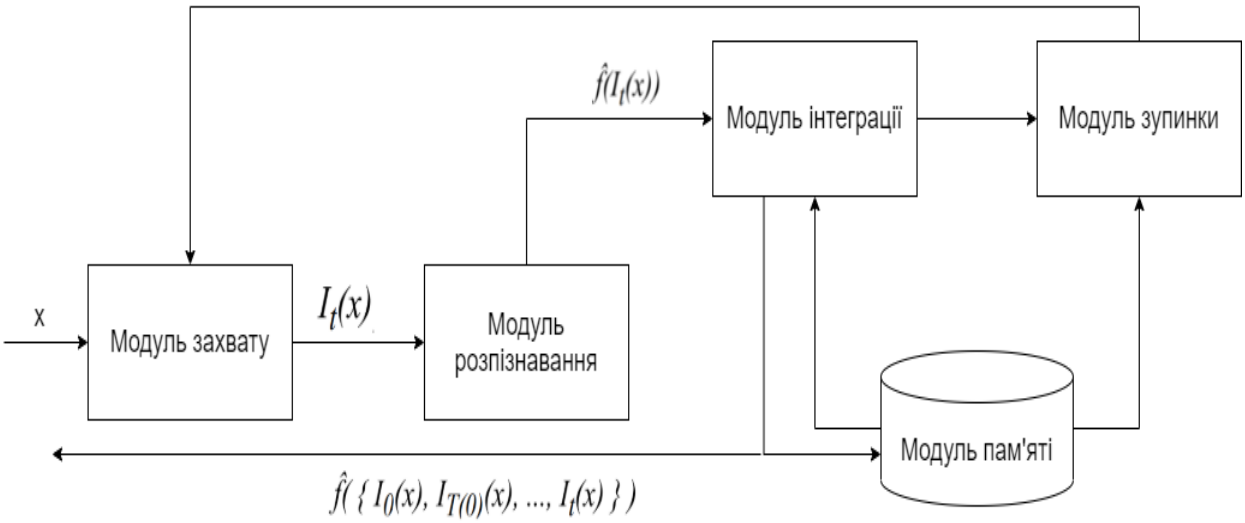


Рисунок 2.5 – Схема методу розпізнавання кирилиці у відеопотоці

У рамках такої системи виникають задачі, нетипові для традиційних систем розпізнавання об'єктів: завдання комбінування (інтеграції) результатів розпізнавання одного й того самого об'єкта на різних зображення

в єдиний результат R_{t_2}, і завдання зупинки процесу розпізнавання, тобто. прийняття в момент часу t_2 рішення про те, що процес захоплення слід припинити та накопичений до поточного моменту результат прийняти за остаточний. Як функціонал ефективності системи в момент зупинки $t = t_{stop}$ пропонується розглядати лінійну комбінацію :

$$a \cdot \rho(R_{t_{stop}}, \nu(x)) + b \cdot W(t_{stop}), \tag{2.8}$$

де, a,b – константи;

$\rho(R_{t_{stop}}, \nu(x))$ – відстань від інтегрованого результату R_t до справжнього значення $\nu(x)$, що характеризує якість результату;

$W(t)$ – штрафна функція від часу [33].

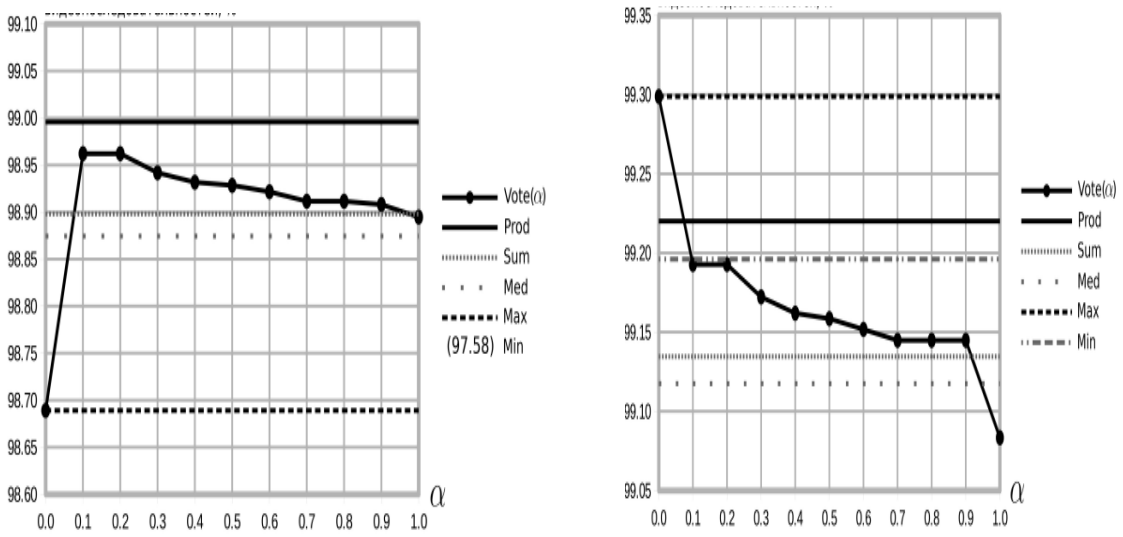
На побудову модуля інтеграції результатів розпізнавання окремих зображень може впливати як модель результату розпізнавання, так і природа об’єкта, що розпізнається. Важливим питанням у рамках цієї завдання є питання про вибір стратегії, яка буде оптимальною за заданої моделі вхідних даних. Для дослідження впливу моделі вхідних даних на вибір оптимальної стратегії комбінування в рамках атестаційної роботи було поставлено наступний експеримент: були підготовлені набори даних, що містять відеопослідовність окремих друкованих символів із спотвореннями, характерними для мобільної відеозйомки, та результати їхньої класифікації за допомогою згорткової нейронної мережі, навченої на незалежній навчальній вибірці.

У відеопослідовності, об’єднаних у групу *A*, містилися зображення з помилками попередньої обробки (викликані некоректною або недостатньо точною роботою алгоритмів локалізації документа, сегментації текстових рядків на окремі символи тощо).

У відеопослідовності групи *B* не було помилок попередньої обробки. На отримані набори даних проведено порівняння базових стратегій

комбінування класифікаторів у рамках Байєсівської моделі результату класифікації: правило твору оцінок, суми оцінок, мінімуму, максимуму, медіани, а також узагальнене правило голосування: лінійна комбінація частоти класу з коефіцієнтом α та його максимальної оцінки з коефіцієнтом $(1-\alpha)$.

На рисунку 2.6 продемонстровано значну різницю в оптимальному виборі стратегії комбінування в залежності від моделі вхідних даних: на тестових наборах групи *A* вищу точність розпізнавання відеопослідовностей забезпечують правило твору, голосування та правило суми. При цьому на тестових наборах групи *B*, в яких не було помилок попередньої локалізації та сегментації символів, більш високу точність розпізнавання забезпечує правило максимуму [34].



а) б)

Рисунок 2.6 – Порівняння точності розпізнавання відеопослідовностей символів із використанням базових стратегій комбінування:
а) група *A*; б) група *B*

Слід зазначити, що пряме застосування розглянутих правил комбінування неможливо у разі, якщо модель результату розпізнавання об’єкта складніша, ніж простий результат класифікації. У якості прикладу

такого об’єкта можна назвати текстовий рядок (рядковий об’єкт), для якого класифікація проводиться незалежно для кожного символу.

2.6 Алгоритм комбінування результатів розпізнавання строкового об’єкту

Результат розпізнавання одиночного символу розглядався як відображення з безлічі класів C, об’єднаного з позначкою порожнього класу λ, в множину нормалізованих оцінок власності. Як метрика на безлічі $\hat{C}$ всіляких результатів розпізнавання символ використовувався варіант манхеттенської метрики з безліччю значень, укладених у відрізку [0, 1].

Нехай задана функція r комбінування результатів розпізнавання одиночних символів $r : \hat{C}^N \times (\mathbb{R}_0^+)^N \to \hat{C} \setminus \{\hat{\lambda}\}$, що приймає на вхід N результатів розпізнавання одиночних об’єктів $a_1, a_2, ..., a_N$ таких, що $\exists_i : a_i \neq \lambda$, та набір асоційованих з ними невід’ємних ваг $\omega_1, \omega_2, ..., \omega_N$, що відбивають значимість кожного з результатів. Вимагаємо від функції комбінування r наступної властивості:

$$r(a_1,...,a_N,\omega_1,...,\omega_N) = r(r(a_1,...,a_{N-1},\omega_1,...,\omega_{N-1}),a_N,\omega_1+...+\omega_{N-1},\omega_N). \quad (2.9)$$

В рамках запропонованого алгоритму як функція комбінування одиночних символів використовувалося зважене середнє:

$$r(a_1,...,a_N,\omega_1,...,\omega_N)(c) = \frac{1}{W_N}\sum_{i=1}^{N} a_i(c)\cdot\omega_i, \ \forall c \in C \cup \{\lambda\}. \quad (2.10)$$

Позначимо λ як порожній результат класифікації одиночного символу $\hat{\lambda} \overset{def}{=} \{(\lambda,1),(c_1,0),...,(c_K,0)\}$. Результатом $\mathcal{X}$ розпізнавання рядкового об’єкта

називатимемо послідовність елементів, що належать множині $\hat{C}\setminus\{\hat{\lambda}\}$. В якості метрики $\rho_{\aleph}$ на множині $\aleph$ всіх можливих результатів розпізнавання малих об'єктів використовувалося нормалізоване узагальнене відстані Левенштейна.

Нехай задані *N* результатів розпізнавання строкового об'єкту $X_1,...,X_N$, де $|X_i| = n_i > 0$:

$$X_1 = x_1^1 x_2^1 ... x_{n_1}^1,\ X_2 = x_1^2 x_2^2 ... x_{n_2}^2,\ ...,\ X_N = x_1^N x_2^N ... x_{n_N}^N, \tag{2.11}$$

а також для кожного результату X_i задана його вага ω_i.

При розрахунку інтегрованого результату розпізнавання рядкового об'єкта породжуватимемо набір проміжних результатів $R^{(i)}(X_1,...,X_i,\omega_1,...,\omega_i)$. На першому кроці алгоритму: $R^{(1)}(X_1,\omega_1) = X_1$. На кожному наступному і-му кроці алгоритму будується оптимальне вирівнювання рядків X_i та $R^{(i-1)}(X_1,...,X_{i-1},\omega_1,...,\omega_{i-1})$ за допомогою схеми динамічного програмування.

Нехай $d(l,m) \overset{def}{=} \rho_{\aleph}(X_{i1...l}, R^{(i-1)}(X_1,...,X_{i-1},\omega_1,...,\omega_{i-1})_{1...m})$, а $P_p(l,m)$ – допоміжні функції для $\rho \in \{1,2,3\}$. Розрахунок $d(l,m)$ і $P_p(l,m)$ проводиться згідно з наступною процедурою:

$$\begin{aligned}
&d(0,0) = 0,\ d(l,0) = \sum_{k=1}^{l} \rho_C(x_k^i, \hat{\lambda}),\ d(0,m) = \sum_{k=1}^{m} \rho_C(\lambda, r_k^{\hat{(i-1)}}) \\
&P_1(l,m) = \rho_C(x_l^i, \hat{\lambda}) + d(l-1,m), \\
&P_2(l,m) = \rho_C(\lambda, r_m^{\hat{(i-1)}}) + d(l,m-1), \\
&P_3(l,m) = \rho_C(\lambda, r_m^{\hat{(i-1)}}) + d(l-1,m-1), \\
&d(l,m) = \min\{P_1(l,m), P_2(l,m), P_3(l,m)\}.
\end{aligned} \tag{2.12}$$

Для розрахунку результату на i-му кроці $R^{(i)}(X_1,...,X_i,\omega_1,...,\omega_i)$ введемо дві допоміжні функції $t_x:\{0,...,n_i+n_{R_{i-1}}\}\to\{1,...,n_i\}$ та $t_R:\{0,...,n_i+n_{R_{i-1}}\}\to\{1,...,n_{R_{i-1}}\}$, розрахунок яких проводиться за наступною рекурентною процедурою:

$$
\begin{aligned}
&t_x(0)=n_i,\ t_R(0)=n_{R_{i-1}},\\
&t_x(k+1)=\begin{cases} t_x(k), \textit{якщо}\ {}^{P_2(t_x(k),t_R(k))=d(t_x(k),t_R(k))\wedge}_{\wedge P_1(t_x(k),t_R(k))\neq d(t_x(k),t_R(k))}\\ t_x(k)+1, \textit{в інших випадках}, \end{cases}\\
&t_R(k+1)=\begin{cases} t_R(k), \textit{якщо}\ P_1(t_x(k),t_R(k)=d(t_x(k),t_R(k))) \\ t_R(k)+1, \textit{в інших випадках}. \end{cases}
\end{aligned}
\tag{2.13}
$$

Результат на i-му кроці розраховується так:

$$
\begin{aligned}
&n_{R_i}=\min\{k: t_x(k)=t_R(k)=0\},\\
&R(X_1,...,X_i,\omega_1,...,\omega_i)=r_1r_2...r_{n_{R_i}},\\
&r_k=\begin{cases} r(r^{i-1}_{t_{R(t(k))+1}},\hat{\lambda},W_{i-1},\omega_i), \textit{якщо}\ t_x(t(k))=t_x(t(k-1)),\\ r(\hat{\lambda},x^i_{t_{x(t(k))+1}},W_{i-1},\omega_i), \textit{якщо}\ t_R(t(k))=t_R(t(k-1)),\\ r(r^{i-1}_{t_{R(t(k))+1}},x^i_{t_{x(t(k))+1}},W_{i-1},\omega_i), \textit{в інших випадках}, \end{cases}
\end{aligned}
\tag{2.14}
$$

де $W_i\overset{def}{=}\sum_{k=1}^{i}\omega_k$;

допоміжна функція $t(k)\overset{def}{=}n_{R_i}-k+1$;

r – функція інтеграції результатів розпізнавання одиночних об'єктів (2.8).

Трудомісткість обчислення функції r (2.8) та метрики на результатах розпізнавання одиночних символів складає O(K), де K – кількість класів, яким відбувається класифікація кожного символу. Оскільки верхня оцінка на довжину результуючого рядка R після виконання i-ї ітерації алгоритму складає $O(\sum_{j=1}^{i}|x_i|)\le O(i\cdot\max_{j=1}^{i}|x_i|)$, трудомісткість кожної ітерації

алгоритму можна оцінити як $O(M^2NK)$, де $M = \max_{i=1}^{N} |X_i|$, і загальну трудомісткість запропонованого алгоритму як $O(M^2N^2K)$.

Перевіривши алгоритм на практиці можна судити про те, що інтеграція має властивість спадної прибутковості (у термінології алгоритмів «anytime»). Ця властивість є важливою для вирішення завдання зупинення розпізнавання об'єкта у послідовності відео.

2.7 Метод зупинки розпізнавання об'єкту у відеотопоці

Запропонуємо новий метод зупинки процесу розпізнавання об'єкта у відеопотоці на основі порогового відсікання оцінки очікуваної відстані між поточним і наступним інтегрованими результатами, і представлений новий алгоритм зупинки розпізнавання рядкового об'єкта. Для вирішення завдання зупинки з функцією збитку пропонується використовувати наступний метод:

1. Оцінити очікувану відстань (у термінах метрики ρ) від поточного інтегрованого результату розпізнавання об'єкта R_n (відомого на кроці n) до невідомого наступного результату R_{n+1}.

2. Приймати рішення про зупинення процесу на кроці n, виробляючи граничне відсікання відстані, оціненої в пункті 1, таким чином апроксимуючи поведінку правила N_B.

Побудуємо на основі запропонованого методу алгоритм зупинення процесу розпізнавання рядкового об'єкта. Нехай задана функція інтеграції результатів розпізнавання рядкового об'єкта *R*. Як метрика ρ на рядкових об'єктах пропонується використовувати нормалізована узагальнена відстань Левенштейна. Для того щоб апроксимувати поведінку правила зупинки N_B, на n-му кроці процесу необхідно обчислювати оцінку очікуваної відстані між сусідніми інтегрованими результатами розпізнавання $\Delta_n \overset{def}{=} E(\rho(R_n, R_{n+1}))$, маючи доступ до спостережень $X_1 = x_1, ..., X_n = x_n$. Для обчислення оцінки

пропонується провести моделювання наступного інтегрованого результату виходячи з припущення, що нове спостереження буде близько до вже отриманих на попередніх кроках спостережень:

$$\triangle_n \overset{def}{=} \frac{1}{n+1}(\delta + \sum_{i=n}^{n} \rho(R_n, R(x_1, x_1, ..., x_n, x_i))), \quad (2.15)$$

де δ – зовнішній параметр, що настроюється.

3 ПРОГРАМНА РЕАЛІЗАЦІЯ ПРОТОТИПУ МЕТОДУ РОЗПІЗНАВАННЯ КИРИЛИЦІ У ВІДЕОПОТОЦІ

3.1 Обґрунтування вибору середовища програмної реалізації

У рамках кваліфікаційної роботи був розроблений прототип методу розпізнавання кирилиці у відеопотоці. Була розроблена математична модель розпізнавання кирилиці у відеопотоці та запропоновані алгоритми комбінування результатів розпізнавання та зупинки. Програмна реалізація прототипу математичної моделі методу розпізнавання кирилиці у відеопотоці було розроблена у вигляді додатку під опереаційну систему Android. Розроблений додаток дозволяє розпізнавати дані (український текст) з зображення паспортів в режимі реального часу які надходять до застосунку напряму з камери смартфону користувача.

Для написання програмного коду була використана мова програмування – Kotlin, разом з інтеграцією OCR модулю Google Vision OCR API.

Kotlin представляє сучасну, статично типізовану і одну з мов програмування, що швидко розвивається, створена і розвивається компанією JetBrains. Kotlin можна використовувати для створення різних програм. Це і програми для мобільних пристроїв - Android, iOS. Причому Kotlin дозволяє писати кросплатформовий код, який застосовуватиметься на всіх платформах. Це і веб-додатки, причому як серверні програми, які відпрацьовують на стороні сервера - бекенда, так і браузерні клієнтські програми - фронтенд. Kotlin також можна застосовувати для створення десктопних програм, для Data Science і так далі [34].

Google Cloud Vision OCR є частиною Google cloud vision API для позпізнавання тексту із зображень. Зокрема, є дві моделі, які допомагають розпізнавати символи:

– Text_Annotation: витягує та виводить закодовані тексти з будь-якого зображення (наприклад, фотографії вулиць чи пейзажів). Оскільки спочатку вона була розроблена для використання в різних ситуаціях освітлення, модель у певному сенсі більш надійна в читанні слів різних стилів, але лише на більш розрідженому рівні;

– Document_Text_Annotation: розроблено для розпізнавання текстових документів (відсканованих книг, документів). Таким чином, хоча вона підтримує читання менших і більш стислих текстів, вона менш адаптована до зображень у дикій природі.

Зображення обробляється віддалено в Google Cloud і створює відповідні формати JSON з результатами відповідно до викликаної функції.

Головним недоліком Google Cloud Vision OCR є те, що API не підтримує розпізнавання символів українського алфавіту, тож для реалізації прототипу системи розпізнавання кирилиці у відеопотоці окремо був інтегрований розроблений метод розпізнавання кирилиці.

В якості системи сховища даних була обрана хмарна база даних Firebase. Firebase - це хмарна база даних, яка дозволяє користувачам зберігати та отримувати збережену інформацію, а також має зручні засоби та методи взаємодії з нею. Firebase зберігає текстові дані в JSON форматі та надає зручні методи для читання, оновлення та вилучення даних. Також, Firebase може допомогти з реєстрацією та авторизацією користувачів, зберіганням сесій (авторизовані користувачі), медіафайлів до яких легко надає доступ завдяки Cloud Storage [35].

Головна причина, через яку було вирішено використовувати Firebase, це гнучкість і швидкість інтеграції в проект. Firebase дозволяє зберігати швидкість, що є дуже важливими, адже текст повинен розпізнаватися в режимі реального часу. Не потрібно відволікатися на якісь інші речі (створення бази даних, написання API прийому та отримання даних). Уся серверна частина лягає на плечі цього сервісу.

Для тестування базового функціоналу використовувалася бібліотека модульного тестування Junit, яка дуже легко інтегрується до мови програмування Kotlin. Для поглибленого тестування прототипу системи з використанням методу розпізнавання кирилиці у відеопотоці, була інтегрована бібліотека інтеграційного тестування – Cucumber.

3.2 Структура програми

Процес функціонування системи, що розробляється, складається з двох етапів.

Перший етап – навчання системи – полягає в наповненні бази знань системи відомостями про символи кириличного алфавіту, що зустрічаються в текстах, та різноманітних шрифтів. Ці знання можна отримати з відкритих джерел. У навчальній виборці формується зображення всіх букв алфавіту і вказується відповідність кожного зображення необхідному коду електронного представлення. Зображення символів повинні відповідати використовуваним у різних шрифтах. Система робить онлайн-розпізнавання зображень, що вводяться, і заносить в базу знань отриману інформацію про процес синтезу зображень.

База знань і двох пов’язаних частин:

- база фреймів букв;
- словник.

Фрейми букв відображають знання про елементи, що формують зображення символів та їх просторові взаємини. Словник складається зі слів, що становлять лексикон програми. Слова представлені у вигляді кадрових структур, що посилаються на кадри букв з відповідної частини бази знань і відображають порядок проходження букв у кожному конкретному слові. Заповнення словникової частини бази знань здійснюється автоматичним перетворенням текстового словника.

Другий етап функціонування системи є робочім. На ньому виконується безпосередньо розпізнавання зображень та формування їх текстових уявлень.

Блок розпізнавання системи складається з наступних компонентів:

- модуль захвату та попередньої обробки зображення;
- модуль розпізнавання літер;
- модуль розпізнавання слів;
- модуль зупинки.

Модуль захвату та попередньої обробки зображення є компонентом системи, що відповідає за графічний аналіз растру зображення та послідовне виділення в ньому структурних елементів зображень на запит модулю розпізнавання букв. Він отримує в запиті вказівку на вигляд очікуваного елемента. Тоді, проводячи обробку зображення, він намагатиметься слідувати таким шляхом, який призведе до виявлення елемента шуканого типу. Закінчивши виконання завдання, модуль повертає інформацію про виявлений елемент та точки перетину його іншими елементами. У разі неможливості виконати завдання повертається спеціальна, негативна відповідь.

Завданням модулю розпізнавання літер є отримання за набору елементів зображення, визначення взаємозв’язків між ними та прийняття рішення на основі отриманої інформації про літеру. Отримуючи від модулю захвату та попередньої обробки чергову порцію інформації, він здійснює в базі кадрів букв пошук кадру або набору кадрів, які можуть бути інстанційовані наявними даними. Один з таких кадрів оголошується активним і використовується як гіпотеза про літеру, що спостерігається. На підставі гіпотези модуль розпізнавання літер може робити передбачення про наступні елементи, що зчитуються, і спеціалізувати запити до модулю захвату та попередньої обробки зображення.

Модуль розпізнавання слів працює зі словниковою частиною бази знань. За аналогією з принципом роботи модулю розпізнавання літер, він виконує завдання розпізнавання слів, оперуючи одержуваними від модулю

розпізнавання літре даними про знайдені літери. Проводячи пошук за основою фреймів слів, модуль розпізнавання слів висуває гіпотези про знайдені слова і робить прогнози щодо очікуваних букв. На рисунку 3.1 описана структурна схема проектованої системи.

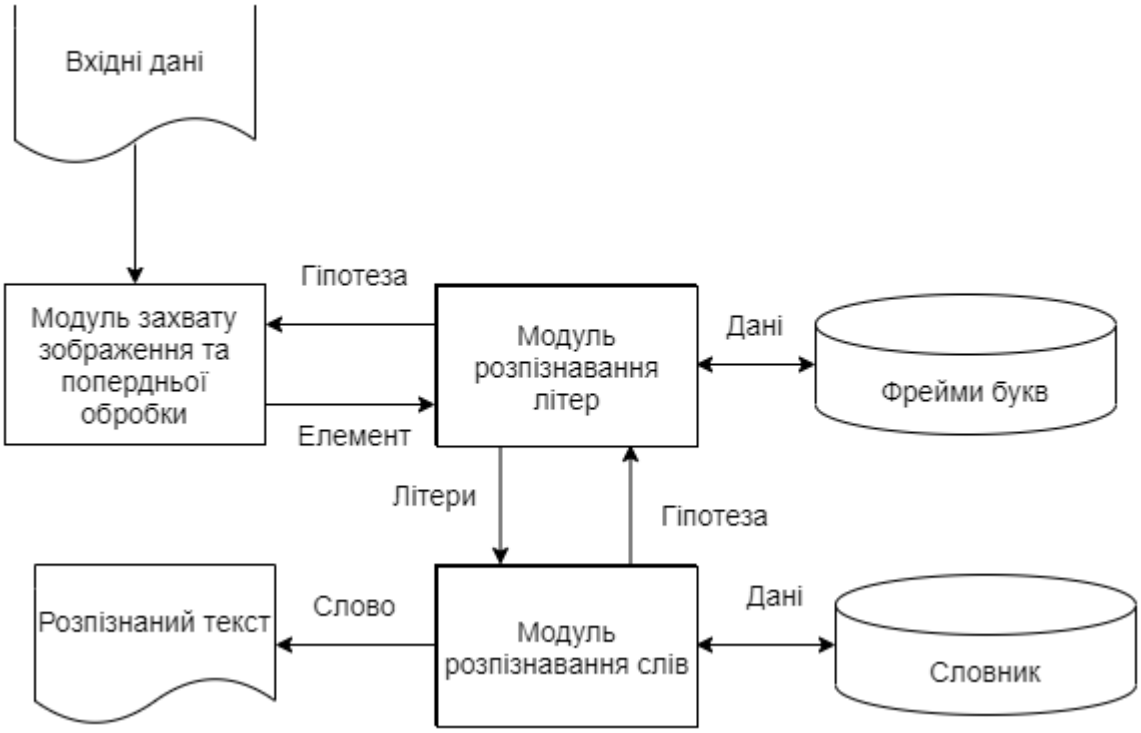


Рисунок 3.1 – Структурна схема проектованої системи

3.3 Вхідні дані

Розроблений прототип працює з вхідним відеопотоком із зображенням паспорту громадянина України, який надходить на напряму з камери смартфону користувача. Для коректного розпізнавання тексту документа важливо, щоб він знаходився у фронтальній площині та мав нахил від горизонтальної осі не більш ніж 30 градусів. З відеопотоку виділяється зображення документа, яке буде використано для подальшої обробки та розпізнавання текстової інформації (рис. 3.2).

Рисунок 3.2 – Вхідне зображення документу

3.4 Захват та попередня обробка зображення

Після того, як система отримує на вхід відеопотік із зображенням документу відбувається захват конкретного кадру з відеопотоку, який передається на подальшу обробку за для детектування шукомого елементу на зображенні [36].

По-перше проводиться ротація та попередня обробка зображення документа, на основі уявлень про вид зображення у стандартному вигляді: на всю робочу область, без нахилу. Для цього виконуються такі маніпуляції над вихідним зображенням:

Крок 1. Переведення зображення у відтінки сірого (рис. 3.3).

Рисунок 3.3 – Зображення у відтінках сірого

Крок 2. Згладження шумів на кордонах за допомогою фільтра Гауса

Крок 3. Знаходження межі зображення за допомогою алгоритму Кані

Крок 4. Бінаризація зображення

Крок 5. Знаходження всіх необхідних ліній

Крок 6. Знаходження кутів для всіх ліній

Крок 7. Кластеризація кутів за двома центрами

Крок 8. Вибір кластеру із великою кількістю елементів

Крок 9. Знаходження кута для ротації (мається на увазі, що зображення мало відхилення від 0 до 30 градусів)

Крок 10. Відрізання невідфільтрованих областей (що не містять зображення паспорта) (рис 3.4).

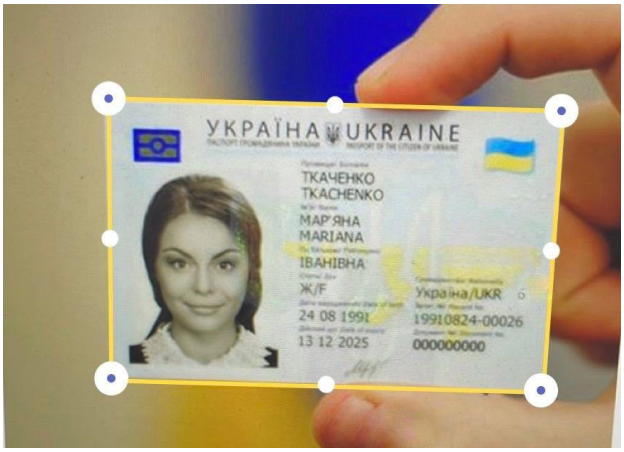


Рисунок 3.4 – Виділення контурів документу

Крок 11. Отримання фінального зображення.

Рисунок 3.5 – Фінальне зображення документу

3.5 Розпізнавання даних

Оскільки розташування даних у паспорті заздалегідь визначено, й у всіх зразків однакове – отримуємо області, що містять лише необхідну інформацію.

По-перше проводиться бінаризація зображення, головною метою якої є радикальне зменшення кількості інформації, з якою доводиться працювати. Результат бінаризації зображення наведений на рисунку 3.6.

Рисунок 3.6 – Результат бінаризації зображення

Далі відбувається виявлення контурів у бінаризованому зображенні та отримання контурів обмежених областей та сортування областей інтересу (передбачуване місце знаходження інформації) на документі (рис. 3.7).

Рисунок 3.7 – Документ з виділеними областями інтересу

Наступним кроком є групування областей по лініях, сортування за площами, висотами та відстанню між областями та поділ областей на поля. Результат виділення основних полів документу зображений на рисунку 3.8.

Рисунок 3.8 – Результат виділення основних полів документу

3.5.1 Алгоритм локальної адаптивної бінаризації

Алгоритм локальної адаптивної бінаризації уявляє собою наступні кроки:

Крок 1. Пороги для поверхні зображення обчислюються, під час ковзання прямокутного вікна (кожен піксель бінаризується ґрунтуючись на інформації про своїх сусідів).

Крок 2. Поріг «T» у центральному пікселі вікна обчислюється за формулою:

$$T = m - k\alpha(m - M), \alpha = 1 - \frac{S}{R}, R = \max(s), \quad (3.1)$$

де m – мінімальне значення інтенсивності пікселя зображення у відтінках сірого у вікні;

k – константа, встановлена рівною 0.2;

M – мінімальне значення інтенсивності пікселя у всьому зображенні в відтінках сірого;

S – стандартне відхилення;

R – максимальне зі стандартних відхилень по всіх околицях.

Крок 3. Зображення порівнюється з граничною поверхнею і всі пікселі з інтенсивністю більшої граничної, робляться білими (255), а всі інтенсивності менші граничної робляться чорними(0) [38].

Для цього алгоритму єдиний вхідний параметр – розмір вікна. Експериментально було встановлено, що найкращі результати розпізнавання досягаються з розміром вікна, що дорівнює половині висоти тексту. Таким чином, алгоритм здатний обробляти зразки з різним розміром тексту, але потрібен модуль, що оцінює висоту тексту.

3.5.2 Оцінка висоти тесту

Для оцінки висоти тексту використовувався наступний алгоритм:

Крок 1. Для кожного осередку на зображенні у «відтінках сірого» обчислюються інтенсивності. Це дозволяє отримати масив дисперсій, у яких довжина дорівнює висоті зображення [39].

Крок 2. Області дисперсії, у яких розбіжність більша за поріг T_V - розглядаються як текст. (T_V - середнє значення всього набору дисперсії)

Крок 3. Медіанна ширина зон тексту велика T_V - висота тексту.

3.5.3 Видалення ліній через аналіз областей навколо тесту

Алгоритм видалення ліній через аналіз областей навколо тексту виконує наступні кроки:

Крок 1. Локалізація пов’язаних областей тексту у бінаризованому зображенні.

Крок 2. Виявлення ліній на зображенні без тексту.

Крок 3. Видалення ліній з оригінального зображення.

Локалізація пов’язаних областей тексту уявляє собою наступі кроки:

Крок 1. Бінаризоване зображення розширюється за допомогою наступного ядра:

$$K_d = [111...1]_{1\times\frac{h}{2}}. \tag{3.2}$$

Крок 2. Зображення обробляється нормалізованим $h \times h$ ядром:

$$K_C = \frac{1}{h^2}\begin{bmatrix} 1 & \cdots & 1 \\ \vdots & \ddots & \vdots \\ 1 & \cdots & 1 \end{bmatrix}. \tag{3.3}$$

Крок 3. Результуюча обробка обмежена порогом 0.25.

Крок 4. Отримані блоби (Blob – регіон цифрового зображення, що має постійними властивостями) найімовірніше були отримані з тексту, тобто пов’язані регіони тесту знайдено.

На етапі виявлення ліній відбувається видалення всіх зон тексту (все, що знаходиться всередині пов’язаних регіонів) та застосування трансформації Хафа щоб знайти горизонтальні лінії (Лінії в більшості випадків «зламані» і мають не однакову товщину, тому трансформація Хафа застосовується з порогом рівним половині висоти, і максимальній відстані між лініями дорівнює половині ширини). Завдяки чому знаходится список позицій горизонтальних ліній.

На етапі видалення ліній використовуємо позиції горизонтальних ліній та витягуємо пікселі з оригінального зображення, що належать до ліній, але не до текстових символів. Припускаємо, що товщина ліній не стала, але не більше ніж *h/10*. Далі для кожної можливої товщини і кожної координати на лінії накладаємо на оригінальне зображення маску і відзначаємо всі пікселі, які збігаються з маскою. Створюємо відхилення, рухаючи маску вгору-вниз зі стартової позиції. Дане переміщення не повинно бути більшим за товщину лінії.

Використовувана маска:

$$x = \begin{bmatrix} 0 \\ 1 \\ \vdots \\ 1 \\ 0 \end{bmatrix}_{(d+2)\times 1}, \tag{3.4}$$

де d – це товщина лінії.

Нулі додані, щоб переконатися, що маска не накладається на текст. Якщо при накладенні маски, знайдені пікселі, що підходять їй, то він маркуються для видалення.

Вхідне зображення має містити текст, з навколишньою зоною, досить великий, щоб мати фонові лінії після того, як текстові зони будуть видалені. Лінії можуть бути під кутом, поки можливе їх виявлення по шматочках. При обробці текст не буде втрачено. Іноді невеликі ділянки тексту залишаються, але їх можна видалити в наступних кроках обробки [40].

3.6 Навчання методу розпізнавання кирилиці у відеопотоці

Навчальна вибірка складається зі стандартного пакету української мови для Google Vision. Результати обробки з її допомогою були не ідеальні, тому були покращені шляхом внесення змін до файлів навчальної вибірки. Під час навчання, метод отримує на вхід зображення у відеопотоц, аналізує всі позиції пікселів і вирівнює коефіцієнти, мінімізуючи помилку. Таким чином, досягається найкращий результат розпізнавання.

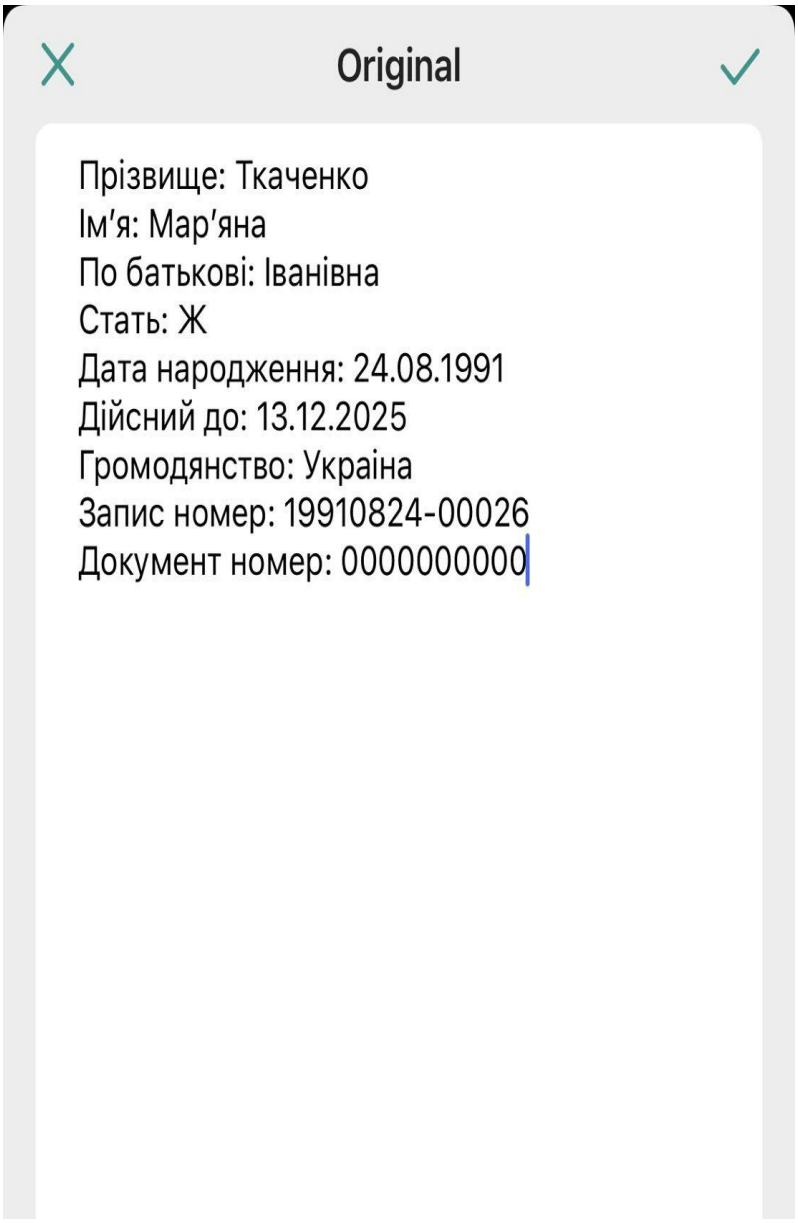
3.7 Інструкція користувача

Розроблений додаток має простий та інтуітивно зрозумілий інтерфейс. Для того, щоб почати користуватися застосунком, не потрібно проходити реєстрацію та авторизацію у системі. Користувачу достатньо просто завантажити застосунок на свій смартфон та відкрити його.

При запуску додатку користувач бачить основне вікно (рис. 3.9).

Рисунок 3.9 – Стартовий екран застосунку

Для того, щоб почати розпізнавання тексту, користувач повинен натиснути на кнопку «+», далі додаток запросить доступ до камери смартфону (лише при первинному запуску), після надання дозволу користувачу залишається лише навести камеру на документ. Система автоматично захватує кадр із відеопотоку та розпізнає текстову інформацію. Після завершення розпізнавання тексту на екрані з’явиться сторінка з розпізнаними текстовими даними (рис 3.10).



Риснок 3.10 – Результати розпізнавання тексту з паспорту

Завдяки збереженню результатів розпізнавання у базі даних, користувач може переглянути історію розпізнавань. Історія відображається на головній сторінці застосунку у вигляді списку з датою розпізнавання. При натисканні на елемент списку, користувач може переглянути повну текстову інформацію. Головна сторінка з історією розпізнавання зображена на рисунку 3.11.

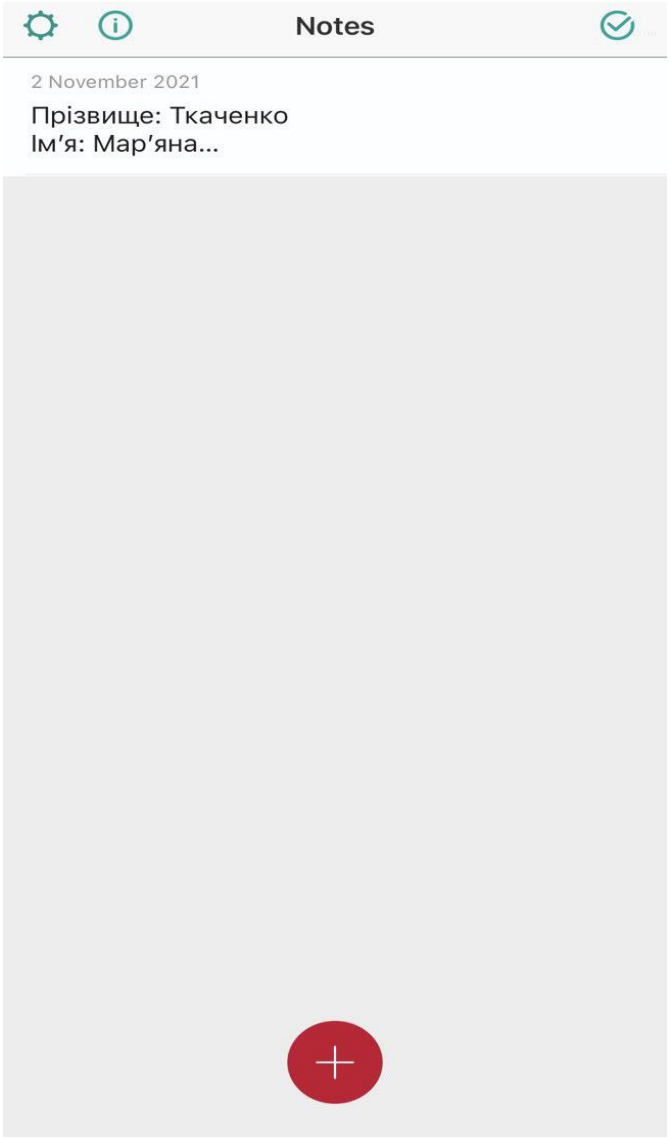


Рисунок 3.11– Головна сторінка з історією розпізнавання

3.8 Тестування розробленого методу

Під час тестування розробленого методу розпізнавання кирилиці у відеопотоці були використані як звичайні модульні junit тести, для тестування базових функцій алгоритму, так і BDD сценарії з використанням бібліотеки Cucumber.

BDD – це методологія розробки програмного забезпечення, що є відгалуженням методології розробки через тестування (TDD). Основною ідеєю даної методології є поєднання в процесі розробки суто технічних

інтересів та інтересів бізнесу, дозволяючи цим керуючому персоналу та програмістам говорити однією мовою. Для спілкування між цими групами персоналу використовується предметно-орієнтована мова, основу якої представляють конструкції з природної мови, зрозумілі неспеціалісту, які зазвичай виражають поведінку програмного продукту та очікувані результати.

Даний підхід був використаний для комплексного тестування програмного продукту.

BDD не надає жодних формальних правил, але наполягає на тому, щоб використовувався обмежений стандартний набір фраз, який включав би всі елементи специфікації поведінки. У 2007 році Деном Нортом був запропонований шаблон для специфікації, який отримав популярність і згодом став відомим як мова Gherkin.

Основні фрази мови Gherkin представлені у наступній таблиці:

Таблиця 3.1 – Основні фрази мови Gherkin

Ключове слово	Опис
Story(Feature)	Кожна нова специфікація починається з цього ключового слова, після якого через двокрапку в умовній формі пишеться ім'я історії.
As a	Роль тієї особи у бізнес-моделі, якій дана функціональність цікава.
In order to	У стислій формі які цілі переслідує обличчя.
I want to	У короткій формі описується кінцевий результат.
Scenario	Кожен сценарій однієї історії починається з цього слова, після якого через двокрапку у умовній формі пишеться мета сценарію. Якщо сценаріїв однієї історії кілька, то після ключового слова має писатися його порядковий номер.

Продовження таблиці 3.1

Ключове слово	Опис
Given	Початкова умова. Якщо початкових умов є кілька, то кожна нова умова додається з нового рядка за допомогою ключового слова And.
When	Подія, яка ініціює цей сценарій. Якщо подію не можна розкрити однією пропозицією, всі наступні деталі розкриваються через ключові слова And і But.
Then	Результат, який користувач повинен спостерігати зрештою. Якщо результат не можна розкрити однією пропозицією, всі наступні деталі розкриваються через ключові слова And і But.
And	Допоміжне ключове слово, аналог кон'юнкції.
But	Допоміжне ключове слово, аналог диз'юнкції.

На рисунку 3.12 зображений тестовий сценарій для розробленого застосунку.

```
Feature: Ukrainian text recognizer test

  @integration
  Scenario: User recognize the information from the Ukrainian passport in runtime
    Given Application is open
    When  User press on the button '+'
    And User points the camera at the image of the document 'passport.png'
    Then Page with the results of the text recognition should be open and contains the text:
      | Прізвище: Ткаченко          |
      | Ім'я: Мар'яна               |
      | По батькові: Іванівна       |
      | Стать: Ж                    |
      | Дата народження: 24.08.1991 |
      | Дійсний до: 13.12.2025      |
      | Громодянство: Україна       |
      | Запис номер: 19910824-00026 |
      | Документ номер: 0000000000  |
    When User press the button 'ok'
    Then the application main page should be open
    And 'recognition_history' table should contains 1 row with the current date
```

Рисунок 3.12 – Тестовий сценарій розпізнавання тексту з паспорту

Сценарій описує послідовність дій які виконує користувач, за для того, шоб розпізнати текстову інформацію з паспорту. Тест можна вважати успішно позитивним, якщо кожен окремий степ було пройдено вдало (про це свядчать зелені маркери біля кожного кроку).

Для тестування методу розпізнавання кирилиці у відеопотоці були створені тести, де вхідними даними є зображення документів з текстовою інформацією, які обробляються методом та на виході результат розпізнавання порівнюється за очікуваним результатом. Тобто для кожного зображення формується файл з розпізнаними даними у тому ж форматі, що й файл з очікуваними даними розпізнання (використовується для перевірки). Далі відбувається порівняння отриманих та коректних даних, використовуючи Відстань Левенштейна. Тестування відбувалося на вибірці з 20 документів, при різних нахилах та умовах освітлення. За результатам тестування генерується звіт у форматі html, який містить: загальну кількість тестів; кількість успішних тестів; кількість провальних тестів; детальну статистику кожного теста; процент точності розпізнавання. Фрагмент звіту з результами тестування зображений на рисунку 3.13.

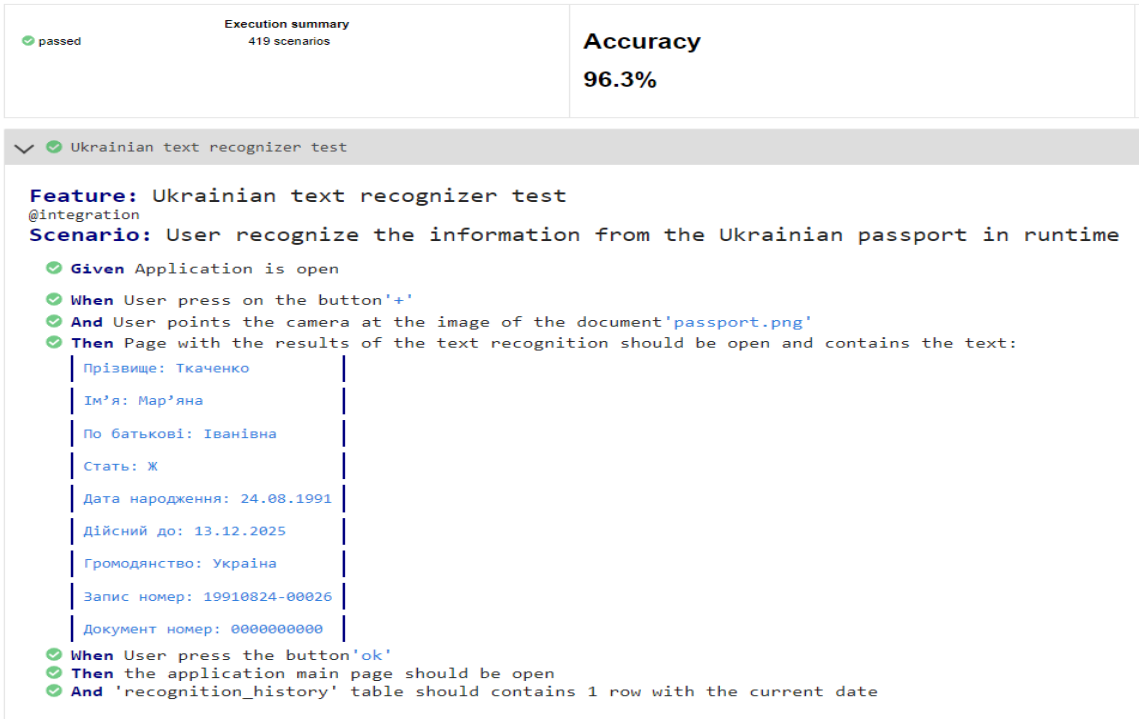


Рисунок 3.13 – Фрагмент звіту з результатами тестування.

ВИСНОВКИ

У рамках кваліфікаційної роботи були проведені дослідження та реалізація методу розпізнавання кирилиці у відеопотоці. Був розроблений прототип системи, з використанням реалізованого методу розпізнавання кирилиці у відеопотоці, для розпізнавання українського тексту з паспорту громадянина України.

Був проведений аналіз різноманітних методів розпізнавання об’єктів у відеопотоці та текстових даних на зображенні. Це дозволило зрозуміти проблеми, існуючі в даній області та виявити області застосування різноманітних методів, їх переваги та недоліки у рамках вирішення задачі перевірки документів у відеопотоці.

У ході дипломної роботи була розглянута частина невирішених питань систем розпізнавання тексту, а саме:

- розпізнавання кирилиці;
- комбінування результатів розпізнавання з декількох кадрів;
- словникова перевірка результатів розпізнавання.

Для вирішення проблем була побудована математична модель обробки, розпізнавання та інтеграції результатів.

Розроблений метод розпізнавання кирилиці у відеопотоці дозволяє досягти точності розпізнавання текстової інформації в 96%. Результати були отримані на тестовій вибірці документів.

Для демонстрації роботи методу розпізнавання кирилиці у відеопотоці, був розроблений прототип сервісу, що представляє собою застосунок для смартфонів на базі ОС Android. Архітектура розробленого прототипу дозволяє легко розширювати його функціонал та адаптувати алгоритм роботи не порушуючи його працездатність.

Результаті роботи було апробовано на конференції «Trends in science and practice of today».

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Rabotiahov, A., Kobylin, O., Dudar, Z., & Lyashenko, V. (2018, February). Bionic image segmentation of cytology samples method. In 2018 14th International Conference on Advanced Trends in Radioelecrtronics, Telecommunications and Computer Engineering (TCSET) (pp. 665-670). IEEE.

2. Работягов, А. В., Ляшенко, В. В., & Кобылин, О. А. (2016). Сегментация сложных изображений цитологических препаратов.

3. Lyashenko, V., Mohammad, A., & Kobylin, O. (2015). Experiments with Fusion of Images with Use of Wavelet Transformation in Problems of the Text Information Analysis.

4. Деркач, О. І. (2016). Аналітична обробка текстової інформації за допомогою засобів кластеризації. Young, 34(7).

5. Kobylin, O., Vyskrebentseva, S., & Petrova, R. (2019). Обробка даних, що містять пропуски в задачах кластеризації. Системи управління, навігації та зв'язку. Збірник наукових праць, 5(57).

6. Little, R. J., & Rubin, D. B. (2019). Statistical analysis with missing data (Vol. 793). John Wiley & Sons.

7. Jain, A. K., Murty, M. N., & Flynn, P. J. (1999). Data clustering: a review. ACM computing surveys (CSUR), 31(3), 264-323.

8. Perret, B., Chierchia, G., Cousty, J., Guimarães, S. J. F., Kenmochi, Y., & Najman, L. (2019). Higra: Hierarchical graph analysis. SoftwareX, 10, 100335.

9. Steinley, D. (2006). K-means clustering: a half-century synthesis. British Journal of Mathematical and Statistical Psychology, 59(1), 1-34.

10. Ackermann, M. R. (2009). Algorithms for the Bregman k-Median problem (Doctoral dissertation, University of Paderborn).

11. Khachumov, M. V. (2012). Distances, metrics and cluster analysis. Scientific and Technical Information Processing, 39(6), 310-316.

12. Huang, Z., & Ng, M. K. (1999). A fuzzy k-modes algorithm for clustering categorical data. IEEE Transactions on Fuzzy Systems, 7(4), 446-452.

13. Montgomery, D. C., Jennings, C. L., & Kulahci, M. (2015). Introduction to time series analysis and forecasting. John Wiley & Sons.

14. Zhang, J., Zhao, Z., Xue, Y., Chen, Z., Ma, X., & Zhou, Q. (2017). Time series analysis. Handbook of Medical Statistics, 269.

15. Крашений, І. Е., Попов, А. О., Раміpез, Х., Горріз, Х. М., Крашеный, И. Э., Попов, А. А., ... & Горриз, Х. М. (2016). Використання методів кластеризації в системах нечіткого виводу для діагностики хвороби Альцгеймера на основі ПЕТ-зображень.

16. Штовба, С. Д. (2006). Побудова функцій належності нечітких множин за кластеризацією експериментальних даних. Інформаційні технології та комп'ютерна інженерія, (2), 92-95.

17. Xu, J., Han, J., Xiong, K., & Nie, F. (2016, July). Robust and Sparse Fuzzy K-Means Clustering. In IJCAI (pp. 2224-2230).

18. Bodyanskiy, Y. V., Tyshchenko, O. K., & Mashtalir, S. V. (2019, June). Fuzzy Clustering High-Dimensional Data Using Information Weighting. In International Conference on Artificial Intelligence and Soft Computing (pp. 385-395). Springer, Cham.

19. Oleg, K., Sergii, M., & Mykhailo, S. (2017, October). Video Clustering via Multidimensional Time-Series Analysis. In Proceedings of the 9th International Conference on Information Management and Engineering (pp. 60-63). ACM.

20. Mashtalir, S., Mashtalir, V., & Stolbovyi, M. (2017). Video shot boundary detection via sequential clustering. International Journal "Information Theories and Applications, 24(1), 50-59.

21. Mashtalir, S., Mashtalir, V., & Stolbovyi, M. (2018, August). Representative Based Clustering of Long Multivariate Sequences with Different Lengths. In 2018 IEEE Second International Conference on Data Stream Mining & Processing (DSMP) (pp. 545-548). IEEE.

22. Bodyanskiy, Y., Kobylin, I., Rashkevych, Y., Vynokurova, O., & Peleshko, D. (2018, February). Hybrid fuzzy-clustering algorithm of unevenly and asynchronously spaced time series in computer engineering. In 2018 14th

International Conference on Advanced Trends in Radioelecrtronics, Telecommunications and Computer Engineering (TCSET) (pp. 930-935). IEEE.

23. Bodyanskiy, Y., Vynokurova, O., Kobylin, I., & Kobylin, O. (2016). Adaptive fuzzy clustering of short time series with unevenly distributed observations in Data Stream Mining tasks. Information Technology and Management Science, 19(1), 23-28.

24. Женбинг, Х., Бодянский, Е. В., Тыщенко, А. К., & Ткачев, В. Н. (2017). Fuzzy Clustering Data Arrays with Omitted Observations.

25. Kate, R. J. (2016). Using dynamic time warping distances as features for improved time series classification. Data Mining and Knowledge Discovery, 30(2), 283-312.

26. Hu, Z., Mashtalir, S. V., Tyshchenko, O. K., & Stolbovyi, M. I. (2018). Clustering matrix sequences based on the iterative dynamic time deformation procedure. International Journal of Intelligent Systems and Applications, 10(7), 66-73.

27. Wang, D., Lu, X., & Rinaldo, A. (2017). DBSCAN: Optimal Rates For Density Based Clustering. arXiv preprint arXiv:1706.03113.

28. Lyashenko V., Kobylin O., Selevko O. (2020) Wavelet Analysis and Contrast Modification in the Study of Cell Structures Images. International Journal of Advanced Trends in Computer Science and Engineering. 9(4). – 4701-4706.

29. Bodyanskiy, Y., Shafronenko, A., & Mashtalir, S. (2019, May). Online Robust Fuzzy Clustering of Data with Omissions Using Similarity Measure of Special Type. In International Scientific Conference “Intellectual Systems of Decision Making and Problem of Computational Intelligence” (pp. 637-646). Springer, Cham.

30. Mashtalir, S. V., Stolbovyi, M. I., & Yakovlev, S. V. (2019). Clustering Video Sequences by the Method of Harmonic k-Means. Cybernetics and Systems Analysis, 55(2), 200-206.

31. Mashtalir, V., Ruban, I., & Levashenko, V. (Eds.). (2019). Advances in Spatio-Temporal Segmentation of Visual Data (Vol. 876). Springer Nature.

32. Kobylin, O., & Lyashenko, V. (2016). Contrast Modification as a Tool to Study the Structure of Blood Components.

33. Bukhtoyarov, V. V., Tynchenko, V. S., & Petrovsky, E. A. (2019, June). Multi-stage intelligent system for diagnostics of pumping equipment for oil and gas industries. In IOP Conference Series: Earth and Environmental Science (Vol. 272, No. 3, p. 032030). IOP Publishing.

34. Su, J., Vargas, D. V., & Sakurai, K. (2019). One pixel attack for fooling deep neural networks. IEEE Transactions on Evolutionary Computation, 23(5), 828-841.

35. Lyashenko, V., Mohammad, A., Lyubchenko, V., Kobylin, O., & Khan, A. L. V. E. E. R. A. (2016). Image Processing a New Era in the Study of Natural Polymer Composites.

36. Skjæveland, M. G., Forssell, H., Klüwer, J. W., Lupp, D., Thorstensen, E., & Waaler, A. (2019). Pattern-based ontology design and instantiation with reasonable ontology templates. A Higher-Level View of Ontological Modeling, 69.

37. Griffiths, D., & Griffiths, D. (2019). Head First Kotlin: A Brain-friendly Guide. O'Reilly Media.

38. Каретін, В. Д. (2019). Розпізнавання зображень товарів для потреб ТОВ «Техноторг-Дон».

39. Іорін, І. Р. (2019). Спеціалізований обчислювач для реалізації фрактально-векторного способу аналізу зображень.

40. Kowsari, K., Jafari Meimandi, K., Heidarysafa, M., Mendu, S., Barnes, L., & Brown, D. (2019). Text classification algorithms: A survey. Information, 10(4), 150.

41. Tymofieiev, O., & Vechirska, I. (2021). Research and Implementation of the Video Stream Text Recognition Method.