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 перший (бакалаврський)

РОЗРОБКА ПРОГРАМНОГО ЗАСОБУ ПО ОНЛАЙН РОЗПІЗНАВАННЮ ТЕКСТУ НА ЗОБРАЖЕННЯХ
(тема)

Виконав:
студент 4 курсу, групи ІТІНФ-18-2
Бганцов Є.О.
(прізвище, ініціали)

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

Тип програми освітньо-професійна

Освітня програма Інформатика
(повна назва освітньої програми)

Керівник проф. Машталір С.В.
(посада, прізвище, ініціали)

Допускається до захисту

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

2022 р.

Харківський національний університет радіоелектроніки

Факультет Інформаційно-аналітичних технологій та менеджменту
(повна назва)

Кафедра Інформатики
(повна назва)

Рівень вищої освіти перший (бакалаврський)

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

Тип програми освітньо-професійна

Освітня програма Інформатика
(повна назва освітньої програми)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«____»_____ 2022 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Бганцову Євгенію Олександровичу
(прізвище, ім'я, по батькові)

1. Тема роботи Розробка програмного засобу по онлайн розпізнаванню тексту на зображеннях

затверджена наказом по університету від 16 травня 2022 року № 541Ст
2. Термін подання студентом роботи до екзаменаційної комісії 23 травня 2022 р.
3. Вихідні дані до роботи: Методи виділення тексту на зображенні, методі розпізнавання тексту, бібліотека Python Tesseract, бібліотека OpenCV, мова програмування Python, нейронна мережа Tesseract OCR.

4. Перелік питань, що потрібно опрацювати в роботі
1. Огляд методів виділення тексту на зображенні.
2. Основні методи розпезнавання тексту.
3. Вирішення задачі класифікації за допомогою нейронних мереж.
4. Огляд операторів виділення тексту.
5. Аналіз структури нейронних мереж.
6. Розробка алгоритму розпізнавання тексту.
7. Програмна реалізація застосунку по розпізнаванню тексту на зображеннях з використанням Tesseract OCR та мови програмування Python.
8. Тестування розробленої моделі.

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри) Актуальність проблеми розпізнавання тексту, робота з OCR Tesseract, 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	18.04.2022	
2	Аналіз завдання, аналіз літератури	18.04.22-21.04.22	
3	Дослідження існуючих методів вирішення задачі розпізнавання тексту	22.04.22-25.04.22	
4	Огляд існуючих алгоритмів розпізнавання тексту	26.04.22-30.04.22	
5	Програмна реалізація	01.05.22-14.05.22	
6	Проведення тестування розробленої моделі	15.05.22-23.05.22	
7	Оформлення пояснювальної записки	24.05.22-26.05.22	
8	Перевірка на плагіат	27.05.22	
9	Рецензування	28.05.22	
10	Підготовка презентації та доповіді	29.05.22-30.05.22	
11	Занесення роботи в електронний архів	31.05.22	
12	Попередній захист кваліфікаційної роботи	31.05.22	

Дата видачі завдання 18 04 2022 р.

Студент _____

(підпис)

Керівник роботи _____ проф. Машталір С.В.

(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ/ABSTRACT

Пояснювальна записка до кваліфікаційної роботи: 46 с., 17 рис., 30 джерел.

РОЗПІЗНАВАННЯ ТЕКСТУ, МЕТОДИ РОЗПІЗНАВАННЯ ТЕКСТУ, НЕЙРОННІ МЕРЕЖІ.

Об’єктом роботи є послідовність зображень з текстом.

Метою роботи є розробка программного забеспечення на основі використання нейронної мережі, яка дозволяє розпізнавати ознаки символів у тексті та формувати друкований текст згідно з зображенням.

У рамках кваліфікаційної роботи був розроблений і реалізований метод розпізнавання тексту зі зображень за допомогою нейронної мережі Tesseract ORC та допоміжних бібліотек.

Також було основні методи розпізнавання тексту та технології розробки застосунків, які використоувуються у цій сфері. Наведено спосіб вирішення такої задачі за допомогою нейронної мережі за бібліотек Python, які значно полегшують работу з розпізнаванням тексту. Були здобуті навички з мови програмування Python.

Результатом данної роботи є розпізнаний текст текстового формату в блокноті, вивчено процесс розпізнавання тексту та його алгоритми.

TEXT RECOGNITION, TEXT RECOGNITION METHODS, NEURAL NETWORK.

The object of the work is a sequence of images with text.

The aim of the work is to develop software based on the use of neural network, which allows you to recognize the characteristics of characters in the text and form printed text according to the image.

As part of the qualification work, a method of text recognition from images was developed and implemented using the Tesseract ORC neural network and additional libraries.

There were also basic text recognition methods and application development technologies used in this area. There is a way to solve this problem using a neural network for Python libraries, which greatly facilitate the work with text recognition. Python programming language skills were acquired.

The result of this work is the recognized text of the text format in the notebook, the process of text recognition and its algorithms are studied.

ЗМІСТ

Перелік умовних позначень, символів, одиниць, скорочень і термінів.....6
Вступ.....7
1 Огляд основних методів розпізнавання тексту.....7
1.1 Методи розпізнавання тексту.....8
1.2 Основні методи розпізнавання тексту.....8
1.2.1 Структурний метод.....8
1.2.2 Шаблонний метод.....9
1.2.3 Ознакові методи.....10
1.3 Еталоні методи.....10
1.3.1 Нейронні мережі.....11
1.3.2 Алгоритм скелетизації.....12
1.3.3 Поточкове процентне порівняння з еталоном.....13
1.3.4 Інваріантні числа.....13
1.4 Шрифт-залежні та шрифт-незалежні алгоритми.....13
1.5 Постановка задачі.....20
2 Оптичне розпізнавання символів.....21
2.1 Задача розпізнавання символів.....21
2.2 Підпроцеси OCR.....22
2.2.1 Попередня обробка зображення.....23
2.2.2 Сегментація символів.....30
2.2.3 Розпізнавання символів.....31
2.2.4 Постобробки.....33
3 Комп'ютерна модель фільтрації зображень.....34
3.1 Вибір середовища програмної реалізації.....34
3.2 Програмна реалізація.....36
3.3 Тестування розробленої моделі.....40
Висновки.....42
Перелік джерел посилання.....43

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

СТЗ – система технічного зору

ІП – інваріантна пряма

МН – машинне навчання

OCR – Optical Character Recognition

ANN – Artificial Neural Networks

ADC – Analog-to-Digital Conversion

NN – Neural Networks

LSTM – Long Short-Term Memory

VGSL – Variable-size Graph Specification Language

IDE – Integrated Development Environment

RNN – Recurrent Neural Network

CNN – Convolutional Neural Network

ВСТУП

Комп'ютерна графіка поділяється на три основні області: візуалізація, обробка зображень і розпізнавання образів.

Візуалізація – це відтворення зображення на основі опису (моделі). Це може бути, наприклад, відображення графіки, діаграм, імітація тривимірної віртуальної реальності в комп'ютерних іграх, архітектурно-проектних системах.

Основним завданням розпізнавання образів є отримання семантичного опису описуваних об'єктів. Мета ідентифікації може бути різною: виділення окремих пікселів і класифікація зображення в цілому. У певному сенсі проблема розпізнавання є протилежною проблемі візуалізації. Сфери застосування системи розпізнавання тексту, створення тривимірних моделей людини з фотографій.

Актуальність роботи багато в чому полягає в його застосуванні до життя. Розпізнавання тексту широко використовується для перетворення книг і документів в електронну форму, для автоматизації систем корпоративного обліку або для публікації тексту на вебсайті. OCR дозволяє редагувати текст, шукати слово чи фразу, зберігати їх у більш компактному форматі, відображати чи друкувати матеріал без втрати якості, аналізувати інформацію та використовувати електронний переклад, форматування чи перетворення тексту. Оптичне розпізнавання тексту є проблемою в розпізнаванні образів, штучного інтелекту та комп'ютерного зору.

1 ОГЛЯД ОСНОВНИХ МЕТОДІВ РОЗПІЗНАВАННЯ ТЕКСТУ

1.1 Методи розпізнавання тексту

В наш час у світі існує дуже багато різних програм розпізнавання символів, але слід зауважити, що вміння читати людиною друкований текст не високої якості все ще перевершує можливості комп'ютера. Кожен надрукований текст має деяку особливу властивість – шрифт, яким він набирається. Відповідно, існують два типа алгоритмів для розпізнавання символів: шрифт-залежних і незалежних від шрифта. Шрифт-залежні алгоритми використовують важливу інформацію про шрифт з літерами. Це означає, що програма повинна мати повний зразок тексту, надрукованого цим шрифтом. Програма вимірює і аналізує різні характеристики шрифтів і вставляє їх у свою базу еталоннів.

1.2 Основні методи розпізнавання тексту

1.2.1 Структурний метод

Структурне розпізнавання зберігає інформацію не про поточне написання символу, а про його топологію. Стандарт має інформацію про відносне розташування окремих компонентів. Особливістю цього медоту є опис об'єктів з точки зору їх структури з виділенням окремих складових елементів і зв'язків між цими елементами. У такому підході символ описано як граф, вузлами якого є елементи вхідного об'єкта, а дугами – просторові відносини між ними. Структурними елементами є лінії, що створюють символ. До переваг структурних методів розпізнавання можна віднести інваріантність щодо типів і розмірів шрифтів. Саме тому ці методи будуть ефективними при розпізнаванні рукописного тексту. При застосуванні цього методу будуть такі невідомі ознаки, як розмір букв, які розпадалися, і навіть шрифти, на яких було надруковано. Проте ідентифікація є головною проблемою цього методу, яка

містить певні дефекти, наприклад, розрив лінії або злиття сусідніх ліній, а також невисока швидкість [1].

1.2.2 Шаблонний метод

Основна ідея полягає у порівнянні зображення окремого символу з усіма шаблонами, що є в наявні у базі, і вибір шаблону з найменшою кількістю відмінностей від вхідного зображення. Вирішення про належність зображення символу з тестової вибірки до певного класу символів здійснюються за критерієм мінімуму або максимуму деякої метрики подібності зображення символу і його шаблону.

Шаблонні методи перетворюють зображення окремого символу в растрове, порівнюють його зі всіма шаблонами, наявними в базі і вибирають шаблон з найменшою кількістю крапок, відмінних від вхідного зображення. Шаблонні методи досить стійкі до дефектів зображення і мають високу швидкість обробки вхідних даних, але надійно вони розпізнають тільки ті шрифти, шаблони яких їм «відомі». І у випадку коли розпізнаний шрифт трохи відрізняється від еталонного, шаблонні методи можуть робити помилки навіть при обробці доволі якісних зображень [2].

Шаблонні системи мають високу швидкість обробки вхідних даних, але чітко розпізнають лише ті шрифти, шаблони яких наявні в їх базі. Даний підхід вимагає створення шаблону для кожного шрифту. Доцільно застосовувати ці методи для розпізнавання текстових документів, які мають чіткий шрифт, малий відсоток дефектів та шумів.

1.2.3 Ознакові методи

В цьому методі замість символів аналізується набір властивих їм певних ознак. Зображення приводиться у відповідність N-вимірний вектор ознак. Розпізнавання полягає в порівнянні його з набором еталонних векторів тієї ж розмірності. Для розпізнавання символів можуть використовуватися різні системи ознак. Задача прийняття рішення про приналежність образу до того чи іншого класу на підставі аналізу обчислювальних ознак, має ряд математичних рішень в рамках детерміністичного та ймовірнісного підходів. Основна перевага ознакових методів – хороша узагальнююча здатність та стійкість до змін форми символу, простота реалізації. Недолік методу це велика чутливість до різних дефектів зображення. Крім того, ознакові методи мають ще один недолік – на етапі становлення ознак відбувається втрата деякої інформації про символ. Тому, ці методи доцільно застосовувати при розпізнавання тексту з низьким відсотком дефектів, або застосовуватися у парі з іншими методам [3].

Виділення ознак проходить незалежно, тому інформація про взаємне розташування елементів символів втрачається.

1.3 Еталоні методи

Еталоні методи передбачають порівняння заданого, не розпізнаного символу з набором деяких еталонів. Для цього використовують нейронні мережі, які необхідно заздалегідь заповнити еталонами. Існують кілька алгоритмів порівняння тексту з еталоном [4].

Найпростіший варіант це попіксельне порівняння, однак для нього необхідні рівні по розмірам зображення, що порівнюються. Інші варіанти – накладення та накладення зі зміщенням, у яких зображення ставляться у відповідність одне одному. Методи еталонного порівняння рукописного та друкованого тексту, на перший погляд схожі, однак суттєво

відрізняється. Еталони рукописного тексту лише грубі зразки, на відміну від літер шрифту. Це значно збільшує вірогідність помилки. У типовій системі оптичного розпізнавання символів (OCR) введені символи зчитуються та оцифровуються за допомогою оптичного сканера. Після цього кожен символ піддається локалізації і розподілу, і отримана матриця підлягає попередній обробці, тобто згладжування, нормалізація і фільтрація. В результаті попередньої обробки розрізняють характерні ознаки, після яких здійснюється класифікація.

Основні підходи та рішення: Алгоритм скелетизації, Нейрона мережа, Інваріантні числа, поточкове процентне порівняння з еталоном [5].

1.3.1 Нейронні мережі

Напрямок був дуже популярний у 60-х – 70-х роках, тому що велика кількість вимагає солідарних обтяжливих потужностей, які зазвичай відсутні на простих платформах. Однак слід мати на увазі, що нейронні мережі іноді дають дуже цікаві результати, завдяки яким вони не мають будь-якої структури, тим більше що деякі нейронні мережі здатні розпаковувати зображення інваріантності. Наприклад, мережі, засновані на неокогнетронах, можуть ідентифікувати деякі характеристики, а також їх розривати, як якщо б вони поверталися.

Нейронні мережі це клас аналітичних методів, побудованих на гіпотетичних принципах навчання мислячих істот та функціонування мозку і дозволяють прогнозувати значення деяких змінних у нових спостереженнях за даними інших спостережень (для тих самих або інших змінних) після проходження етапу так званого навчання на існуючих даних. Сказати лише те, що у математичному сенсі нейронна мережу – це лише модель біологічного визначення [6].

Принцип роботи нейронної мережі такий, що повчивши на вхідний шар нейронів, нове зображення мережа реагує імпульсом того чи іншого нейрона. Оскільки всі нейрони названі значеннями букв, отже, нейрон, що зреагував, і несе відповідь розпізнавання. Заглиблюючись у термінологію мереж можна сказати, що нейрон крім виходу має множину входів. Ці входи описують значення пікселя зображення. Тобто якщо є зображення 16×16, входів у мережі має бути 256 [7].

Кожен вхід сприймається з певним коефіцієнтом і в результаті, після закінчення розпізнавання на кожному нейроні накопичується певний заряд, чим заряд буде більший той нейрон і випустить імпульс.

Але щоб коефіцієнти входів були правильно налаштовані необхідно спочатку навчити мережу. Цим займається окремий модуль навчання. Даний модуль бере чергове зображення з навчальної вибірки та згодовує мережі.

Мережа аналізує всі позиції темних пікселів і вирівнює коефіцієнти мінімізуючи помилку збігу шляхом градієнта, після чого певному нейрону зіставляється це зображення.

По закінченню навчання кожен нейрон схожий на полотно художника, де на місцях у яких найчастіше зустрічалися чорні пікселі, найбільш темна фарба там значення заряду більше, а там, де рідше зовсім світлий тон [8].

1.3.2 Алгоритм скелетизації

Це спосіб розпізнавання одиночних бінарних зображень, заснований на побудові скелетів цих зображень і виборі вузлів і скелета ребер. Потім, відповідно до кількості вузлів і чисел співвідношення ребер, будується таблиця відповідності зображенням. Так, наприклад, скелет кола буде одним вузлом, скелет літери «П» – три ребра і два вузли, а ребра – 2: 2: 1. У програмуванні цей метод має кілька можливих реалізацій [9].

1.3.3 Поточкове процентне порівняння з еталоном

Повинна бути якась попередня обробка, щоб отримати інваріантність по відношенню до розміру і положення, потім порівняння з підготовленою базою даних стандартів зображень – якщо збіг більше певної позначки, то ми розглядаємо зображення, яке слід розпізнати [10].

1.3.4 Інваріантні числа

З геометрії зображень можна вибрати декілька чисел, інваріантних по відношенню до розміру і повороту зображень, після чого можна зробити таблицю, що відповідає цим числам конкретного зображення (майже як у алгоритмі скелетонізації) [11]. Прикладами інваріантних чисел є число Ейлера, ексцентриситет, орієнтація (в сенсі розташування головної осі інерції відносно чого-небудь інваріантного) [12].

1.4 Шрифт-залежні та шрифт-незалежні алгоритми

Оптичне розпізнавання символів відбувається певного шрифту.

Недоліком такого підходу є те що алгоритм повинен заздалегідь знати шрифт, який він представляє для розпізнавання, тобто, він повинен зберігати в базі даних такі різні характеристики як: якість розпізнавання тексту, введеного довільним шрифтом, буде прямо пропорційна співвідношенню характеристик цього шрифту з шрифтами, доступні в базі даних програми.

Ці фактори обмежують універсальність таких алгоритмів. Для роботи з програмою розпізнавання потрібен блок налаштування певного шрифту. Очевидно, що ця одиниця внесе свою частину помилки в інтегральній оцінці якості розпізнавання або функції [13]. Доведеться встановити шрифт для

користувача. Програма, заснована на алгоритмі розпізнавання символів шрифту вимагає від користувача мати спеціальні знання про шрифти взагалі, про їхні групи і відмінності один від одного, про шрифти, які друкуються. Загального способу немає дізнатися, на яких шрифтах був надрукований цей документ. З іншого боку, підхід шрифту має ту перевагу, за якою він є активно використовувати і, мабуть, використовуватиметься в майбутньому. А саме, маючи деталізовану пріоритетну інформацію, можливість будувати досить точні та надійні алгоритми розпізнавання. Загалом, з побудови алгоритму розпізнавання шрифтів, на відміну від без шрифту надійність розпізнавання символів інтуїтивно зрозуміла і математично виражена величина. Це значення визначається як відстань у будь-який метричний простір від представленого позначення програми в процесі навчання, до персонажа, який програма намагається визнати. Другий клас алгоритмів не залежить від кадру або не залежить від шрифту, тобто алгоритми, які не мають апріорного знання символів, що надходять до них до входу. Ці алгоритми вимірюють і аналізують різні характеристики, які властиві буквам як такі, незалежно від шрифту та абсолютного розміру, як вони друкуються. У граничному випадку для не-шрифт-залежних. Можливо, немає алгоритму навчання. В цьому випадку характеристики символів вимірюються, кодуються і розміщуються в самій програмі людиною. Однак на практиці зустрічаються рідкісні випадки, коли такий шлях можливий [14].

Більш загальний спосіб створення бази даних характеристик полягає у вивченні програми на зразку реальних символів. Недоліком такого підходу є нижча якість розпізнавання, ніж у алгоритму шрифту. Це пов’язано з тим, що рівень узагальнення з характерними вимірюваннями значно більші, ніж у випадку з шрифт-залежними алгоритмами.

Переваги такого підходу тісно пов’язані з його недоліками. Основні з переваг це універсальність. Це означає, з одного боку, можливість застосування цей підхід у випадках великої різноманітності символів – з іншого боку, через характеристики, встановлені в них.

Підсумовуючи, такі алгоритми можуть екстраполювати накопичені знання межі навчальної вибірки, тобто послідовно розпізнавати образи, здавалося б, далеких від тих, які були присутні у навчальній вибірці. Процес вивчення незалежних алгоритмів. Звичайно, є більш простим і інтегрованим у сенсі навчання зразок не фрагментується в різні класи. Для цього немає необхідності підтримувати в базі даних характеристики різних умов для спільного їх існування.

Проявом технологічної здатності є також те, що вона часто з'являється практично повністю автоматизовані процедури навчання. У випадку, якщо програма побудована на алгоритмах без ковзання, користувачу не обов'язоково знати щось про сторінку, яку він хоче ввести в програму. Це також спрощує інтерфейс користувальницької програми для відшкодування вартості опцій та діалогів навчання персоналу та управління базою даних характеристик. Таким чином, це призведе до збільшення кола користувачів, зокрема для людей з мінімальною комп'ютерною грамотністю таке програмне забезпечення буде дуже корисним [15].

При розпізнаванні символів досить широко використовуються штучні нейронні мережі. Алгоритми, зроблені для нейронних мереж щоб розпізнавати символи, не рідко створюються в такий спосіб: картинку символу, що є вхідним зображенням для зчитування, приводиться до певного стандартного розміру. Як правило, зображення приводиться до розміру 16×16 пікселів.

Значення яскравості в вузлах нормалізованого зображення використовуються як вхідні параметри нейронної мережі. Кількість вихідних параметрів нейронної мережі дорівнює кількості розпізнаваних символів. Результатом розпізнавання є символ, який відповідає найвищому значенню вектора джерела нейронної мережі. Підвищення надійності таких алгоритмів зазвичай пов'язано з пошуком більш інформативних вхідних ознак або з ускладненням структури нейронної мережі.

Надійність розпізнавання і необхідність програми в обчислювальних ресурсах багато в чому залежать від вибору структури і параметрів нейронної

мережі. Зображення фігур зменшуються до одного розміру (28×28 пікселів). Отримане зображення подається на вхід нейронної мережі, яка має три внутрішніх рівня і 10 вузлів верхнього рівня. Нижчі шари мережі не повністю пов'язані між собою. Нижні вузли мають спільний набір шкал. Все це, згідно з розробкою розробників, повинно підвищити здатність нижчих рівнів мережі розрізняти основні функції зображень. Для збільшення здатності мережі до узагальнення та зменшення кількості необхідних обчислень і пам'яті здійснюється видалення невикористаних ваг. В результаті кількість незалежних параметрів зменшується в чотири рази. Навчання нейронної мережі здійснюється на наборі 7300 символів, тест на безліч 2000 символів. Помилки розпізнавання становлять приблизно 1% на навчальному наборі і 5% на тесті [16].

В якості вхідних параметрів нейронної мережі замість значень яскравості в вузлах нормалізованого растру можуть бути використані значення, що характеризують різницю в яскравості. Такі вхідні параметри дозволяють краще розподіляти межі літер. Об'єкти розпізнавання зменшуються до 16×16 пікселів. Після цього вони піддаються подальшій обробці з метою ізоляції ділянок з найбільшими варіаціями яскравості.

Одним з широко використовуваних методів підвищення точності розпізнавання є одночасне використання декількох різних модулів розпізнавання і подальша інтеграція результатів (наприклад, шляхом голосування). Дуже важливо, щоб алгоритми, що використовуються цими модулями, були якомога більш незалежними. Цього можна досягти за рахунок використання модулів розпізнавання, які використовують принципово різні алгоритми розпізнавання, а також спеціальний вибір навчальних даних.

Один з цих методів був запропонований кілька років тому і базувався на використанні трьох модулів розпізнавання (машин). Перший автомобіль навчається звичайним способом. Друга машина дізнається про символи, які були відфільтровані першою машиною, так щоб друга машина бачила суміш символів, 50% з яких були визнані першою машиною правильно і 50%

неправильно. Нарешті, третя машина дізнається про символи, на яких відрізняються результати розпізнавання 1-го і 2-го автомобілів. Під час тестування до всіх трьох машин застосовуються розпізнавані символи. Оцінки, отримані на виході всіх трьох автомобілів, складаються. Символ, який отримав найбільшу загальну кількість балів, видається в результаті розпізнавання.

Як правило, алгоритм розпізнавання заснований на поділі растру з зображенням букв основних ознак і подальшого використання штучної нейронної мережі для оцінки близькості вхідного зображення з символами з заданого набору букв. Результатом роботи є набір оцінок, що відображають ступінь близькості розпізнаваного символу з символами з заданого набору символів [17].

Набір розпізнаваних символів може включати букви і цифри. Входи для розпізнавання символів перетворюються на один розмір.

Відмінною особливістю реалізованого алгоритму є використання нейронної мережі з досить великою кількістю вхідних функцій. У вихідному зображенні виділяються основні функції, що характеризують відмінності яскравості в вузлах растру. Нейронна мережа має один внутрішній рівень, що містить 100 вузлів і є універсально пов’язаним, тобто кожен вузол внутрішнього рівня з’єднаний з усіма вхідними вузлами, і кожен вузол верхнього рівня з’єднаний з усіма вузлами внутрішнього рівня. Для зменшення кількості обчислень при розпізнаванні для кожного розпізнаваного символьного зображення використовуються не всі вхідні ознаки, а лише частина, тобто вектор вхідних параметрів нейронної мережі сильно розбавлений.

Навачання нейронної мережі є звичайним способом, тобто алгоритмом, який використовується для зворотного поширення помилки. Програма навчання отримує вхідний файл із зображеннями символів. При навчанні символи з цієї бази даних є циклічними. Для кожного зображення з бази даних виділяються первинні атрибути, після чого виконуються прямі і зворотні проходи через мережу. Зміна ваги мережі під час тренування проводиться після кожного символу. Крок зміни ваги мережі є постійним. Погано впізнавані

символи розглядаються частіше, ніж інші, для прискорення та вдосконалення навчання. Використовується кеш, який зберігає важкодоступні зображення. Растри для навчання вибираються як з вхідного файлу, так і з кешу. Вибір символу з кешу відбувається з урахуванням якості його розпізнавання, тобто погано впізнавані символи вибираються частіше.

Крім того, при навчанні мережі використовується регуляризація ваги мережі, тобто вводиться їх експоненціальне вицвітання.

Якість розпізнавання залежить не тільки від алгоритмів, що використовуються програмами розпізнавання та навчання нейронної мережі, але й від того, як вивчається нейронна мережа. На якість підготовки нейронної мережі впливають наступні фактори: параметри бази з тренувальним растмібом, розмір, метод вибору растрів, порядок розташування растрів в основі, наявність брудних символів і помилок у розмітці.

Різні фактори оптимізації можна використовувати на різних етапах навчання:

– крок зміни коефіцієнтів мережі;

– використання регуляризації мережі;

– історія навчання мережі;

– використання додаткового шуму й перекручувань символів;

– момент зупинки навчання. Бажано уникати як недостатнього навчання мережі, так і перенавчання;

– розмір кешу поганих растрів і відносна частота вибору растрів з навчальної бази даних і з кешу поганих символів.

Параметри навчання взаємозалежні і повинні вибиратися послідовно. Наприклад, при невеликій кількості навчальних ресурсів, використання спотворень символів може призвести до поліпшення якості навчання, а зі збільшенням розміру бази даних призводить до її погіршення. Використання кеш-пам’яті поганих символів на початку тренування не має особливого сенсу. Навпаки, після декількох проходів на основі тренувальних символів, більшість символів з бази визнаються дуже високою надійністю. Зміна ваги мережі

відбувається в основному за рахунок растра, що міститься в кеші поганих символів.

Регуляризація (тобто введення експоненціального вицвітання ваги в навчання) призводить до деякого зниження якості розпізнавання. Однак використання дуже малого коефіцієнта затухання може підвищити стабільність мережі без помітних втрат для розпізнавання [18].

Щоб визначити найкращий час для зупинки мережі, можна періодично перевіряти якість розпізнавання на невеликій, незалежній базі даних.

Порівняння якості алгоритмів розпізнавання різних символів ускладнюється тим, що відносна величина кількості правильно розпізнаних символів значною мірою залежить від конкретної бази даних, на якій проводиться тестування. На якість розпізнавання впливають також: обсяг набору розпізнаваних персонажів, технологія викладання нейронної мережі, методологія та алгоритми виділення первинних характеристик, технологія підготовки бази даних та інші фактори.

Алгоритм може бути поліпшений шляхом пошуку більш адекватного представлення структурних ознак розпізнаваних символів. Використання великої навчальної бази даних і збільшення пам’яті нейронної мережі також може забезпечити деяке поліпшення якості розпізнавання. Проектована система повинна працювати в режимі, близькому до реального часу, і тому розроблений алгоритм повинен бути досить швидким і в той же час мати достатню точність розпізнавання.ABBYY розробили спеціальний алгоритм MDA (multilevel document analysis, багаторівневий аналіз документа). Структура сторінки аналізується методом зверху-донизу (від складових елементів до окремих символів), а відтворення електронного документа після закінчення розпізнавання відбувається знизу-догори, проте на всіх рівнях додатково діє механізм зворотнього зв’язку [19].

Велика кількість сучасних OCR процюють на трьох рівнях: символів, слів та сторінок. Однак ABBYY, відповідно до принципів IPA, ввела в FineReader ще один рівень – рівень всього багатосторінкового документа.

У першу чергу це знадобилося для коректного відтворення логічної структури, яка в сучасних документах стає складнішою. Через це була розроблена ADRT (Adaptive Document Recognition Technology) – технологія аналізу і синтезу документа на логічному рівні. В кінцевому результаті вона допомагає зробити результат роботи FineReader максимально схожим на оригінал [20].

Серед подібних до Abby FineReader систем можна назвати:

– OmniPage;

– SimpleOCR;

– CuneiForm;

– Readiris.

Окремо слід виділити Tesseract ORC– це система розпізнавання текстів, що у 1985-1994 рр. розроблялася Hewlett-Packard, а з 2006 є вільною і поширюється компанією Google та дуже стрімко розвивається у останні роки.

1.5 Постановка задачі

Актуальним завданням для обробки і розпізнавання тексту э знаходження геометричних примітивів. У зв’язку з цим ставиться завдання розробки та налаштування нейронної мережі для знаходження на картинці елементів тексту та їх розпізнавання [21].

Об’єктом роботи є послідовність зображень з текстом.

Метою роботи є розробка программного забеспечення на основі використання нейронної мережі, яка дозволяє розпізнавати ознаки символів у тексті та формувати друкований текст згідно з зображенням.

Для досягнення мети необхідно вирішити такі завдання:

– провести аналіз існуючих методів розпізнавання елементів тексту;

– розробити алгоритм знаходження тексту;

– розробити алгоритм розпізнавання символів;

– реалізувати застосунок для розпізнавання тексту.

2 ОПТИЧНЕ РОЗПІЗНАВАННЯ СИМВОЛІВ

Оптичне розпізнавання тексту, з англійскої optical character recognition, (OCR) – це механічне або електронне переведення збереженого рукописного, машинописного або друкованого тексту в послідовність кодів, що використовують для представлення в текстовому редакторі [22, 23].

2.1 Задача розпізнавання символів

Оптичне розпізнавання символів лежить в основі цифрової трансформації багатьох компаній. Перший етап успішної диджиталізації – оцифрування всіх паперових документів, з якими ви працюєте та їх структуроване збереження, щоби зробити їх доступними для всіх співробітників вашої організації, яким ця інформація необхідна для виконання їхньої роботи.

Оптичне розпізнавання тексту дозволяє:

– редагувати текст;

– здійснювати пошук по словах або фразах;

– зберігати його в компактнішій формі;

– демонструвати або роздруковувати матеріал, не втрачаючи якості;

– аналізувати інформацію;

– застосовувати до тексту електронний переклад, форматування або перетворення в мовлення.

Іншими словами, системи OCR перетворюють двовимірне зображення тексту, яке може містити машинно надрукований або рукописний текст, з його зображення у текст, який читається машиною [23].

2.2 Підпроцеси OCR

OCR як процес зазвичай складається з кількох підпроцесів (рис. 2.1), які виконуються якомога точніше [24].

Підпроцесами є:

– попередня обробка;

– сегментація;

– розпізнавання символів;

– постобробки.

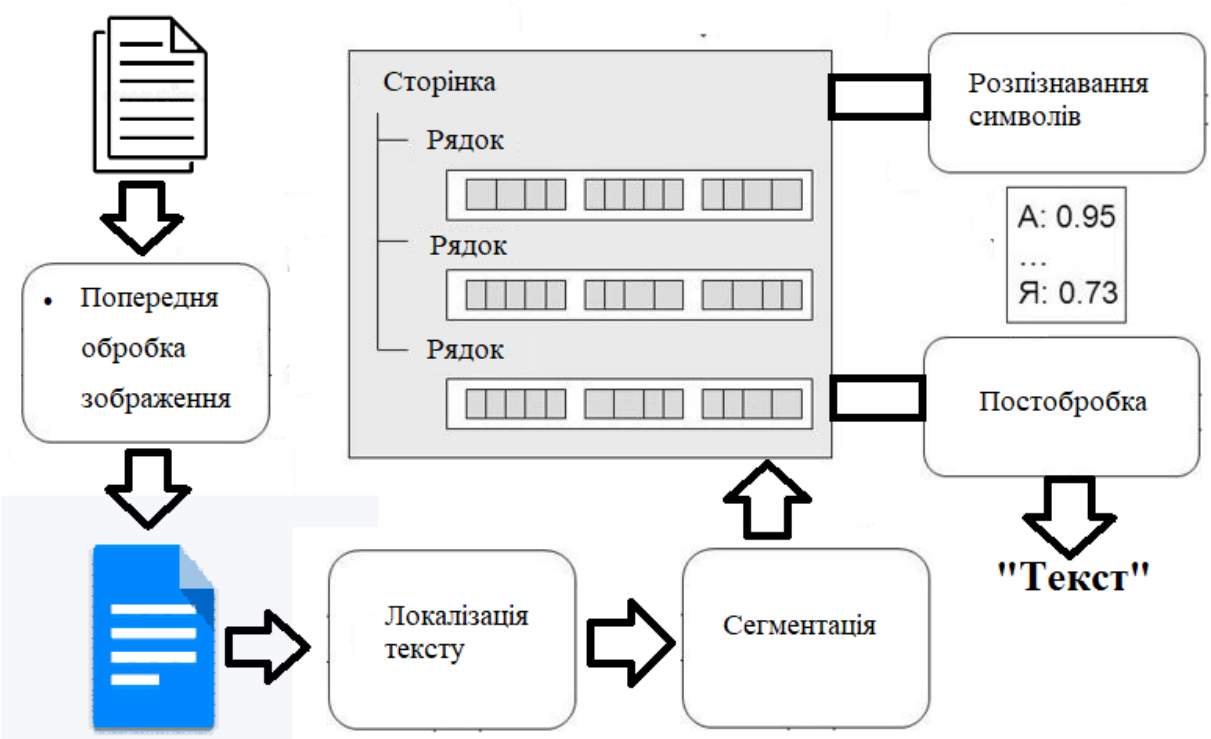

Рисунок 2.1 – Процесс розпізнавання тексту

Існує багато методів для покращення якості зображення, отриманого на основі перетворення просторового сигналу. Наведемо опис декількох методів, використаних для покращення якості зображення.

2.2.1 Попередня обробка зображення

Розглянуто задачі зміни контрасту, яскравості перетворення гістограми розподілу яскравостей, фільтрації зображення. Ці перетворення дозволяють підвищити ефективність візуального представлення та розпізнавання зображень.

На рисунку 2.2 зображена характеристика передачі рівнів, яка потрібна для підвищення контрасту типових безперервних малоконтрастних зображень. Користуючись фотографічними методами можна здійснити коригування таких зображень, проте реалізувати довільну характеристику передачі рівнів з високою точністю зазвичай важко. У разі цифрових зображень отримати необхідну характеристику передачі рівнів відносно легко. але за знаходженні нелінійного оператора слід враховувати помилки квантування. Нехай вихідне зображення, проквантоване на рівні J, має обмежений діапазон яскравостей зображений на рисунку 2.3.

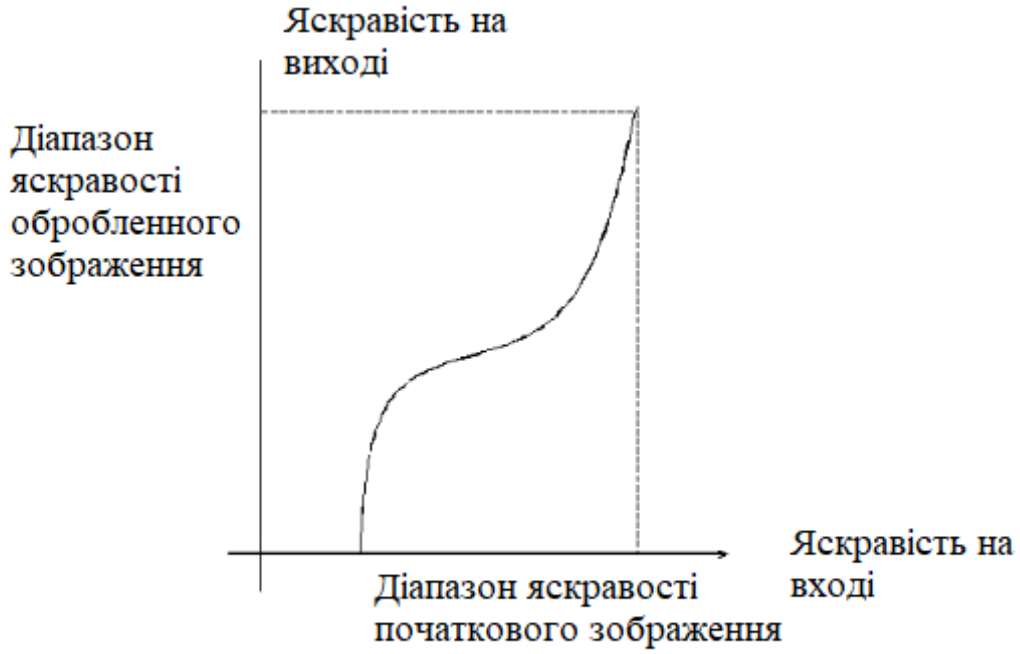

Рисунок 2.2 – Характеристика передачі рівнів зображення

Рисунок 2.3 – Передача рівнів дискретизованого полутонового зображення

Передбачається, що вихідне зображення також квантується на рівні *J*, а перетворення лінійно. Як видно з рисунку 2.3, кожен обраний вихідний рівень є найближчим до рівня, що відповідає рівню. Очевидно, що в діапазоні яскравостей вихідного зображення деякі рівні не використовуватимуться, тому окремі перепади яскравості вихідного зображення будуть перевищувати відповідні перепади вихідного зображення. Внаслідок цього можуть виникнути помітні хибні контури. Якщо вихідне зображення квантувати з більшим числом рівнів, ніж вхідне, можна отримати рівномірне розміщення вихідних рівнів і завдяки цьому зменшити ефект появи помилкових контурів.

Діапазон яскравості вихідного зображення, підданого цифровій обробці, може відрізнятися від діапазону яскравостей вихідного зображення. Крім того, в діапазоні числових значень яскравості обробленого зображення можуть виявитися негативні значення, які не можуть мати прямої відповідності до фізичних яскравостей.

На рисунку 2.4 зображені два можливі способи приведення діапазону яскравостей вихідного зображення згідно з діапазоном яскравостей вихідного зображення. Відповідно до першого, оброблене зображення лінійно

відображається таким чином, щоб повністю охопити данний йому діапазон яскравостей. Другий спосіб використовується для обмеження екстремальних значень яскравості обробленого зображення максимальним і мінімальним пороговими значеннями. Цей спосіб з високою ймовірністю забезпечує більш високу суб'єктивну якість зображення, особливо тоді коли оброблене зображення містить мало елементів з перевищенням рівнів обмеження. У програмах передбачається можливість введення обмеження для фіксованого відсотка рівнів яскравостей [25].

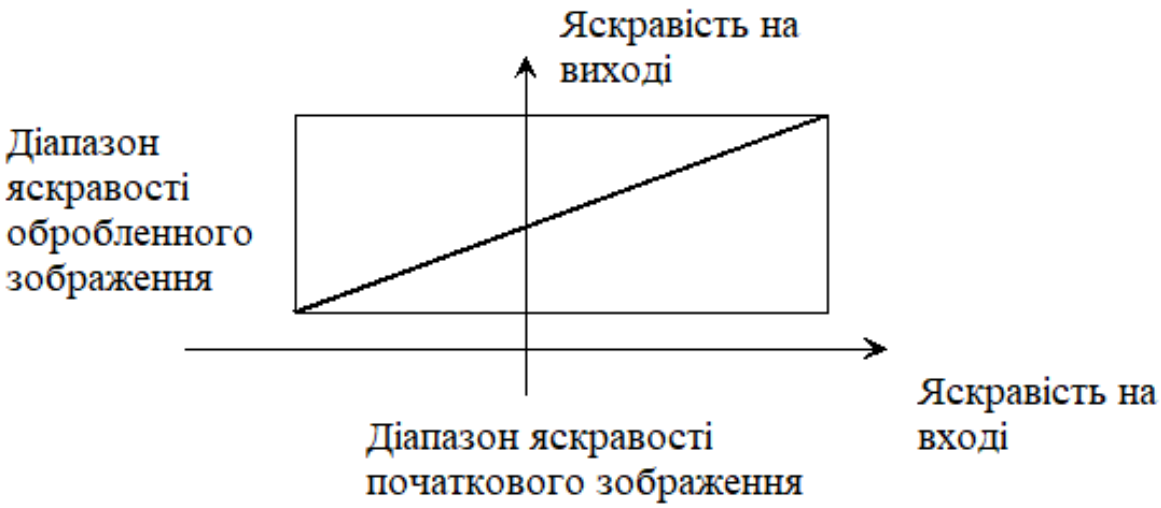

Рисунок 2.4 – Зміна яскравості зображень

У деяких випадках обробки зображень доцільніше користуватися монотонно спадаючою або монотонною характеристикою.

На рисунку 2.5 зображено характеристику передачі рівнів, яка забезпечує обіг шкали яскравостей. Цю характеристику доцільно застосовувати коли дисплей має суттєву нелінійність в області чорного. Таким чином темні ділянки вхідного зображення будуть переведені до світлих, котрі відповідають лінійній ділянці шкали яскравостей у дисплеї.

Рисунок 2.5 – Зміна характеристик передачі рівнів

Рисунок 2.6 зображує характеристику передачі рівнів, призначену для пилоподібного контрастного масштабування. Таке перетворення часто використовують, щоб отримати зображення з широким динамічним діапазоном на екрані дисплея з обмеженим динамічним діапазон.

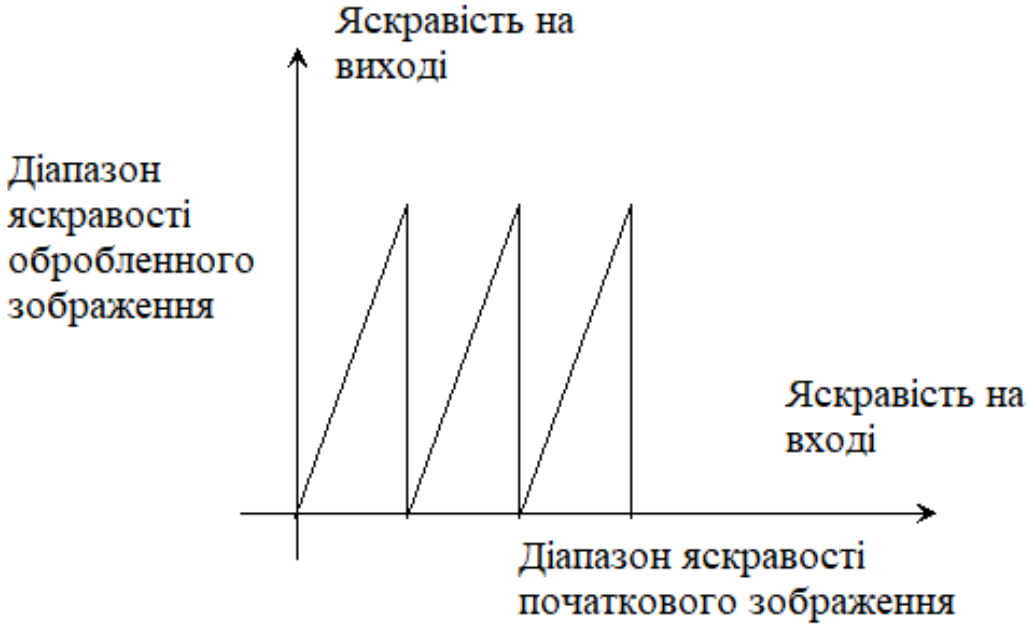

Рисунок 2.6 – Пилоподібне контрастне масштабування

Зображення може ушкоджуватися шумами та перешкодами різного походження, наприклад шумом відеодатчика, шумом зернистості фото матеріалів та помилками в каналі передавача. Їх вплив можна мінімізувати, користуючись класичними методами статистичної фільтрації. Інший можливий підхід ґрунтується на використанні інших евристичних методів просторової обробки.

Шуми відеодатчиків або помилки в каналі передачі зазвичай виявляються на зображенні як розрізнені зміни ізольованих елементів, що не мають просторової кореляції. Спотворені елементи часто дуже помітно відрізняються від сусідніх елементів. Це спостереження послужило основою багатьох алгоритмів, які забезпечують придушення шуму [26].

При застосуванні цифрової фільтрації зображень суттєво покращується якість зображення, яке одержується в процесі зондування. Далі застосовують лінійну фільтрацію для згладжування шумів на зображенні (низькочастотна фільтрація), виявлення меж об'єктів при використанні високочастотної фільтрації, а також метод медіанної фільтрації, який використовується для усунення перешкод імпульсного типу.

У тому разі якщо яскравість елемента перевищує середню яскравість групи ближніх елементів на деяку порогову величину, яскравість елемента замінюється на середню яскравість:

$$[x - \frac{1}{8}\sum_{i=1}^{8} O_i] > \varepsilon, \text{то } [x = \frac{1}{8}\sum_{i=1}^{8} O_i]. \qquad (2.1)$$

Через те що шум просторово декорельований, у його спектрі містяться більш високі просторові частоти, ніж у спектрі звичайного зображення. Отже, проста низькочастотна просторова фільтрація може бути ефективним засобом згладжування шумів. Масив Q розміру $M \times M$ вихідного зображення формується шляхом дискретної згортки масиву F розміру $N \times N$ вихідного зображення зі згладжуючим масивом H розміру $L \times L$ згідно з формулою.

$$Q(m_1, m_2) = \sum_{n_1} \sum_{n_2} F(n_1, n_2) H(m_1 - n_1 + 1, m_2 - n_2 + 1). \quad (2.2)$$

Згладжування шуму забезпечується низькочастотним фільтруванням за допомогою масиву H з позитивними елементами. Нижче наведені масиви трьох різновидів, що згладжують, часто звані шумоподавлюючими масками:

$$H = \frac{1}{9} \begin{vmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{vmatrix}, \quad (2.3)$$

$$H = \frac{1}{10} \begin{vmatrix} 1 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 1 \end{vmatrix}, \quad (2.4)$$

$$H = \frac{1}{16} \begin{vmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{vmatrix}. \quad (2.5)$$

Масиви нормовані на отримання одиничного коефіцієнта передачі, щоб процедура придушення шуму не викликала усунення середньої яскравості обробленого зображення. Якщо необхідне придушення шуму пов'язане з використанням масивів більшого розміру, доцільно виконувати згортку непрямим чином, застосовуючи перетворення Фур'є, так як це зазвичай дає виграш в обсязі обчислень.

Медіанна фільтрація – метод нелінійної обробки сигналів, розроблений Тьюкі. Цей метод є корисним при придушенні шуму на зображенні. Одномірний медіанний фільтр є ковзним вікном, що охоплює непарне число елементів зображення. Центральний елемент замінюється медіаною всіх елементів зображення у вікні. Медіаною дискретної послідовності *a1, a2,..., aN* для непарного *N* є той елемент, для якого існують *(N-1)/2* елементів, менших або рівних йому за величиною, *(N-1)/2* великих або рівних йому за величиною. Нехай у вікно потрапили елементи зображення рівнями 80, 90, 200, 110, 120. У цьому випадку центральний елемент слід замінити значенням 110, яке є

медіаною впорядкованої послідовності 80, 90, 110, 120, 200. Якщо в цьому прикладі значення 200 є шумовим викидом монотонно зростаючої послідовності, то медіанна фільтрація забезпечить істотне поліпшення. Навпаки, якщо значення 200 відповідає корисному імпульсу сигналу (при використанні широкосмугових датчиків), обробка призведе до втрати чіткості відтворюваного зображення. Таким чином, медіанний фільтр в одних випадках забезпечує зменшення шуму, в інших – викликає небажане придушення сигналу [27].

Медіанний фільтр не впливає на ступінчасті або пилкоподібні функції, що зазвичай є бажаною властивістю. Однак цей фільтр пригнічує імпульсні сигнали, тривалість яких не перевищує половини ширини вікна. Фільтр також викликає сплощення вершини трикутної функції.

Можливості аналізу дії медіанного фільтра обмежені. Можна показати, що медіана добутку постійної *K* та послідовності *f(j)* дорівнює:

$$med\{K f(j)\} = K\, med\{f(j)\}. \tag{2.6}$$

Крім того,

$$med\{K + f(j)\} = K + med\{f(j)\}. \tag{2.7}$$

Але медіана суми двух будь яких послідовностей *f(j)* и *g(j)* не буде суммою їх медіан [28]:

$$med\{g(j) + f(j)\} = med\{g(j)\} + med\{f(j)\}. \tag{2.8}$$

Медіанний фільтр ефективніше пригнічує розрізнені імпульсні перешкоди, ніж гладкі шуми. Медіанну фільтрацію зображень для придушення шумів слід вважати евристичним методом. Її не можна застосовувати у сліпу. Навпаки, слід перевіряти результати, щоб переконатися в доцільності медіанної фільтрації.

2.2.2 Сегментація символів

Сегментація означає виділення областей однорідних за будь-яким критерієм, наприклад, за яскравістю. Розберемо метод сегментації на основі кластер-аналізу.

Метод сегментації на основі кластер-аналізу є природним узагальненням методу порогової обробки. Якщо у випадку класичної порогової обробки йдеться про перетворення полів числових значень, алгоритми кластер-аналізу дозволяють здійснювати схожу операцію над векторними полями. Другий аспект полягає у використанні оптимальних статистичних вирішальних правил при призначенні тій чи іншій точці зображення мітки α_1. Ця важлива обставина звільняє від необхідності підрахунку конкретних значень порогів T_1.

Завдання сегментації, що трактується як пошук кластерів у просторі характеристичних ознак, має таку формальну постановку. Нехай $\Phi(x, y)$ вихідне сегментоване (загалом векторне) випадкове поле, а K – число областей сегментації. При цьому кожній точці зображення (x, y) поставлений у відповідність вектор характеристичних ознак.

$$Z^T = (z_1, z_2, \dots, z_n). \tag{2.9}$$

Позначимо через $p(Z)$ густину розподілу ймовірностей випадкового поля. Передбачається, що $p(Z)$ являє собою кінцеву суміш, що розділиться, розподілів і визначається співвідношенням.

$$p(Z) = \sum_{j=1}^{K} P_j p_j(Z), \sum_{j=1}^{K} P_j = 1. \tag{2.10}$$

де P_j – числові коефіцієнти;

$p_j(Z)$ – щільність розподілу ймовірностей j-ї складової суміші.

Змістовно ці коефіцієнти можна інтерпретувати як апріорну ймовірність події (точка зображення належить j-й області) і як закон розподілу точок, що

належать j-й області зображення відповідно. Завдання зводиться до статистичної оцінки коефіцієнтів P_j та функцій $p_j(Z)$ по виборці.

$$Z = |Z_1, Z_2, \dots, Z_{N^2}, |. \quad (2.11)$$

Після отримання оцінок P_i та $p_j(Z)$ розмітка точок області може відбуватися за правилом: точка зображення (x, y), яка має вектор ознак Z_n, отримує мітку α_1, якщо:

$$P_i p_i(Z_0) = max_k P_k p_k(Z_0), k = 1, 2, \dots, K. \quad (2.12)$$

Після етапу сегментації ми маємо список об'єктів з деякими відомими значеннями атрибутів. Лише відсутні значення тотожності символів. Теоретично ми можемо застосувати будь-який метод розпізнавання символів, якщо він розроблений для відповідного типу даних (тобто онлайн або оффлайн).

2.2.3 Розпізнавання символів

Розпізнавання символів та аналіз структури двовимірних рисунків широко вивчалися протягом десятиліть. Багато методів розпізнавання символів працюють з припущенням, що символи вже ізольовані один від одного. Якщо це припущення виконується, розпізнавання символів може бути просто використовувати якийсь існуючий метод. Однак така ж ситуація не трапляється на етапі структурного аналізу. Двовимірні візерунки в різних доменах зазвичай мають дуже різні простори. Хоча деякі прийоми використовуються для розпізнавання інших двовимірних візерунків можуть, в принципі, також застосовуватись до математичного розпізнавання таких методів зазвичай вимагають значних модифікацій перед тим, як їх можна використовувати. Більше того, деякі ситуації дуже специфічні для математичного розпізнавання виразів і вимагають спеціально розроблених методи.

Є два основних типи основного алгоритму OCR, який може створити рейтинговий список символів-кандидатів.

Відповідність матриці передбачає порівняння зображення із збереженим гліфом на основі пікселів; він також відомий як «узгодження шаблонів», «розпізнавання образів», або «кореляція зображення». Це залежить від того, що вхідний гліф правильно ізольований від решти зображення, а збережений гліф – подібним шрифтом і в тому ж масштабі. Цей прийом найкраще працює із введеним на машинці текстом і погано працює, коли створюються нові шрифти.

Вилучення особливостей розкладає гліфи на «ознаки», такі як лінії, замкнуті петлі, напрямок лінії та перехрестя ліній. Особливості вилучення зменшують розмірність подання та роблять процес розпізнавання обчислювально ефективним. Ці функції порівнюються з абстрактним вектор-подібним поданням персонажа, який може зменшитися до одного або декількох прототипів гліфів. Загальні методики виявлення особливостей в комп'ютерному зорі застосовуються до цього типу OCR, що часто зустрічається у «інтелектуальному» розпізнавання почерку і справді найсучасніше програмне забезпечення OCR. Найближчі класифікатори сусідів, такі як алгоритм *k*-найближчих сусідів використовуються для порівняння функцій зображення із збереженими елементами гліфів та вибору найближчого відповідника [29].

Програмне забезпечення, таке як Клінопис і Tesseract OCR використовувати двопрохідний підхід до розпізнавання символів. Другий прохід відомий як «адаптивне розпізнавання» і використовує форми букв, розпізнані з високою впевненістю на першому проході, щоб краще розпізнати решту літер на другому проході. Це вигідно для незвичних шрифтів або неякісного сканування, де шрифт спотворений (наприклад, розмитий або зниклий).

Сучасне програмне забезпечення OCR, як наприклад OCRopus або використовує Tesseract нейронні мережі які були навчені розпізнавати цілі рядки тексту, замість того, щоб зосередити увагу на окремих символах.

Нова техніка, відома як ітераційне OCR, автоматично обрізає документ на розділи на основі макета сторінки. OCR виконується в розділах окремо з використанням змінних значень рівня достовірності символів, щоб максимізувати точність OCR на рівні сторінки.

2.2.4 Постобробки

Точність OCR може бути збільшена, якщо вихід обмежений а лексикон – перелік слів, які допускаються в документі. Це можуть бути, наприклад, усі слова англійською мовою, або більш технічний словник для певної галузі. Цей прийом може бути проблематичним, якщо документ містить слова, що не входять до лексикону, наприклад власні імена. Tesseract використовує свій словник для впливу на крок сегментації символів для підвищення точності.

Вихідним потоком може бути а простий текст потік або файл символів, але більш досконалі системи OCR можуть зберегти початковий макет сторінки та створити, наприклад, анотований PDF що включає як оригінальне зображення сторінки, так і текстове представлення для пошуку.

«Аналіз близьких сусідів» може використовувати співіснування частоти виправлення помилок, зазначаючи, що певні слова часто бачать разом. Наприклад, «Вашингтон, округ Колумбія» як правило, набагато частіше зустрічається в англійській мові, ніж «Вашингтонський DOC».

Знання граматики мови, що перевіряється, також може допомогти визначити, чи слово, ймовірно, є дієсловом чи іменником, наприклад, забезпечуючи більшу точність.

Алгоритм відстані Левенштейна також був використаний у подальшій обробці OCR для подальшої оптимізації результатів від OCR API.

3 КОМП'ЮТЕРНА МОДЕЛЬ ФІЛЬТРАЦІЇ ЗОБРАЖЕНЬ

3.1 Вибір середовища програмної реалізації

У данній кваліфікаційній роботі був розроблений алгоритм для розпізнавання тексту з зображень за допомогою оптичного зору та нейронної мережі Tesseract ORC, бібліотеки Python Tesseract та Python OpenCV. Для реалізації було обране середовище Pycharm IDE. Причиною такого вибору э зручніть використання для обранної мови програмування у данному середовищі.

Tesseract – механізм OCR з відкритим кодом, який завоював популярність серед розробників OCR. Незважаючи на те, що іноді впроваджувати та змінювати це може бути боляче, на ринку не було надто багато безкоштовних і потужних альтернатив OCR протягом найдовшого часу. Тессеракт почав як доктор філософії. дослідницький проект у HP Labs, Брістоль. Він набув популярності і був розроблений HP між 1984 і 1994 роками. У 2005 році HP випустила Tesseract як програмне забезпечення з відкритим вихідним кодом [30].

Tesseract сумісний з багатьма мовами програмування та фреймворками через обгортки. Його можна використовувати з наявним аналізом макета для розпізнавання тексту у великому документі, або його можна використовувати разом із зовнішнім детектором тексту для розпізнавання тексту із зображення одного текстового рядка.

Tesseract 4.00 включає нову підсистему нейронної мережі, налаштовану як розпізнавач текстових рядків. Воно бере свій початок у OCRopus Python-based LSTM реалізація, але була перероблена для Tesseract на C++. Система нейронної мережі в Tesseract була попередньою від TensorFlow, але сумісна з ним, оскільки існує мова опису мережі під назвою Variable Graph Specification Language (VGSL), яка також доступна для TensorFlow.

Щоб розпізнати зображення, що містить один символ, ми зазвичай використовуємо згорткову нейронну мережу (CNN). Текст довільної довжини – це послідовність символів, і такі проблеми вирішуються за допомогою RNN, а LSTM – популярна форма RNN.

LSTM чудово вивчають послідовності, але дуже сповільнюються, коли кількість станів занадто велика. Є емпіричні результати, які свідчать про те, що краще попросити LSTM вивчити довгу послідовність, ніж коротку послідовність багатьох класів. Tesseract розроблено на основі моделі OCRopus на Python, яка була форком LSMT в C++, який називається CLSTM. CLSTM – це реалізація моделі рекурентної нейронної мережі LSTM на C++, що використовує бібліотеку Eigen для чисельних обчислень.

Як вже зазначалось вище проект будується на основі ORC Tesseract. Для того щоб реалізувати його можливості треба встановити його на машину та відредагувати змінну середовища «path» і додати шлях до тессеракту.

OpenCV був започаткований Гарі Бредскім в Intel у 1999 році, а перший випуск вийшов у 2000 році. У 2005 році OpenCV був використаний на Stanley, автомобілі, який переміг у 2005 DARPA Grand Challenge. Пізніше його активний розвиток продовжився за підтримки Willow Garage під керівництвом Гарі Бредського та Вадима Писаревського. OpenCV тепер підтримує безліч алгоритмів, пов'язаних з комп'ютерним баченням і машинним навчанням, і розширюється з кожним днем.

OpenCV підтримує широкий спектр мов програмування, таких як C++, Python, Java тощо, і доступний на різних платформах, включаючи Windows, Linux, OS X, Android та iOS. Інтерфейси для високошвидкісних операцій GPU на основі CUDA і OpenCL також знаходяться в стадії активної розробки.

OpenCV-Python – це API Python для OpenCV, що поєднує в собі найкращі якості OpenCV C++ API та мови Python, бібліотека прив'язок Python, призначена для вирішення проблем комп'ютерного зору.

У порівнянні з такими мовами, як C / C ++, Python повільніше. Тим не менш, Python можна легко розширити за допомогою C / C ++, що

дозволяє нам писати інтенсивний обчислювальний код на C / C ++ і створювати обгортки Python, які можна використовувати як модулі Python. Це дає нам дві переваги: по-перше, код такий же швидкий, як і оригінальний код C/C++ (оскільки це фактичний код C++, що працює у фоновому режимі), а по-друге, його легше кодувати на Python, ніж на C/ C ++. OpenCV-Python – це обгортка Python для оригінальної реалізації OpenCV C ++.

OpenCV-Python використовує Numpy, яка є високо оптимізованою бібліотекою для числових операцій із синтаксисом у стилі MATLAB. Усі структури масивів OpenCV перетворюються в масиви Numpy та з них. Це також полегшує інтеграцію з іншими бібліотеками, які використовують Numpy, такими як SciPy і Matplotlib.

3.2 Програмна реалізація

Як вже зазначалось вище проект будується на основі ORC Tesseract. Для того щоб реалізувати його можливості треба встановити його на машину та відредагувати змінну середовища «path» і додати шлях до тессеракту.

Одразу після встановлення перевіряємо що ORC Tesseract підключився (рис. 3.1).

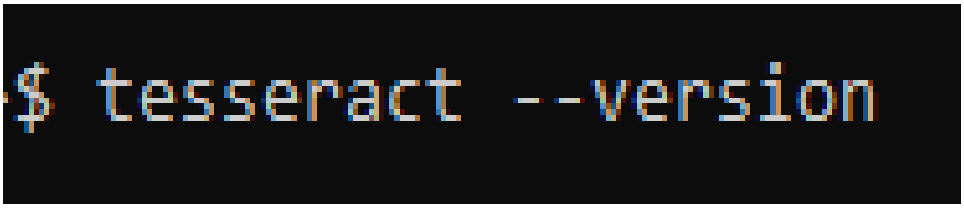

Рисунок 3.1 – Перевірка підключення ORC Tesseract

Для роботи з ORC Tesseract з мовою програмування Python була розроблена бібліотека Python-tesseract.

Python-tesseract – це інструмент оптичного розпізнавання символів (OCR) для Python. Тобто він розпізнає і «читатиме» вставлений у зображення текст.

Python-tesseract – це обгортка для Tesseract-OCR Engine від Google. Він також корисний як окремий сценарій виклику для тессеракту, оскільки він може читати всі типи зображень, які підтримуються бібліотеками зображень Pillow і Leptonica, включаючи jpeg, png, gif, bmp, tiff та інші. Крім того, якщо використовується як сценарій, Python-tesseract надрукує розпізнаний текст замість запису його у файл. підключаємо її з терміналу як на рисунку 3.2.

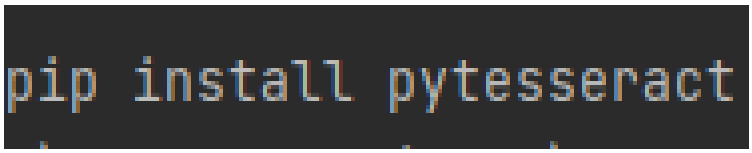

Рисунок 3.2 – Встановлення бібліотеки Python tesseract

Для початку нам потрібно встановити область у якій нейронна мережа буде розпізнавати текст. Щоб вирішити цю задачу було вирішено використати бібліотеку PyAutoGUI.

PyAutoGUI дозволяє скриптам Python керувати мишею та клавіатурою для автоматизації взаємодії з іншими програмами. PyAutoGUI працює в Windows, macOS і Linux і працює на Python 2 і 3.

PyAutoGUI має кілька функцій:

– контроль курсору у вікнах інших програм;

– натискання клавіш у різних додатків, сайтах (наприклад, для заповнення форм);

– робити знімки екрану, надавати йому якийсь маркер (наприклад, кнопку чи прапорець) і знаходити його на екрані;

– знайти вікно програми та перемістити, змінити розмір, розгорнути, згорнути або закрийти його (наразі лише для Windows);

– відображати вікна сповіщень і повідомлень.

Будемо використовувати функцію PyAutoGUI position для знаходження координат миші для виділення, знаходження координат вуглів прямокутної області у якій буде відбуватися розпізнавання тексту на екрані. Щоб визначити

цей прямокутник треба визначити дві точки для його формування. Через pyautogui.position знаходимо координати цих точок (рис. 3.3).

```
Point(x=121, y=141)
Point(x=606, y=417)
```

Рисунок 3.3 – Координати точок області розпізнавання тексту

Далі программа робить скріншот в області екрану (рис. 3.4), яку було визначено двума точками (рис. 3.5).

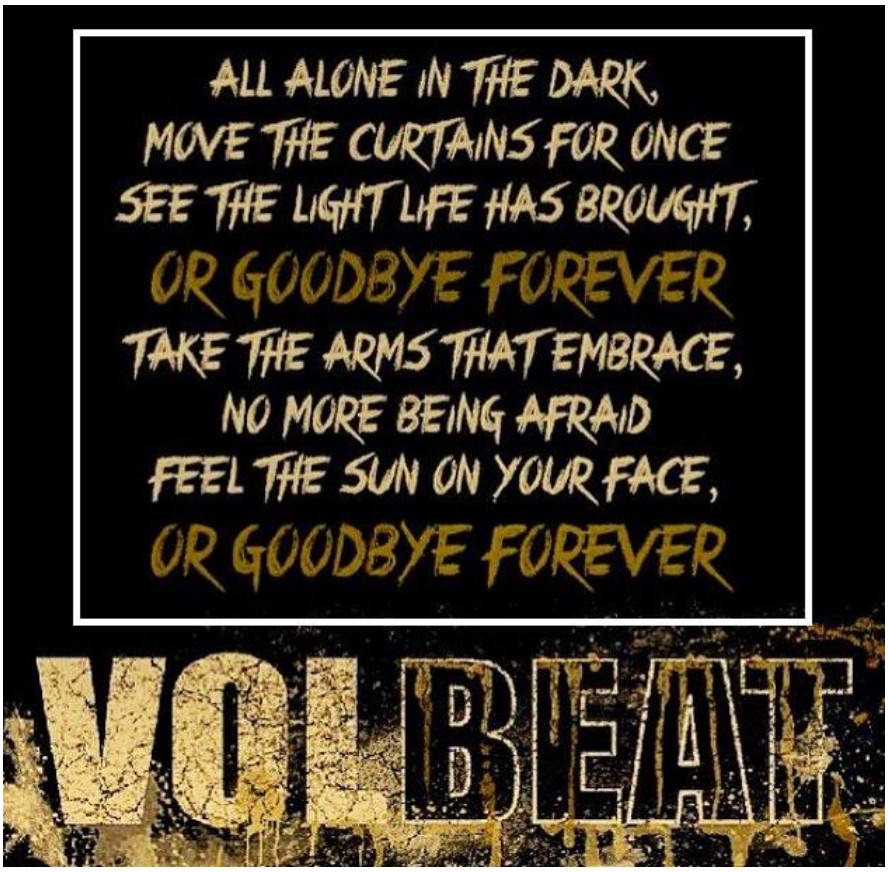

Рисунок 3.4 – Область розпізнавання

```
screen = np.array(ImageGrab.grab(bbox=(x, y, x1, y1)))
```

Рисунок 3.5 – Скріншот визначеної області

Після цього скріншот запам`ятовується у змінну та обробляється бібліотекою Python OpenCV.

Далі три етапи (рис. 3.6):

– знаходимо область з текстом;

– зчитуємо її до рисунку;

– розпізнаємо його й переводимо до формату String.

```
cv2.imshow('window', cv2.cvtColor(screen, cv2.COLOR_BGR2RGB))
cv2.imwrite(filename, screen)
img = cv2.imread('Image.png')
text = pytesseract.image_to_string(img)
```

Рисунок 3.6 – Зчитування та розпізнавання

Оскільки в основному мета розпізнавання тексту це оцифровування документів, то треба записати результат у текстовий документ (рис. 3.7).

```
newfile = open("newtextrec.txt","w+")
newfile.write(text)
newfile.read()
newfile = close()
```

Рисунок 3.7 – Запис у текстовий документ

Результат роботи программи (рис. 3.8).

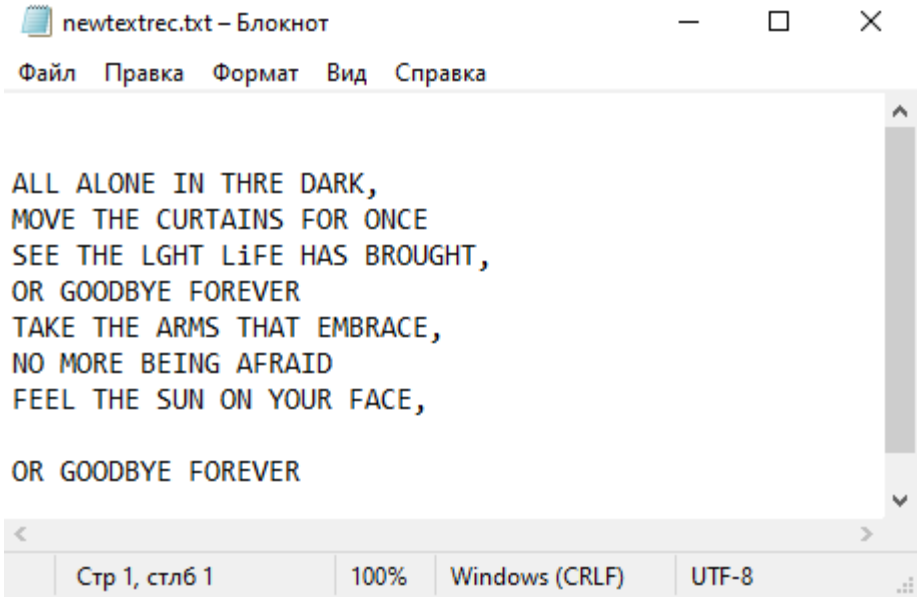

Рисунок 3.8 – Результат

3.3 Тестування розробленої моделі

Проведемо ще декілька тестів застосунку щоб переконатися у його роботі. Візьмемо ще декілька скріншотів з текстами оформлених різними шрифтами.

Перший тест (рис. 3.9).

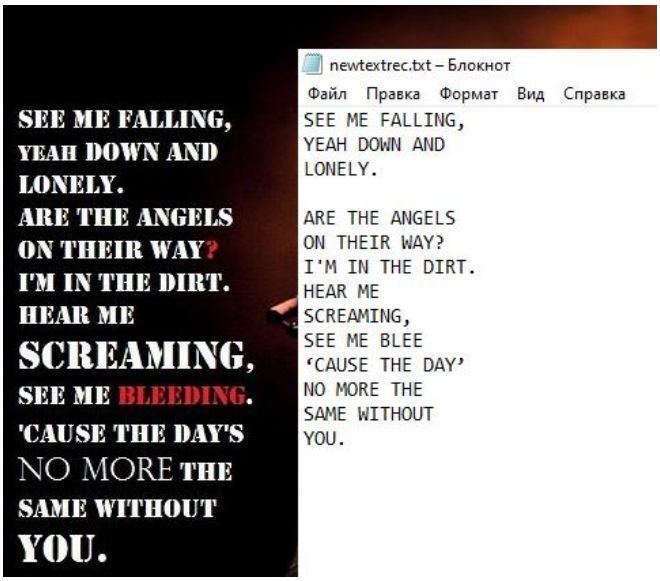

Рисунок 3.9 – Перший тест

Другий тест (рис. 3.10).

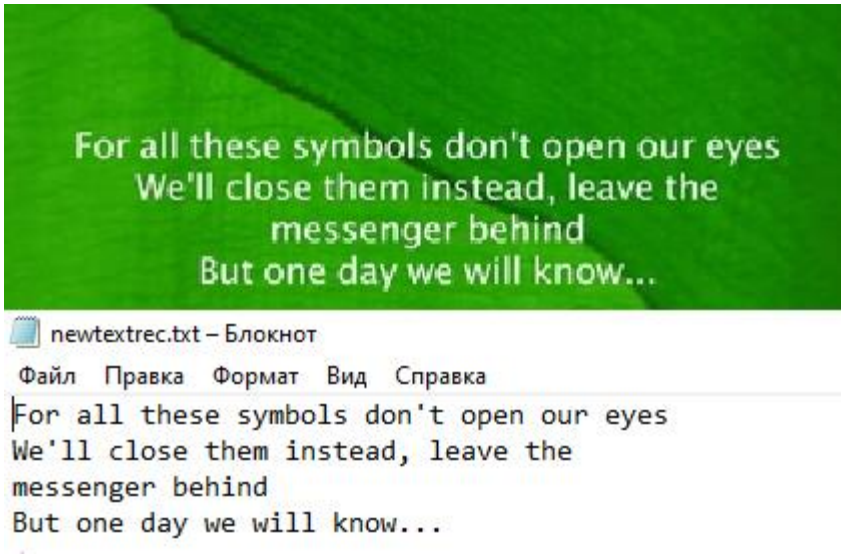

Рисунок 3.10 – Другий тест

Третій тест (рис. 3.11).

Рисунок 3.12 – Третій тест

ВИСНОВКИ

У рамках кваліфікаційної роботи був розроблений і реалізований метод розпізнавання тексту зі зображень за допомогою нейронної мережі Tesseract ORC та допоміжних бібліотек.

Також було основні методи розпізнавання тексту та технології розробки застосунків, які використоувуються у цій сфері. Наведено спосіб вирішення такої задачі за допомогою нейронної мережі за бібліотек Python, які значно полегшують работу з розпізнаванням тексту. Були здобуті навички з мови програмування Python.

Результатом даної роботи є розпізнаний текст текстового формату в блокноті, вивчено процесс розпізнавання тексту та його алгоритми.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

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

2. В.О. Козел. Методи та етапи автоматичного розпізнавання тексту (2008). Вісник Черкаського університету науковий журнал Випуск 172. Серія прикладна математика. Інформатика pp.75-86.

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

4. О. В Афонасенко., О.І. Елизаров (2008). Обзор методів распізнавання структурованих символів. Звіти державного університета систем управління та радіоелектроніки pp.83-88.

5. Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556.

6. He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 770-778).

7. Путятін, Є. П., Гороховатський, В. О., & Матат, О. О. (2006). Методи та алгоритми комп'ютерного зору: навч. посібник.

8. Gorokhovatskyi, V., Gorokhovatskyi, O., Yevgenyi, P., & Olena, P. (2018). Quantization of the Space of Structural Image Features as a Way to.

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

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

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

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

13. The flow of improved BRISK algorithm. URL: https://www.researchgate.net/figure/the-flow-of-improved-BRISK-algorithm_fig1_328946366 (дата звернення 24.04.2022).

14. Tvoroshenko, I., & Kukharchuk, V. (2021). Current state of development of applications for recognition of faces in the image and frames of video captures.

15. Cheriet M., Kharma N., Cheng-Lin Liu, Ching Y. Suen. (2007). Character Recognition Systems: A Guide for Students and Practitioners.

16. Жихаревич С. Е., Остапов І. В. Аналіз методів розпізнавання символів URL:http://nti.khai.edu:57772/csp/nauchportal/Arhiv/REKS/2016/REKS516/Zhikharevich.pdf (дата звернення 02.05.2022).

17. Eikvil L. OCR Optical Character Recognition. URL: https://www.nr.no/~eikvil/OCR.pdf. (дата звернення: 04.05.2022).

18. Lauritzen S. L. and Spiegelhalter D. J. (1988). Local computations with probabilities on graphical structures and their application to expert systems / Journal of Royal Statistics Society, series B – statistical methodology.

19. Casey R. and Lecolinet E. (1988). A survey of methods and strategies in character segmentation.

20. Розенблатт, Ф., Алтаев, В. Я., & Власюк, Б. А. (1965). Принципы нейродинамики: Перцептроны и теория механизмов мозга: Пер. с англ. Мир.

21. Вассерман Ф. (1992). Нейрокомп’ютерні технології: теорія і практика: пер. з англ. Світ.

22. Y-H Pao (1989). Адаптивне розпізнавання образів і нейронні.

23. Герберт Ф. (1982). OCR: історія, розпізнавання символів optisk. Шнц - Манчестер: Манчестерський центр, pp. 156.

24. Python PyAutoGUI documentation URL: https://pyautogui.readthedocs.io/en/latest (дата звернення 24.05.2022).

25. The Python Package Index (PyPI), repository of software for the Python programming language. URL: https://pypi.org/ (дата звернення 29.04.2022).

26. Mashtalir, S., Mikhnova, O. (2017). Detecting Significant Changes in Image Sequences. In: Hassanien, A., Mostafa Fouad, M., Manaf, A., Zamani, M., Ahmad, R., Kacprzyk, J. (eds) Multimedia Forensics and Security. Intelligent Systems Reference Library, vol 115. Springer, Cham. https://doi.org/10.1007/978-3-319-44270-9_8

27. Bodyanskiy, Y.V., Tyshchenko, O.K., Mashtalir, S.V. (2019). Fuzzy Clustering High-Dimensional Data Using Information Weighting. In: Rutkowski, L., Scherer, R., Korytkowski, M., Pedrycz, W., Tadeusiewicz, R., Zurada, J. (eds) Artificial Intelligence and Soft Computing. ICAISC 2019. Lecture Notes in Computer Science(), vol 11508. Springer, Cham. https://doi.org/10.1007/978-3-030-20912-4_36

28. Bodyanskiy, Y., Shafronenko, A., Mashtalir, S. (2020). Online Robust Fuzzy Clustering of Data with Omissions Using Similarity Measure of Special Type. In: Lytvynenko, V., Babichev, S., Wójcik, W., Vynokurova, O., Vyshemyrskaya, S., Radetskaya, S. (eds) Lecture Notes in Computational Intelligence and Decision Making. ISDMCI 2019. Advances in Intelligent Systems and Computing, vol 1020. Springer, Cham. https://doi.org/10.1007/978-3-030-26474-1_44

29. Офіційний сайт Stack Overflow. URL: http://stackoverflow.com/research/developer-survey-2015#tech-super. – (дата звернення 26.04.2022).

30. Andersen, S. K., Jensen, F. V. and Olesen, K. G. (1987) The HUGIN core—preliminary considerations on systems for fast manipulations of probabilities. In Proc. Workshop on Inductive Reasoning: Managing Empirical Information in AI-Systems, Risø.