МАГІСТЕРСЬКА РОБОТА

МР. ІПм - 60.00.00.000 ПЗ

Група ІПм-23-2

Мкртичян Георгій

2024

Івано-Франківський національний технічний університет нафти і газу

Інститут інформаційних технологій

Кафедра інженерії програмного забезпечення

Мкртичян Георгій Георгійович
(прізвище, ім'я, по батькові)

УДК 004.942
(індекс)

МАГІСТЕРСЬКА РОБОТА

Дослідження методів, моделей та алгоритмів розпізнавання зображень

(назва роботи)

Інженерія програмного забезпечення

(назва освітньої програми)

121 - Інженерія програмного забезпечення

(шифр і назва спеціальності)

Мкртичян Г.Г.
(підпис, ініціали та прізвище здобувача освітнього ступеня)

Науковий керівник Лютак Ігор Зіновійович, д.т.н., професор
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Допущено до захисту
Завідувач кафедри
доц. Бандура В.В.
(посада) (підпис) (дата) (ініціали та прізвище)

Нормоконтроль
доц. Вовк Р.Б.
(посада) (підпис) (дата) (ініціали та прізвище)

Робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело

Івано-Франківськ – 2024

Івано-Франківський національний технічний університет нафти і газу

Інститут інформаційних технологій

Кафедра інженерії програмного забезпечення

Освітній рівень магістр

Спеціальність 121 – Інженерія програмного забезпечення

ЗАТВЕРДЖУЮ:
Зав. кафедрою **ІПЗ**

доц. В.В. Бандура

“ 04 ” вересня 2024 р.

ЗАВДАННЯ

НА МАГІСТЕРСЬКУ РОБОТУ СТУДЕНТУ

Мкртичяну Георгію Георгійовичу

(прізвище, ім’я, по-батькові)

1. Тема магістерської роботи “Дослідження методів моделей та алгоритмів розпізнавання зображень ”

керівник проекту (роботи) Лютак Ігор Зінойович, д.т.н., професор

затверджені наказом закладу вищої освіти від “ 22 ” листопада 2024 р. № 781/7

2. Строк подання студентом проекту (роботи) 15 грудня 2024 р.

3. Вихідні дані до проекту (роботи) Архітектура, формальний опис алгоритмів, моделей та технологій функціонування систем розпізнавання зображень.

4. Зміст розрахунково - пояснювальної записки(перелік питань, які потрібно розробити)

1.Теоретичні відомості про технології розпізнавання зображень. Історія розвитку технологій.

2. Дослідження сучасних технологій розпізнавання зображень. Моделі та алгоритми.

3. Удосконалення алгоритму для розпізнавання зображень текстів та його конвертації в текст

5. Перелік графічного матеріалу (з точним зазначенням обов’язкових креслень)

1. Принцип розпізнавання тексту (технологія OCR)(рис. 1.4., ст. 1)

2. Схема взаємодії ШІ з камерами(рис.1.5, ст.19)

3. Схема роботи алгоритму Backpropagation(рис.2.6., ст. 42)

4. Візуалізація алгоритму роботи додатку RecQuick (рис.3.2, ст. 66)

5. Зовнішній вигляд додатку «RecQuick»(рис. 3.8., ст. 78)

6. Консультанти розділів проекту (роботи)

Розділ	Консультант	Підпис, дата
Перевірка на плагіат	доц. к.т.н. Вовк Р. Б.	

7. Дата видачі завдання 04 вересня 2024 р.

Керівник _____
(підпис)

Завдання прийняв до виконання_____
(підпис)

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Назви етапів магістерської роботи	Строк виконання етапів роботи	Примітка
1	Підбір та вивчення літератури	20.09.2024	виконано
2	Аналіз існуючих алгоритмів та моделей розпізнавання зображень	01.10.2024	виконано
3	Проектування власного методу на основі існуючого алгоритму	12.10.2024	виконано
4	Реалізація власної системи розпізнавання текстів на зображеннях	25.10.20234	виконано
5	Виконання попереднього тестування та налагодження програмного забезпечення	05.11.2024	виконано
6	Оформлення пояснювальної записки	22.11.2024	виконано
7	Затвердження пояснювальної записки роботи завідувачем кафедри	15.12.2024	виконано

Студент – магістр _____
(підпис)

Керівник роботи _____
(підпис)

АНОТАЦІЯ

Магістерська робота: 92с., 23 рис., 40 джерел, 1 додаток.

Тема: Дослідження методів моделей та алгоритмів розпізнавання зображень.

Об'єктом дослідження: моделі та алгоритми та існуючі рішення розпізнавання об'єктів на зображень.

Мета роботи: дослідження моделей розпізнавання зображень та виявлення переваг і недоліків кожної з них. Удосконалення одного з алгоритмів розпізнавання тексту на зображення та його реалізація у мобільному додатку.

Предмет дослідження: інформаційні технології розпізнавання зображень для виявлення та розпізнавання об'єктів на них.

Результати дослідження: Виконано аналіз та порівняння існуючих рішень розпізнавання зображень і на їх основі було удосконалено один з алгоритмів для розпізнавання тексту.

Висновок: В результаті дослідження було оптимізовано власний додаток для розпізнавання текстів на зображеннях, який базується на одному з вже існуючих алгоритмів.

РОЗПІЗНАВАННЯ ЗОБРАЖЕНЬ, МАШИННЕ НАВЧАННЯ, НЕЙРОННІ МЕРЕЖІ, МОБІЛЬНІ ДОДАТКИ, ІНОВАНЦІЯ, РОЗПІЗНАВАННЯ ОБЄКТІВ.

ABSTRACT

Master's Thesis: 92 pages, 23 figures, 40 sources, 1 appendices.

Topic: Research on Methods, Models, and Algorithms for Image Recognition.

Research Object: Models and algorithms for object recognition in images.

Objective: To study of image recognition models and identification of advantages and disadvantages of each of them. Improvement of one of the text-to-image recognition algorithms and its implementation in a mobile application.

Subject of Research: Information technologies for image recognition to detect and recognize objects in images.

Research Results: An analysis and comparison of existing image recognition solutions were performed, and based on this, one of the text recognition algorithms was improved.

Conclusion: As a result of the study, an application for text recognition in images was optimized, based on one of the existing algorithms.

IMAGE RECOGNITION, MACHINE LEARNING, NEURAL NETWORKS, MOBILE APPLICATIONS, INNOVATION, OBJECT RECOGNITION.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

ШІ – Штучний інтелект

РЗ – Розпізнавання зображень

OCR – Оптичне розпізнавання тексту

ЕОМ – Електронно-обчислювальна машина

ПЗ – Програмне забезпечення

КЗ – Комп’ютерний зір

API – прикладний програмний інтерфейс

GPU – Графічний процесор

CPU – Центральний процесор

ВСТУП

Актуальність роботи

Розпізнавання зображень є важливою і дуже актуальною темою у наш час, аргументів у захист даного твердження є дуже багато. Оскільки в сучасному житті можливість виявляти певні проблеми у різних галузях, таких як, медицина, транспорту, розваг, логістики, військової справи і багато ще де. Дана технологія дозволяє виявляти їх “природу” як можна швидше, що економить час, а іноді і життя.

Також технології розпізнавання зображень дозволяють людям економити час та кошти, наприклад оцифрування книжок і розпізнавання номерів машин з допомогою технології оптичне розпізнавання тексту (OCR), чи виявлення певних об’єктів для їхнього відстеження на певній території для аналізу, наприклад як можливість відстеження кількості зафіксованої популяції тварин та території природного парку і тому подібне.

Загалом дана технологія є дуже широко поширеною вже не перший рік і вона пройшла технічну еволюцію починаючи з 60-х років, а саме алгоритми на основі математичних правил і геометрії, такі як виявлення контурів та порогова обробка об’єктів і до сьогоднішніх днів де вже використовуються більш обширні технології такі як машинне навчання чи нейронні мережі.

По при усю перевагу сучасних технологій все одно люди ще не дійшли до піку у даній технології, оскільки з кожним роком ростуть потужності обчислювальних машин і систем вчасності GPU та CPU, що призводить до створення нових алгоритмів та моделей, а також удосконаленням старих з використанням нових технічних засобів. Наприклад багатошаровий перцептрон (MLP) не був дуже ефективним на момент свого створення, а саме в кінці 50-х років, оскільки не хватало певних алгоритмів (Adam, RMSprop, та Adagrad) і обчислювальних технологій. В результаті чого він не міг вирішувати задачі, які не є лінійно роздільними (наприклад, проблему XOR).

Таким чином я ще раз акцентую на тому , що тема є важливою, актуальною бо з кожним роком технології які почали розвиватися ще у 80-90-х роках стають все більш розвинуті і здатні виконувати нові задачі.

Ще не дуже приємною, але актуальною причиною чото варто розглянути технології РЗ є ситуація у нашій державі. Так ситуація не легка і вже багато років йде війна, але завдяки даним технологіям було спасенне життя. Може хтось заперечити на рахунок цього, але я вважаю, що можливість моментально отримувати і аналізувати дані отримані з супутників, статичних камер чи дронів може допомогти, як не виграти війну, так хоч спасти життя людям, аби вони могли сховатися від обстрілів чи провести ротацію при масовому натиску сил противника.

Також в кінці треба акцентувати на тому, що дана технологія може допомогти у розвитку інших сферах не зв'язаних на пряму з обробкою зображень. Це не моя власна думка, а конкретно слова Йошуа Бенджіо можливо ця людина мало кому знайома, але він є лауреат премії Тюрінга та один з провідних фахівців у сфері нейронних мереж. Він каже, що розвиток технологій РЗ дозволить у майбутньому створювати універсальні моделі, які зможуть аналізувати не лиш зображення, але й такі не очевидні речі як звукові чи світлові хвилі для подальшого застосування у певних галузях [13,14].

Мета і задачі дослідження

Метою магістерської роботи є дослідження моделей розпізнавання зображень та виявлення переваг і недоліків кожної з них. Удосконалення одного з алгоритмів розпізнавання тексту на зображення та його реалізація у мобільному додатку.

Досягнення мети включало розв'язання таких задач:

- огляд існуючих алгоритмів, моделей, методів РЗ;
- порівняння існуючих алгоритмів, моделей, методів та рішень РЗ;
- вибір релевантного алгоритму РЗ для удосконалення та обґрунтування доцільності його використання;

- програмна реалізація функції РЗ у додатку та його аналіз для подальшого розвитку.

Об’єктом дослідження є моделі та алгоритми та існуючі рішення розпізнавання об’єктів на зображень.

Предметом дослідження є інформаційні технології розпізнавання зображень для виявлення та розпізнавання об’єктів на них.

Методи дослідження Для виокремлення вимог до програмного рішення та для загального аналізу технології РЗ було застосовано такі методи дослідження, а саме порівняння існуючих рішень в предметній області та проведення експерименту використання реалізованого програмного рішення.

Наукова новизна одержаних результатів

Здійснено глибокий аналіз систем розпізнавання об’єктів зображень у вчасності для розпізнавання тексту, в результаті чого було удосконалено архітектуру для одної з систем, також було виявлено недоліки та переваги існуючих рішень.

Практичне значення одержаних результатів.

На основі проведеного дослідження та аналізу було розроблено функціонуючий додаток для розпізнавання текстів на зображеннях з використанням ML Kit, яке можна використовувати в якості додатку відповідно до функціоналу та поліпшувати його у майбутньому.

Особистий внесок

1. Проведено порівняння існуючих рішень РЗ.
2. Виявлено недоліки існуючих рішень РЗ.

3. На основі отриманих результатів, було удосконалено архітектуру одного з алгоритмів для розпізнавання текстів на зображеннях, в результаті чого було отримано додаток з великою швидкістю та надійністю.

Структура магістерської роботи.

Магістерська робота викладена на 92 сторінках друкованого тексту, який складається з вступу, трьох розділів, висновків, списку використаних джерел (40 найменувань). Робота містить 23 рисунків.

РОЗДІЛ 1 ТЕОРЕТИЧНІ ВІДОМОСТІ ПРО ТЕХНОЛОГІЇ РОЗПІЗНАВАННЯ ЗОБРАЖЕНЬ. ІСТОРІЯ РОЗВИТКУ ТЕХНОЛОГІЙ

1.1 Огляд технологій розпізнавання зображень

1.1.1. Вступ до технологій розпізнавання зображень

Загалом РЗ – це певний процес для аналізу даних для рішення конкретних задач чи цілей. В результаті аналізу даних комп’ютер вчиться відрізняти певні об’єкти на зображеннях класифікує їх і надає їх певні властивості, що в свою чергу допомагає людям вирішувати певні задачі у різних сферах життя, але конкретніше у іншому пункті даної роботи.

Перечитавши велику кількість літератури, а саме «Сегментація зображень: Огляд»[3], «Цифрова обробка зображень»[4], «Посібник з обробки зображень»[10], «Глибоке навчання»[13], «Штучний інтелект: сучасний підхід»[19] , «Алгоритми та застосування комп'ютерного зору»[26] та багато іншої можу сказати, що систем аналізу зображення мають три основні напрямки діяльності, а саме виявлення, класифікація і сегментація об'єктів на зображеннях, що в свій час допомагає економити велику кількість часу. Поділ на три різні групи є на мою думку дуже зручним бо кожен з пунктів фактично створений для певної задачі хоча вони час від часу можуть і суміщатися.

Особливості і цілі систем аналізу зображення:

Виявлення – це фактичний поділ зображення на певні об’єкти та визначення їхнього положення на зображенні

Класифікація дуже банальна річ бо всі люди хто працюють у сфері ІТ знають, що класифікація це розподіл певних об’єктів відповідно по властивостям по різним групам. Такий метод часто використовується у багатьох сферах нашого життя самий банальний приклад – це камери на світлофорах, які можуть автоматично підстроювати час переключення світла на зелене чи червоне залежно від кількості автомобілів чи людей біля дислокації світлофора

Сегментація самий найменш зрозумілий для мене система обробки, але від цього вона не є менш цікавою. Фактично вона ділить зображення на окремі області, або по іншому кажучи сегменти, що в свій час полегшує аналіз зображень для їх подальшого використання.

Хочеться зробити висновок по даному підпункту на рахунок технологій РЗ, а саме в чому їх ціль я думаю що все ж вона у тому, щоб облегшувати людям рутині процеси і допомагати приймати рішення в критичних ситуаціях, коли йде час навіть не на дні, а на години чи хвилини.

1.1.2. Історія розвитку технологій РЗ

Загалом дана технологія має дуже не малу історію і фактично її я поділив на 3 основні періоди, а саме:

Зародження технології(1960-1970рр)

В даний період не було достатньої кількості технологій і обпилювальних систем для аналізу текстів і тим більше складних об’єктів. Загалом у цей період вчені-програмісти намагалися навчити систему елементарним навичкам, а саме відрізняти прості геометричні фігури такі, як кола, квадрати чи прямі лінії.

Правильно сказати що цей період є першим «товчком» в сторону розвитку технологій РЗ і тогочасні програмісти фактично дали початок усім теперішнім графічним системам РЗ.

Етап машинного навчання (1980-2000рр.)

В цей період появляються вже більш менш продуктивні електронно обчислювальні машини(ЕОМ), а потім і портативних комп’ютерів, що призвело до розвитку більш сучасних технологій навчання за рахунок нових алгоритмів, а також росту продуктивності ЕОМ порівняно з попередніми. Результатом усього цього стала поява перши НМ і статистичних методів навчання, що відкрило нові можливості тогочасним програмістам для автоматизації розпізнавання зображень, що дало ще більший скачок для технологій зв’язаних з РЗ та конкретніше з ШІ.

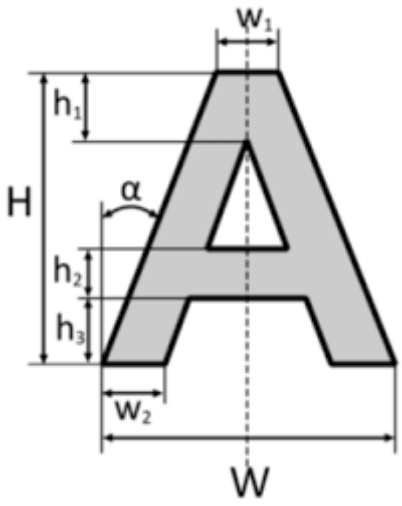

Рис. 1.1. Ілюстрація основ теорії розпізнання образів

Цей період є теж дуже важливим оскільки з'явилися нові алгоритми, комп'ютери стають в рази потужнішими, а також стає більше людей зв'язаних з сферою IT що призводить до швидшого удосконалення технологій, хоча вони не була такими точними як сучасні, але при цьому були революційними на той час.

Глибоке навчання (2000-донині)

Дана технологія зробила велику революцію у розпізнаванні зображень, у тому числі так звана технологія FaceID, яка отримала непоганий ривок у розвитку за рахунок розвитку гаджетів, а точніше смартфонів.

Рис. 1.2. Революція техгології FaceID

Також в цей час активно розвивається компанія GeForce із своєю технологією CUDO, яка в рази збільшила швидкість навчання різних моделей за рахунок використання продуктивних GPU, які і по сьогоднішній час активно використовуються для навчання різних нейронних мереж. В результаті на сьогоднішній день ми маємо велику кількість нових можливостей для розвитку глибокого навчання і ШІ для РЗ у різних сферах діяльності.

1.1.3. Типи розпізнавання зображень.

На сьогоднішній день існує чотири основні задачі, які вирішують за допомогою технологій РЗ:

Розпізнавання об’єктів.

Дана технологія існує для класифікації на зображеннях об’єктів, тварин, людей і так далі для локалізації у межах зображення для подальших маніпуляцій з ними, наприклад підрахунок кількості людей які проходять протягом дня через касовий апарат. Також принципи розпізнавання об’єктів широко використовуються у сучасних гаджетах для облегшення розпізнавання речей на зображеннях, наприклад автоматичне виявлення що за модель телефона на фото і тому подібне.

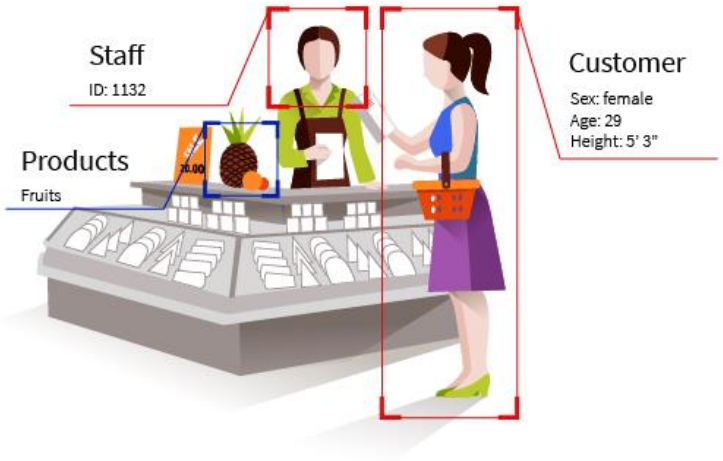

Рис. 1.3. Візуалізація роботи системи розпізнавання об’єктів

Розпізнавання тексту (OCR).

Дана технологія по іншому ще називається – оптична розпізнавання символів, що означає, що вона вміє виділяти текст з фотографій та перетворює його в редагований формат.

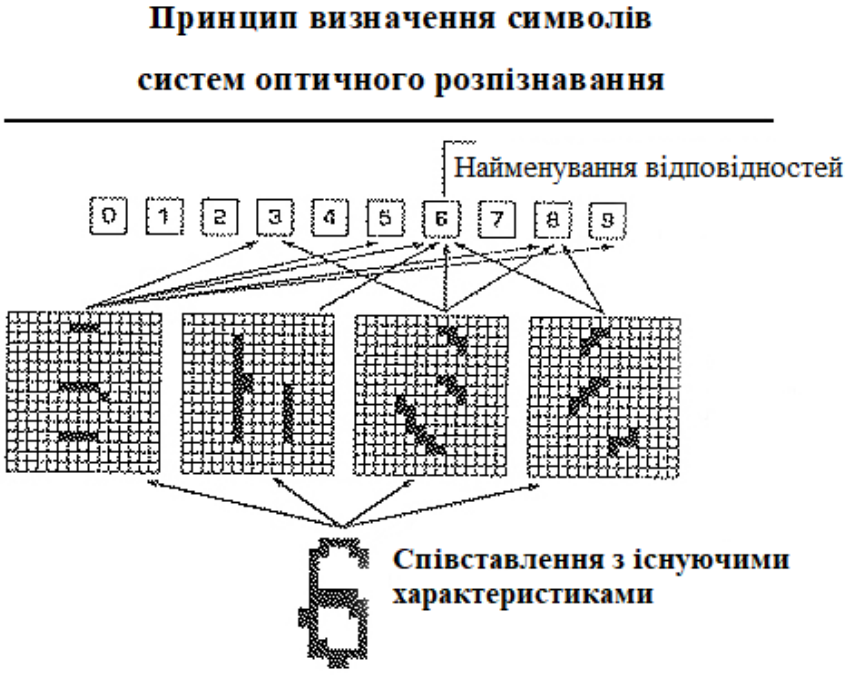

Рис. 1.4. Принцип розпізнавання тексту (технологія OCR)

Також дана технологія широко використовується для зчитування номерів автомобілів для автоматичної видачі штрафів і базується вона на тому що вона шукає найбільш схожі символи серед отриманих результатів.

Відеоаналіз.

Дана технологія схожа на розпізнавання об’єктів лиш з тою відмінністю що вона здатна реагувати на певні події так звані аномалії, які призводять до оповіщень чи відповідних реакцій у бік користувача. Також широко використовується в тих же сферах що і розпізнавання об’єктів, але при інших сценаріях, коли потрібна не лиш статична інформація, а пошук і фіксація певних відхилень.

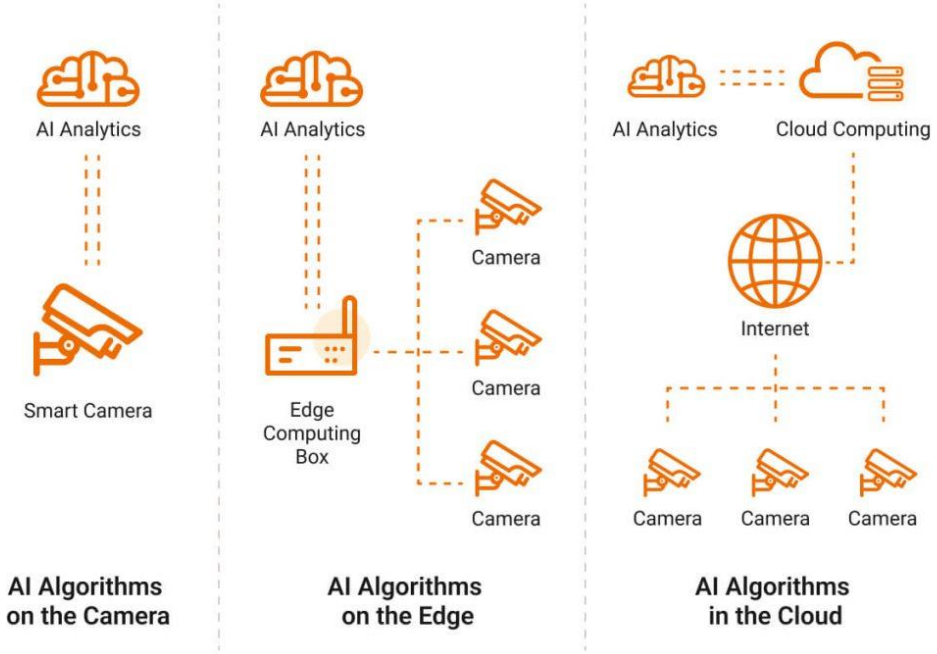

Рис. 1.5. Схема взаємодії ШІ з камерами

Медичне зображення.

Також існує окремий вид технології де алгоритми зв'язаних на пряму і виключно для медичної діяльності. Основна їхня ціль це аналіз зображень рентгенівські знімки, КТ, МРТ для виявлення аномалій, раку, переломів, пухлин, тощо. Так покрайні мірі написано у всіх джерелах які я читав але попри це я не знайшов ні одного фото МРТ чи ще чогось в інтернеті де би була використана технологія РЗ для виявлення хвороби чи поставки діагнозу.

1.2 Існуючі методи обробки та аналізу зображень

Ще на початку двадцять першого століття основною причиною неточностей у технологіях машинного навчання вчасності у РЗ була нестача обчислювальних ресурсів. Не спорю на сьогоднішній час обчислювальних ресурсів теж не є безмежні, але тепер скорі рішає правильний аналіз даних та використання, або створення самого підходящого(актуального) алгоритму РЗ для подальшого

використання у робочих потребах, або для удосконалення певних аспектів життєдіяльності.

1.2.1. Використання комп'ютерного зору для обробки зображень

Людський зір являє собою важливим аспектом діяльності людини. Він допомагає координувати рух, сприймати зовнішню інформацію, аналізувати відстані між об'єктами, сприймати певні об'єкти і умовно їх помічає в голові і багато ще чого. Це все сказано, аби простіше було сприйняти інформацію про таке поняття, як комп'ютерний зір.

Комп'ютерний зір або ж Комп'ютерне бачення — являє собою теорію та технологію для створення машин, які можуть проводити виявляння, відстежування та визначення об'єктів. Простіше кажучи – це аналог нашого ока прикладів є дуже багато, але фактично їх можна назвати по іншому ще камери стеження, оскільки, як це не дивно вони виконують схожу з ними і нашими очима функцію тільки на відмінно від камер КЗ має певний алгоритм дії залежно від поставленої мети та задач які переслідує користувач чи розробник. Дана технологія має дуже багато аспектів застосування але в нашому випадку він використовується для розпізнавання літер, об'єктів, символів і багато ще чого на зображеннях для виконання тих чи інших технічних завдань.

В основному для обробки зображень використовують наступні три етапи КЗ, а саме:

Аналіз контурів і об'єктів.

Являє собою дуже важливу ланку КЗ, а саме він допомагає виділяти краї об'єктів та здатний сприймати особливості об'єктів на зображеннях. Дана методика є дуже популярною у задачах зв'язаних з сегментацією, розпізнанням форм та класифікацією об'єктів. Сам аналіз контурів та об'єктів виконується з допомогою наступних методів.

Оператор Собеля. Він являє собою метод виділення контурів(країв) зображення, який використовує градієнт інтенсивності пікселів для визначення напрямку змін освітленості.

Даний оператор застосовую два ядра, або як їх ще називають по іншому фільтри розміром 3x3. Перший фільтр використовується для горизонтального напрямку, а інший для вертикального. В результаті даних маніпуляцій ми отримуємо наступний результат, а саме зображення на якому видні яскраві області різкими змінами інтенсивності.

Рис. 1.6. Приклад використання оператора Собеля

Основною сферою застосування даного оператора являє собою сегментація або розпізнавання об’єктів за рахунок хорошої контрастності зображення, що в свою чергу облегшує задачу для алгоритму для пошуку обєктів.

Оператор Лапласа. Він являє собою метод який виявляє краї зображення на основі функцій похідних другого порядку, що визначає точні зміни мінімумів і максимумів інтенсивності зображення.

Основною його метою є виділення на зображенні контурів та меж зображення, проте в нього є і свої недостати, а саме потреба використовувати згладжування зображень, оскільки даний оператор є дуже чутливий до шумів що призводить до нечітких зображень. Якщо казати простими словами без формул і всього подібного то даний оператор працює за рахунок зміни інтенсивності навколо кожного пікселя, що в процесі обробки зображення дозволяє знаходити області де зображення робить перехід від світлових відтінків до темних і навпаки.

Детектор Кенні. Він являє собою одним із самих точних алгоритмів для виявлення країв, який забезпечує високу точність для зображення, при цьому

всьому він доволі стійкий до шумів, але при цьому він має і певні мінуси а саме те, що плавно переходячи об’єкти він іноді затирає одним кольором через погану контрастність. Загалом схема роботи даного алгоритму доволі складна і включає в себе декілька інших методів які виконуються покроково, а саме :

•Перший етап це згладжування з допомогою фільтру Гаусса для зменшення шумів у зображення.

•Другим етапом є розрахунок градієнта, а точніше виявлення країв за рахунок обчислення значень інтенсивності зображення.

•Третій етап це зменшення усіх країв зображення до одного пікселя після чого йде розподіл(класифікація) країв на основні і можливі тобто не обов’язкові, або інакше кажучи слабкі(більшість світлих тонів).

•І п’ятий, але фактично шостий етап це – об’єднання можливих країв і основних для отримання зображення.

В результаті ми отримуємо доволі чітке зображення але теж не без нюансів, а саме є ймовірність зникнення певних об’єктів через їхню нечіткість що я замічав не один раз.

Основним середовищем застосування є відеоспостереження бо даний алгоритм дає доволі чіткі показники у даній сфері діяльності. Також даний алгоритм широко застосовується в сферах де потрібні чіткі дані на зображеннях а саме робототехніка і медицина, де дані властивості можуть прискорити певні процеси.

Для ліпшого розуміння даного розділу було створено з використанням додатку Paint візуальне зображення для порівняння методів розпізнавання контурів де видні всі їх плюси та мінуси. Для реалізації була використана мова Python та відповідні бібліотеки. Код роботи програми представлений у лістингу 1.1.

Лістинг 1.1.

```
import cv2
import numpy as np
import matplotlib.pyplot as plt
```

```
# Завантаження зображення
image_path = '/mnt/data/image.png'
image = cv2.imread(image_path)

# Перехід зображення в градації сірого
gray_image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)

# Застосування оператора Собеля для виявлення горизонтальних і вертикальних країв
sobel_x = cv2.Sobel(gray_image, cv2.CV_64F, 1, 0, ksize=3)
sobel_y = cv2.Sobel(gray_image, cv2.CV_64F, 0, 1, ksize=3)
sobel_edges = cv2.magnitude(sobel_x, sobel_y)
# Застосування оператора Лапласа
laplacian_edges = cv2.Laplacian(gray_image, cv2.CV_64F)……
# Застосування детектора Кенні
canny_edges = cv2.Canny(gray_image, 100, 200)
# Показ результатів
plt.figure(figsize=(12, 8))
# Оригінальне зображення
plt.subplot(2, 4, 1)
plt.imshow(cv2.cvtColor(image, cv2.COLOR_BGR2RGB))
plt.title("Original Image")
plt.axis("off")…
```

Отримане зображення з порівнянням різних методів розпізнавання контурів:

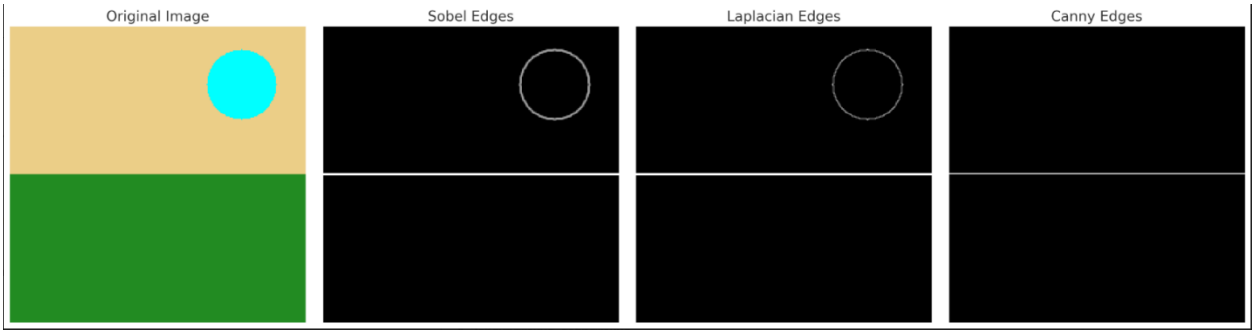

Рис. 1.7. Зміна зображення залежно від використаної технології

В результаті отримую наступний результат, а саме зображення оригіналу і всіх інших методів. З нього видно що самий наворочений варіант в даному випадку виявився не самим ефективним бо він ідеально підходить для реальних задач таких, як відеоспостереження, а не для векторних картинок і через це пропало коло. Як не дивно на мою суб’єктивну думку самим ефективним виявися варіант із використанням оператора Собеля, бо він показав саму чітку картинку, що облегшить подальший аналіз зображення без втрати деталей.

1.2.2. Детекція та відстеження об'єктів

Детекція та відстеження об'єктів являються одними з основних завданнями комп’ютерного зору. Завдяки ним відбувається пошук та відстеження об’єктів на зображеннях та відео. Завдяки даній технології виявляється не лиш що за об’єкт на кадр, але і його коректність відповідно до своїх координат.

Для детекції об’єктів використовують відповідні алгоритми:

- YOLO (You Only Look Once)
- Faster R-CNN (Region-Based Convolutional Neural Networks)
- SSD (Single Shot MultiBox Detector)
- RetinaNet
- Mask R-CN

А також існують відповідні алгоритми для їх відстеження:

- SORT (Simple Online and Realtime Tracking)
- DeepSORT
- Correlation Filters (KCF, MOSSE)
- Optical Flow
- Siamese Networks

Загалом на сьогоднішній день є чотири основні задачі для детекції і відстеження обєктів:

1. Одинична детекція.

Використовується для знаходження лиш одного об’єкту наприклад знаходження на зображені певну машину і стеження виключно за нею ігноруючи інші об’єкти.

2. Детекція декількох об’єктів.

Найпопулярніша задача для детекції оскільки часто потрібно розпізнавати не один об’єкт а декілька і вона в цьому допомагає їх знаходити і класифікувати відповідно від потреб.

3. Детекція у реальному часі.

Аналогічно попереднім двом вміє класифікувати об’єкти, але при цьому може бути використана для виявлення і маніпуляцій з ними у реальному часі, що потрібно для деяких задач.

4. Детекція у складних умовах

Дані задачі заточені на розпізнаванні зображень у дуже поганих умовах, таких, як сильні шуми чи низьке освітленням.

1.2.3. Виділення ознак

Даний процес дозволяє виявити, знаходити унікальні та значні властивості зображення для подальшого аналізу і використання. Загальне визначення у слова ознака – це елемент зображення, яке відображає його певні характеристики, такі як текстури, патерни, контури, кути і багато чого іншого.

Ознаки забезпечують:

- Інформативність.

Усі виділені ознаки є досить унікальні, щоб відрізняти один об’єкт від іншого.

- Стійкість.

На ознаки не повинні впливати зміна обставин і вони мають лишатися стабільними. Тобто масштабі, освітленні, повороті або шумі не повинні впливати на фінальний результат.

- Ефективність.

Ознаки мають бути не великі за об’ємом, але при цьому повинні бути достатньо інформативними для виявлення об’єктів.

Основні етапи процесу виділення ознак:

1.Попередня обробка

Для більш чіткого виділення ознак за зображеннях часто обробляють перед виділенням ознак, а саме використання фільтрації шумів, використання карт нормалей, а також згладжування для покрашення контрастності об’єктів на зображеннях.

2. Визначення ключових точок.

Ключові точки – це певні місця на зображені, що забезпечують легкий знайти, при цьому вони є досить інформативними. Прикладом є кути, перетини ліній, вершини фігур і багато чого іншого. Для того, щоб ключові точки були такими, потрібно, щоб на них не впливали зміни освітлень, масштабу чи повтору.

3. Опис ознак.

В даному етапі використовується дескриптора. Іншими словами – це побудова числових представлень, які містять певну інформацію про ключові точки у зображення. За рахунок даного опису , виконується перевірка різних точок на зображені або між зображеннями.

4.Формування набору ознак.

В даному етапі створюються ключові точки об’єднуються в набори ознак, що являє собою компактною формою зображення.

Також ознаки розподіляють за їх призначенням та рівнем деталізації, а саме: Локальні.

Основана мета даних ознак виявлення і порівняння невеликих ознак на зображеннях, наприклад як кути чи крах для подальшого пошуку відмінностей на них.

Глобальні.

Даний вид розпізнавання ознак включає в себе виявлення особливостей на всій ділянці зображення незалежно від обєта дослідження що допомагає в деяких робочих сценаріях.

Текстурні.

Даний розподіл оснований на пошуку особливостей, структурних властивостей поверхні зображення, наприклад певні шаблони чи градієнти.

Колірні.

Даний розподіл здійснюється за рахунок розподілу кольорів на зображені, наприклад, як гістограма кольорів.

Загалом виділення ознак має багато складносте і проблем, які виникають в процесі їх використання. Основні з них – це шуми, які можуть сильно змінити сприйняття зображень, різноманітність об’єктів, які спричинлять велике навантаження і будуть потребувати відповідних алгоритмів так, як складні умови зйомок чи великі об’єми даних для обробки.

1.2.4. Сегментація

Сегментація – це процес поділу зображення на певні області(сегменти), кожен з яких несе у собі певну особливу інформацію(характеристику) таку, як колір, форма чи текстуру. Основною задачею сегментації у технологіях РЗ і вчасності в роботах зв’язаних з КЗ – це виділення важливих об’єктів на фото та відео для подальшого їх аналізу.

Загалом існує багато підвидів сегментації, але в даній роботі будуть розглянуті основні чотири, а саме:

1.Семантична сегментація.

Принцип роботи даної сегментації полягає у детальному аналізі кожного з пікселів на зображені, після чого відбувається класифікація даної області. Дана сегментація має великий мінус, а саме те що вона не здатна розрізняти різні об’єкти, які знаходяться в одному класі. Прикладом семантичної сегментації зображений на рис 1.8, а саме розподіл дороги синім кольором, людей зеленим, транспорту червоним і всього іншого фіолетовим.

Рис. 1.8. Приклад семантичної сегментації

2. Інстанційна сегментація.

Інстанційна сегментація протилежна семантичній, і виконує трохи іншу роль, а саме вміння розпізнавати окремі об’єкти одного класу. Спочатку визначення межі кожного з об’єктів з використанням алгоритмів, а потім виконується їхня сегментація в просторі одного класу.

3.Паноптична сегментація.

Дана сегментація є об’єднанням двох попередніх сегментацій, а саме використання таких моделей, які одночасно враховують категорії пікселів, як в семантичній сегментації, так і межі об’єктів як в інстанційній.

4. Сегментація з урахуванням області.

Дана сегментація ділить зображення на окремі області на основі схожості пікселів в межах кожного з регіонів. Дана сегментація дуже популярна в моделях та алгоритмах зв’язаних з медичним обладнанням вчасності у апаратах МРТ.

1.2.5. Класифікація

Класифікація, є одним з найважливіших аспектів зв’язаних з КЗ, тому, що за рахунок неї відбувається відношення об’єктів на зображеннях на основі своїх ознак чи характеристик до відповідного класу, що потрібно для роботи великої кількості

алгоритмів та моделей зв’язаних з РЗ. Видів і підвидів класифікацій є теж дуже багато, але для економії часу, листків та загалом для розуміння вистачить чотирьох видів.

1. Бінарна класифікація.

Принцип роботи бінарної класифікації заключається у визначені до якого з двох класів належить зображення чи об’єкт на ньому. Для цього зазвичай використовують відповідні нейронні мережі, які передбачають до якого класу віднести зображення. Прикладом є перебірка зображень нейронною мережею зображень і поділ їх на зображення автомобілів(перший клас) та без автомобілів(другий клас).

2. Багатокласова класифікація.

Як стає зрозуміло з назви дана класифікація виконує вибірку не з двох класів з багатьох для обчислення ймовірності належності зображень чи об’єктів на них до відповідного класу. Тут прикладом визначення що є на зображенні наприклад легкова машина, вантажівка чи велосипед і розподіл зображень відповідно по класам.

3. Класифікація із використанням множинних міток.

В даному типі класифікації відбувається класифікація об’єктів на зображені до декількох класів. Умовно кажучи якщо на зображенні одночасно є пішоходи та автомобілі, а існують два різні класи «Транспорт» і «Люди» то при даному типі класифікації об’єкт і зображення на ньому будуть додані до обох класів.

4. Ієрархічна класифікація.

При даному типі класифікації, а конкретно при виникненні залежність одних класів(підкласів) від інших(батьківських) відбувається ієрархічний розподіл. Прикладом даного типу класифікації може послужити розподіл транспорту на дорозі, тобто є зображення велосипеда і спочатку буде відбуватися класифікація по батьківському класу, а саме в даному випадку він умовно буде називатися «Транспорт», а вже потім буде вибиратися до якого підкласу його віднести у даному випадку – це клас «Велосипеди».

1.2.6. Роль попередньої обробки зображень

Попередня обробка зображення – це процес перетворення зображення для ліпшого розуміння його комп'ютерним зором та його алгоритмами.

Основні цілі попередньої обробки – це поліпшення якості зображень, підготовка даних, а також зменшення(спрощення) зображень. Тепер конкретніше про кожен з них:

1. Поліпшення якості.

Як стає зрозуміло з назви пунктах одна з основних цілей даної обробки є поліпшення якості, а точніше усунення шумів, поліпшення контрасту та різкості зображення, або ж поліпшення його деталізації для ліпшого бачення дрібних деталей на ньому

2. Підготовка даних.

В даному етапі усі зображення підводяться до одного певного стандарту для облегшення і прискорення роботи алгоритмів, що забезпечує їх стабільність і швидкодію.

3. Зменшення складності.

В даному етапі основна роль – це спростити структуру зображення, для зосередження на конкретній «мішені», такій як кути, контури, краї чи текстури.

Опис основних методів попередньої обробки зображень:

4. Фільтрація.

Даний метод дозволяє позбавитися від більшої кількості шумів на зображені, що дозволяє створити більш контрастне зображення, яке буде мати більш точні краї та контури.

Загалом розрізняють два види фільтрації залежно від мети обробки, а саме фільтрація низькочастотна та високочастотна. Основна роль низькочастотної фільтрації є згладжування зображення та зменшення шумів. Прикладами таких фільтрів є «Гауссовий фільтр» чи «Медіанний фільтр». А основна роль фільтрації високих частот є виділення деталей зображення таких, як контури чи текстур. Прикладами такої фільтрації є «Фільтр Собеля» чи «Лапласіан».

4. Нормалізація та корекція зображення

Основною задачею нормалізації є вирівняння параметрів зображення, тобто поліпшення загального вигляду зображення, наприклад за рахунок розподілу інтенсивності на ньому. Отже основною властивістю даного методу є корекція яскравості та контрасту на зображені для видимості потрібних елементів на ньому. Прикладом застосування є використання «Гістограми вирівнювання» для зменшення темних ділянок за зображені.

5. Перетворення простору кольорів.

Також часто зображення для спрощення змінюють кольорову гамму для облегшення обчислень алгоритмів. Прикладами є чорно-білий формат, який змінює вид зображення на чорно-білий, що зменшує обсяг зображення і залишає інформацію лише інтенсивність і напрямок світла. Даний формат є дуже корисний коли кольоровою гаммою можна знехтувати для виконання основної цілі розпізнавання зображення. Також існують такі інтересні формати, як HSV чи YCbCr, які мають певні застосування для облегшення розпізнавання об’єктів на зображені. HSV розподіляє кольори на відтінки , що дозволяє вивчати кольорові властивості та аналізувати їх. У свій час YCbCr вміє виділяти яскравість об’єктів на зображенні чи відео окремо від кольорової гамми. Часто використовується у відеообробці за виділення тіней об’єктів.

6. Масштабування та вирівнювання.

В процесі аналізу зображення часто змінюють його масштаб чи розміри для зменшення обсягів інформації, або для прискорення даного аналізу. Вирівнювання в свою чергу є процесом виправлення орієнтації об’єкта для того щоб їх можна було порівняти з іншими зображеннями. Отримана попередні обробки дозволяють підвищити точність алгоритмів розпізнавання, оскільки забезпечують стандартизацію вхідних даних для подальшого аналізу. Дані дії є критично важливими для забезпечення стабільності роботи алгоритмів та мінімізації впливу шумів або спотворень у вихідному зображенні. Доволі широко використовується при машинному навчанні, особливо коли йде концентрація на конкретній цілі чи об’єкті, що сильно прискорює процес пошуку потрібних характеристик на зображені.

1.3 Висновок до розділу

Даний розділ охоплює у собі основні аспекти технології розпізнавання зображень. На початку було зроблено загальний і короткий огляд технології РЗ і що вона з себе представляє. В другому підрозділі було описано історію технології РЗ, а саме як забавка для вчених перетворилася у великий механізм який допомагає рішати багато проблем та облегшувати наше життя. А в кінці було описані основині види розпізнавання зображень, тобто фактично їх основні сфери для застосування та розвитку. Було проведено ознайомлення з існуючими методами для розпізнавання зображень. Спочатку був короткий аналіз поняття комп'ютерного зору, після чого був проведений аналіз найпопулярніших методів для розпізнавання зображень, а також об'єктів на них. Були обговорені такі важливі аспекти РЗ, як сегментація та класифікація та виявлення характеристик об'єктів на зображені.

РОЗДІЛ 2 ДОСЛІДЖЕННЯ СУЧАСНИХ ТЕХНОЛОГІЙ РОЗПІЗНАВАННЯ ЗОБРАЖЕНЬ. АЛГОРИТМИ ТА МОДЕЛІ ДЛЯ РОЗПІЗНАВАННЯ ОБЄКТІВ НА ЗОБРАЖЕННЯХ

2.1 Сучасні алгоритми розпізнавання зображень

Загалом алгоритми розпізнавання зображень є дуже багато і вони основані на різних технологіях залежно від потреб і функцій які вони виконують у даному пункті будуть розглянуті основні з них буде проведений їхній опис аналіз а в кінці порівняння у вигляді таблиці з усіма їхніми плюсами та мінусами. Загалом будуть розглянуті по п’ять алгоритмів класичній і глибоких алгоритмів і будуть наведень їх сфери застосування.

2.1.1. SIFT (Scale-Invariant Feature Transform)

Класичний алгоритм для виявлення ключових точок об’єктів та їх опису за зображені, на які не впливає не зміна масштабу, не зміна освітлення чи масштабування та багато інших факторів не впливають на його працездатність. Основною ціллю даного алгоритму є визначення важливих точок на зображені, а потім використання цієї інформації для подальшого відстеження чи порівняння на об’єктів зображеннях.

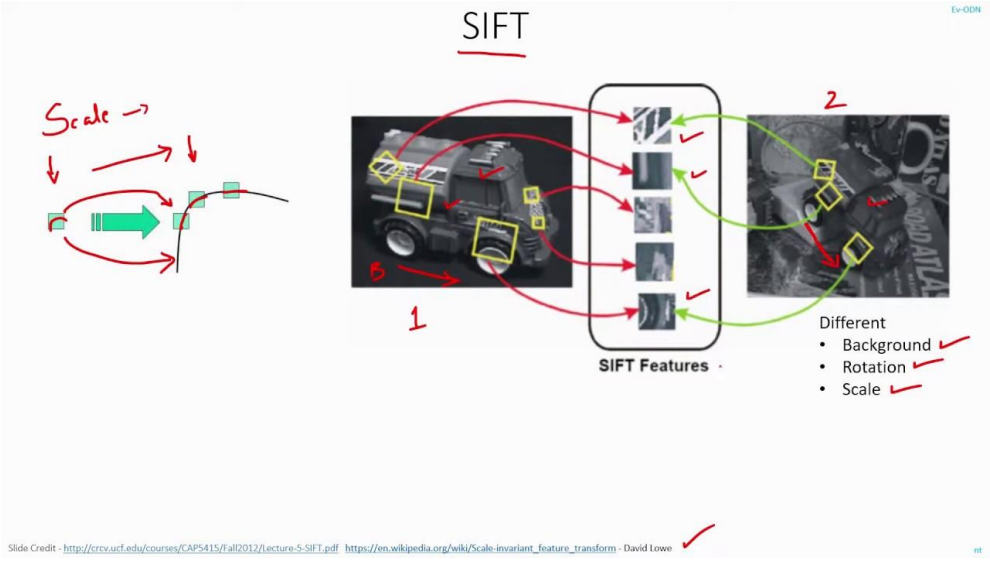

Рис. 2.1. Приклад роботи алгоритму SIFT

Процес роботи даного алгоритму поділяється на декілька ключових етапів, а саме:

1. Виявлення ключових точок, а конкретніше точок з високим контрастом чи зміною інтенсивності світла для облегшення подальшої роботи алгоритму.

2. Створення опису ключових точок для кожної з ключових точок та надання їй певний набір характеристик, при цьому точки має «імунітет» до таких змін як зміна масштабу чи обертання.

3. Перевірка точок на відповідність. В даному етапі виконується фактична перевірка, тобто чи точки відповідають зображенням, по іншому кажучи їх співставляють між собою.

З цього стає зрозумілим, що основним застосуванням даного алгоритму є співставлення зображень та виявлення на них схожості, також широко даний алгоритм використовується у деяких компаніях для реконструкції об'єктів у 3D, бо він здатний з декількох зображень 2D реконструювати в 3D об'єкти.

Загалом алгоритм дуже цікавий має високу надійність і високу продуктивність, але і має ряд мінусів. Один з основних на мою думку – це те що починаючи з 2020 року він отримав патент і його використання потребує відповідних документів. Також в мінуси можна віднести і потреба у великих обчислювальних ресурсах, для коректної роботи та обробки зображень, в результаті чого виникає ще один мінус, а саме не можливість коректно, відзиваймо працювати у реальному часі, тому даний алгоритм з точки зору логіки має більше мінусів ніж реальних плюсів порівняно з іншими.

2.1.2. SURF (Speeded-Up Robust Features)

Даний алгоритм є удосконаленням SIFT. Порівняно з попереднім він набагато швидше виконує обробку зображень, хоча і працює за схожим принципом. Даний алгоритм використовує методи на основі лінійних фільтрів для виявлення ключових точок і для їх опису. Як і попередній алгоритм здатний не сприймати масштабування і обертання, та працювати з різним освітленням.

Ключові етапи схожі з попереднім алгоритмом, а саме:

1. Доволі швидке виявлення ключових точок для чого використовуються вище вказані лінійні фільтри, вчасності фільтри Гаусса для прискорення процесу пошуку точок.

2. Ідентичний попередньому алгоритму, а саме опис характеристик для ключових точок, правда і тут воно проходить швидку бо використовується набір локальних ознак який базується на основі фільтрування та статистичних операціях.

3. І ключовий етап співставлення для перевірки на відповідність точок на зображені шляхом співставлення відстаней між їхніми описами.

Хоча і даний алгоритм має ряд переваг порівняно з своїм попередником, а саме велика швидкість обчислення, він має і ряд мінусів. Одним з основних мінусів є погана стійкість до деформації об’єктів, тобто при ушкоджені об’єкта на зображені даний алгоритм може перестати його розлічати, що може бути дуже критичним у багатьох ситуаціях. Також в нього лишилися і мінуси попередника, а саме висока обчислювальна складність, що робить його дуже слабким конкурентом у багатьох сферах. Також недоліком даного алгоритму є патент не дуже критичний недолік, але він є.

Сфера застосування у даного алгоритму збільшилася і тепер крім статичних зображень він здатний розпізнавати чітко зображення у реальному часі, що є великим прогресом у плані технологічності.

2.1.3. HOG (Histogram of Oriented Gradients)

Даний алгоритм сильно відрізняється від попередніх і принципом розпізнавання і технологічністю. Фактично – це алгоритм опису зображення, що використовує орієнтацію градієнтів у малих частинах зображення для виділення певних ознак, особливостей чи характеристик об’єктів, зазвичай для виявлення форм людських фігур).

Ключові етапи алгоритму:

1. Обчислення градієнтів, а точніше на зображенні обчислюється зміна яскравості для кожного з пікселів

2. На другому етапі зображення розбивається на блоки в яких обчислюються гістограми орієнтацій градієнтів.

3. В третьому етапі відбувається процес нормалізації гістограм, що дозволяє даному алгоритму бути стійким до інтенсивності зміни освітлення на зображені.

Система має великий ряд застосування починаючи від систем відеоспостереження по виявленню об’єктів закінчуючи чи пішоходів на дорогах. Також за рахунок того що алгоритм є досить простий його можна використати у невеликих проектах з не дуже об’ємним обсягом інформативних даних.

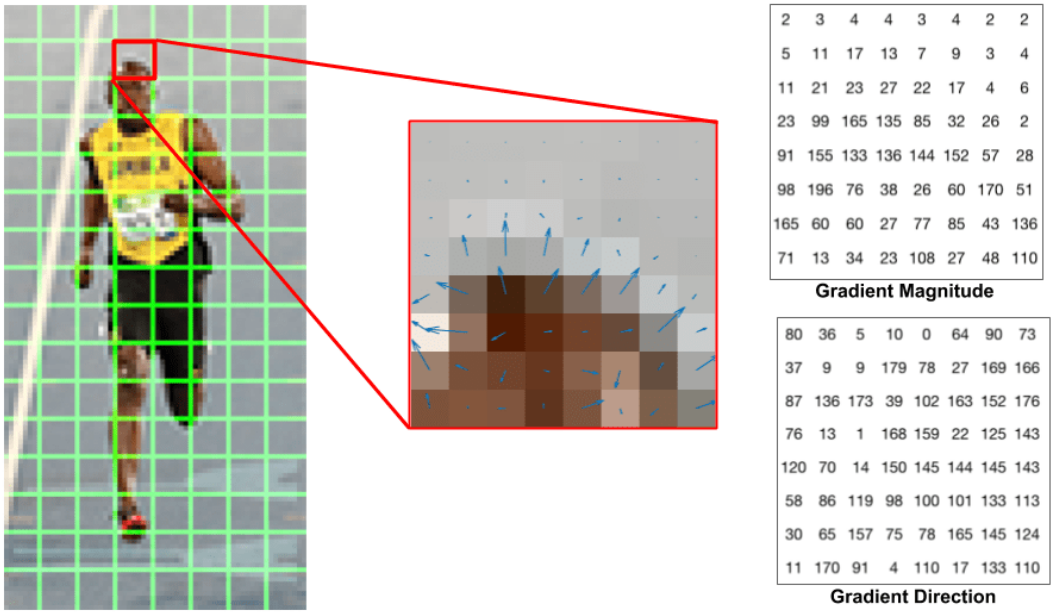

Рис. 2.2 Процес обчислення градієнтної величини

Загалом система має певний ряд плюсів достатньо добре розпізнає не дуже чіткі об’єкти на дорогах і не тільки, з близька може розрізняти лиця і може ігнорувати невеликі зміни у об’єктів, але має і ряд мінусів.

Даний алгоритм навідмінно від попередніх дуже чутливий до масштабування і через це здатний дуже псувати результати обчислень, також даний алгоритм майже як і всі класичні потребує великої кількості обчислювальних ресурсів, що не дуже добре впливає на кінцевий результат. Ще значним мінусом є те, що алгоритм HOG погано моментами може бути чутливий до освітлення

особливо в наборі з поганим масштабуванням, що може призвести до поганого виконання алгоритму, що призведе до не бажаних результатів в процесі застосування. Однак, при правильному налаштуванні можна значно покращити його продуктивність і зменшити вплив цих недоліків.

2.1.4. ORB (Oriented FAST and Rotated BRIEF)

Даний алгоритм є «гібридом» і походить від двох інших а саме з FAST для швидкого пошуку ключових точок та BRIEF для їх опису з використанням модифікації оброки обертання. За рахунок своєї швидкодії алгоритм невідмінно від попередників ідеально підходить для мобільних і браузерних додатків.

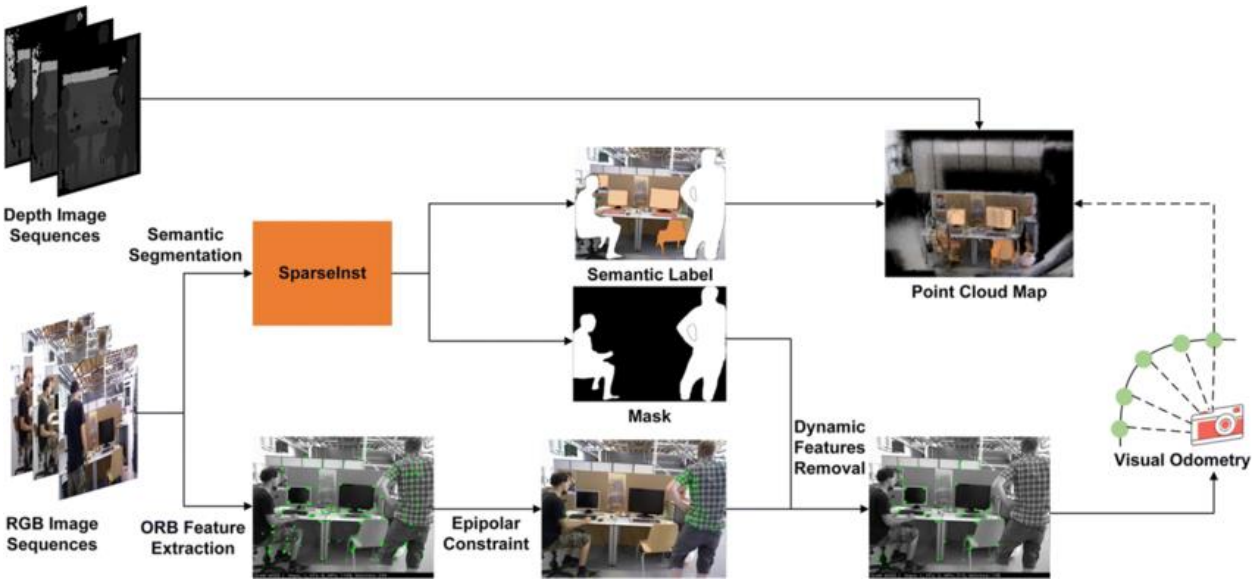

Рис. 2.3. Алгоритм візуальної з ORB та сегментацією

Основні етапи роботи:

1. Виявлення ключових точок з використанням алгоритму FAST. Даний алгоритм є доволі швидким і він шукає ключові точки на зображені з високою контрастністю, які можуть бути потенційно ключовими для нього.

2. Описання точок з використання BRIEF за рахунок даного алгоритму створюється унікальний бінарний опис ключових точок зображення, що дозволяє йому лишатися стійким при зміні освітлення чи обертанні зображення.

3. Використання модифікацій для обертання. Алгоритм ORB враховує невеликі чи значні повороти зображені, що дозволяє йому лишатися стійким до обертів.

Загалом даний алгоритм має декілька сфер застосування, одна основних – це відстеження рухів на зображеннях чи відео у реальному часі, а також в робототехніці для навігації та розпізнавання об’єктів роботами. Ще має широке застосування в мобільних додатках через хорошу оптимізацію і швидкодію алгоритму.

З плюсів у програми є швидкодія алгоритму, що було сказано раніше. Також з плюсів є хороша оптимізація та ефективна робота з повернутими об’єктами. Мінусів у даного алгоритму теж хватає, а саме чутливість до шумів, а також він гірше порівнює об’єкти ніж інші алгоритми. Ще значним мінусом є те, що при обробці складних сцен, чи деформації об’єктів алгоритм починає некоректно працювати що призводить до негативних наслідків при користуванні.

2.1.5. RANSAC (Random Sample Consensus)

RANSAC доволі специфічний алгоритм з своїми особливостями та інтересним принципом роботи, а саме він виконує випадкову вибірку для оцінки стійкості зображення до шуму з використанням математичних моделей. Основною специфікою застосування є пошук геометричних об’єктів таких як кола, прямі, площини, криві у наявності великої кількості шумів.

Ключові етапи роботи даного алгоритму:

1. Створення випадкової вибірки множини точок. Алгоритм випадково вибирає та створює підмножину точок для побудови моделі.

2. Обчислення моделі. У даному етапі обчислюється підмножини моделі, наприклад площини чи інші геометричні об’єкти.

3. Виконується перевірка на узгодженість точок. Тобто виконується перевірка скільки точок з множини підходять для обчислення в моделі, визначають її правильність.

Застосування у даної моделі є декілька одне з них це виявлення дорожнього покриття та знаків на дорогах, а інше більш цікавіше – це реконструкції 3D-моделей з множини 2D зображень.

2.1.6. YOLO (You Only Look Once)

Що дослівно переводиться «Ти дивишся лиш один раз». Доволі популярний метод, який широко застосовується для виявлення певних об’єктів на зображеннях для подальшої взаємодії з ними. Одним з основних вмінь даного методу є те, що він одночасно вміє і виявляти об’єкт на зображенні і одразу його класифікувати, при цьому всьому він це може робити у живому часі тобто, що може бути потрібне у деяких випадках.

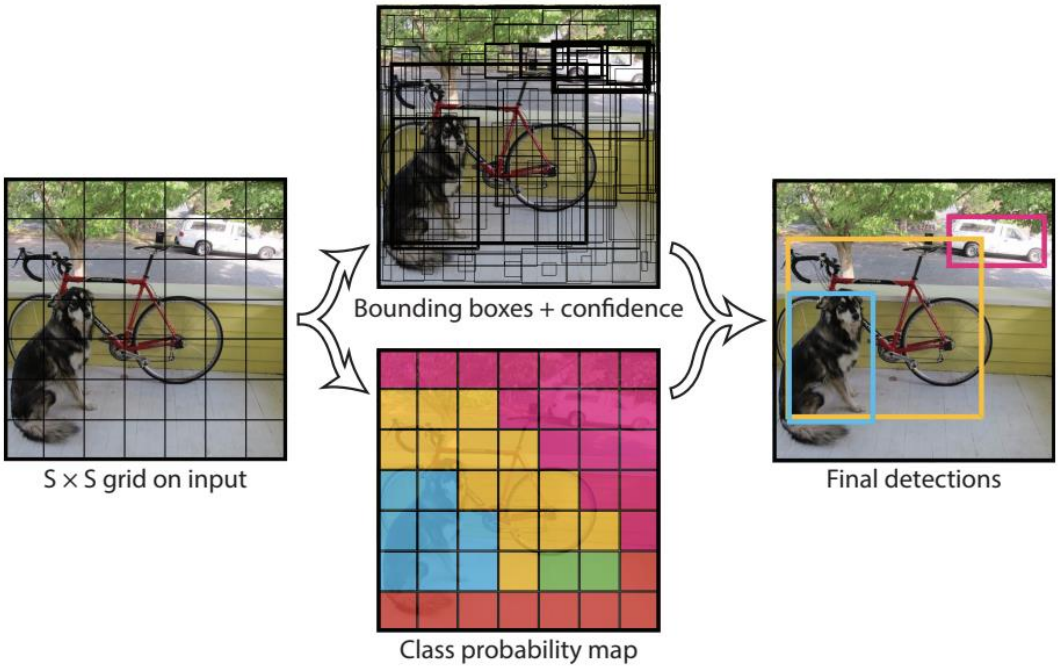

Рис. 2.4. Приклад застосування моделі YOLO

Основною особливістю є поділ зображення сіткою на певну кількість сегментів (умовних квадратів) з врахуванням для кожного з них координат обмежувальних рамок для кожної клітинки, що допомагає виявити ймовірностями розташування об'єкта на тих чи інших координатах, що допомагає і прискорює класифікацію об’єктів на зображення.

Покроковий принцип YOLO алгоритму:

- Поділ зображення сіткою на певну кількість сегментів.
- Визначення об’єктів на зображенні. Кожна клітинка в сітці YOLO має декілька обмежувальних рамок, інакше їх називають – bounding boxes. Кожен з них в свою чергу вираховує вірогідність того що в тій чи іншій клітинці розташований об’єкт.
- Класифікація об’єктів. Простіше кажучи визначення потрібного класу для класифікації(машини, дерева, люди, вікна, тварини, тощо) для подальшої оброки.
- Вичислення ймовірностей та координат об’єктів на зображеннях. Алгоритм починає прораховувати координати рамок і ймовірності розташування для кожного об’єкта які були потрібно було виявити.

Тепер про сфери застосування. Алгоритм широко використовується для автопілотів, по тій причині що він в реальному часі вміє розпізнавати та класифікувати знаки автомобілі і пішоходів для подальшої взаємодії з ними(наприклад екстерне гальмування при виявлені транспорту занадто близько до машини).

Також за рахунок своїх вище перечислених особливостей YOLO використовують в камерах відеоспостереження, або для пошуку осіб по зображеннях(наприклад для ідентифікації осіб підозрілих осіб з баз даних, або як варіант для проходження певного фейс контроля на деяких контрольно пропускних пунктах відповідно до допуску, як альтернатива їм бо лице тяжче підмінити ніж пластику карточку з допуском). Інтересною сферою застосування є у сфері технологій зв’язаних з доповненою реальністю, а саме розпізнання об’єктів в реальному часі і підмінювати на потрібний контент у віртуальній частині інтерфейсу користувача.

2.1.7. Faster R-CNN (Region-based Convolutional Neural Networks)

Такий же дуже популярний метод , як і YOLO. Попри це його принципи і швидкодія відлучаються від попередника багатьма аспектами. Він заключається у

об’єднанні методу пропозицій регіонів (Region Proposal Networks, RPN) з традиційними методами класифікації. В результаті отримується система, яка з великою точністю вміє дуже точно виявляти об’єкти для класифікації хоча має і ряд мінусів порівняно з іншими.

Ключові етапи алгоритму Faster R-CNN:

1. Відбувається побудова регіональної пропозиції мереж, або ж скорочено RPN.

2. Генеруються області інтересу на зображені.

3. Далі проводиться класифікація для кожної точки інтересу на зображені для подальшого аналізу.

4. В даному етапі визначаються та відтворюються обмежувальні рамки об’єктів на зображені.

5. Застосування технології NMS для зменшення перекриття рамок для об’єктів на зображені.

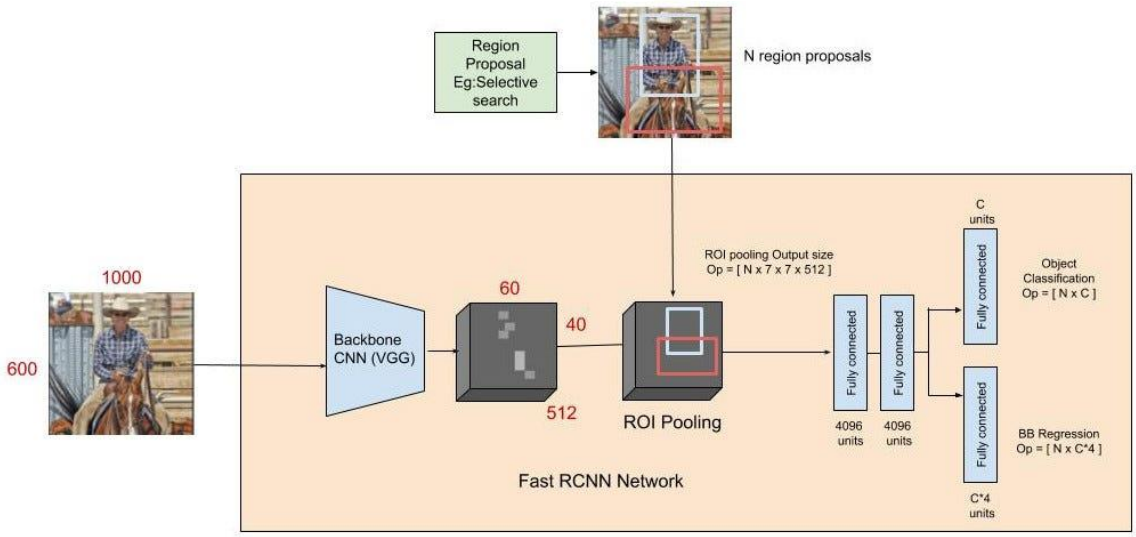

Рис. 2.5. Використання Fast R-CNN для виявлення об'єктів на зображені

Сфер застосування у даного алгоритму є декілька один з основних – це виявлення об’єктів (часто в наукових цілях) у складних умовах для подальшого їх аналізу. Також через свої особливості часто використовується в медицині для

виявлення пухлин чи інших відхилень в організмі людей, що може значно прискорити процес лікування хворих пацієнтів особливо у початкових стадіях.

2.1.8. Backpropagation (Зворотне поширення помилки)

Backpropagation – це дуже популярний, передовий, ключовий та зручний алгоритм для навчання багатошарових нейронних мереж. За рахунок використання ітераційних процесів даний алгоритм дозволяє оперативно та з доволі великою точністю оновляти дані в нейронній мережі, що призводить до зменшення помилок(змінює в позитивну сторону різницю між передбаченням і правильними результатами). Даний алгоритм використовує методи градієнтного спуску для корекції кожного шару на основі отриманої похідної функції втрат.

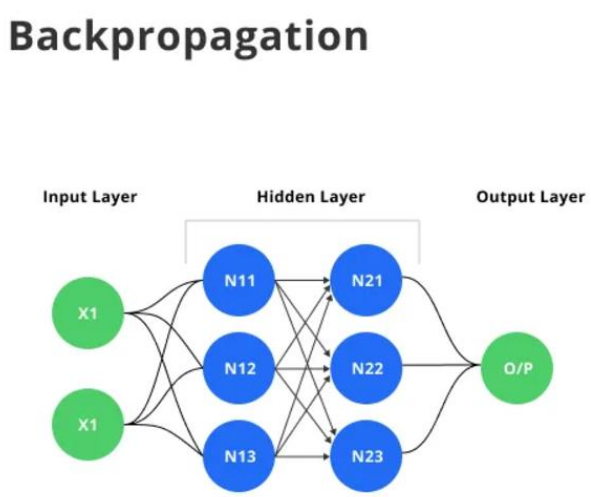

Рис. 2.6. Схема роботи алгоритму Backpropagation

Ключові етапи роботи алгоритму:

1. Forward pass. В даному етапі отримані вхідні дані проходять через всі шари нейронної мережи, при цьому на кожному з шарів виконується обчислення ваг активацій. Також виконується порівняння вихідних даних з очікуваним результатом(тестовими даними) за допомогою функції втрат.

2. Обчислення значення функції втрат. В даному етапі виконується обчислення помилок між передбаченням та фактичним результатом. Як приклад для задач регресії вираховують середньоквадратичну помилку, а для задач класифікації – крос-ентропію.

3. Backward pass. А тепер за допомогою правила ланцюга виконується обчислення для кожного з параметрів градієнти функції втрат. Фактично виконується передача градієнтів від вхідного до вихідного шару.

4. Оновлення ваг. В даному етапі відбувається оновлення ваг з використанням методу градієнтного спуску на основі обчислених градієнтів які були отримані для зображення.

Завдань для даної моделі є дуже багато не лиш у сферах зв’язаних з графікою, але як приклад – це навчання багатошарових персептронів для задач класифікації чи Створення моделей для розпізнавання мови та зображень у складних умовах.

Переваг та недоліків такий алгоритм має масу. До плюсів треба віднести велику ефективність у навчанні складних моделей з багатьма параметрами, універсальність бо може бути використана для рішення великої кількості задач не лиш зв’язаних з зображеннями на пряму, але загалом задачі класифікації чи регресії. До недоліків можна віднести часточкове або повне зникнення градієнтів у глибоких мережах він(градієнт) здатний зменшуватися на стільки, що перестають впливати на ваги у алгоритмі.

2.1.9. Adam Optimizer

Adam або ж по іншому Adaptive Moment Estimation – це доволі сучасний і зручний алгоритм для оптимізації, який об’єднує в собі адаптивні методи навчання і моментів (наприклад, SGD з моментом). Даний алгоритм вміє автоматично регулювати швидкість навчання для кожного з параметрів навчання залежно від дисперсії і середнього значення градієнтів, що робить його доволі зручним, гнучким та стабільним у певних задачах машинного навчання.

Ключові етапи роботи алгоритму Adam:

1. Виконується обчислення градієнтів, а точніше вичислюється для кожного параметра значення градієнта функції втрат.

2. Момент першого порядку (m). На даному етапі визначається експонеційно згладжене середнє значення градієнтів.

3. Момент другого порядку (v). А на даному етапі вже визначається дисперсія градієнтів і вона теж експонеційно згладжена.

4. Виконується корекція моментів. Для компенсування початкових зміщень моментів здійснюється обчислення значень m та v.

5. Оновлення ваг. В даному етапі здійснюється адаптивне оновлення ваг з використанням вже обчислених моментів.

Даний алгоритм за рахунок своєї універсальності здатний виконувати дуже багато задач зв'язаних, як з темою даної роботи так і з багатьма іншими сферами за рахунок роботи з великою кількістю різних глибоких нейронних мереж та здатності з його допомогою оптимізувати великі моделі які зв'язані з комп'ютерним зором, а саме задачі класифікація та виявлення об'єктів на зображеннях, реалізовувати процеси сегментації та суперрезолюції зображень, обробку медичних зображень і багато ще чого іншого зв'язаного розпізнаванням зображень та об'єктів на них.

Плюсів даний алгоритм має набагато більше ніж мінусів, а саме велика швидкість навчання, адаптивність для швидкого навчання конкретних параметрів та висока стабільність навіть з невеликою кількістю даних та не дуже чіткими градієнтами. Недоліки в нього теж хватає один з основних на мою думку є стандартний для подібних моделей, а саме їх дуже легко перенавчити і всі значення вона почне видавати не коректними, тобто умовно на картинці вона буде казати і велосипед і автомобіль одне і теж і буде класифікувати їх не правильно. Також ще критичним мінусом для деяких задач є складність налаштування, а точніше вибирати значення гіперпараметрів. Ще моментами є нестабільна робота з великою кількістю локальних мінімумів, що призводить до деяких проблем у певних задачах. Для стабільної роботи алгоритму часто рекомендують використовувати регуляризацію, для запобігання перенавчання алгоритму.

2.1.10. Grad-CAM (Gradient-weighted Class Activation Mapping)

Grad-CAM – це дуже популярний алгоритм, який використовується для пояснення роботи певних нейронних мереж, особливо часто використовується для класифікації зображень. Основний принцип даного алгоритму – це створення теплових карти, які показують, яка області зображення найбільше впливають на передбачення об’єктів та елементів зображення. Grad-CAM по функціональній частині схожий на інші ШІ алгоритми, але має і свої певні особливості, а саме на основі градієнту функції втрат будую карту для передбачення зображення.

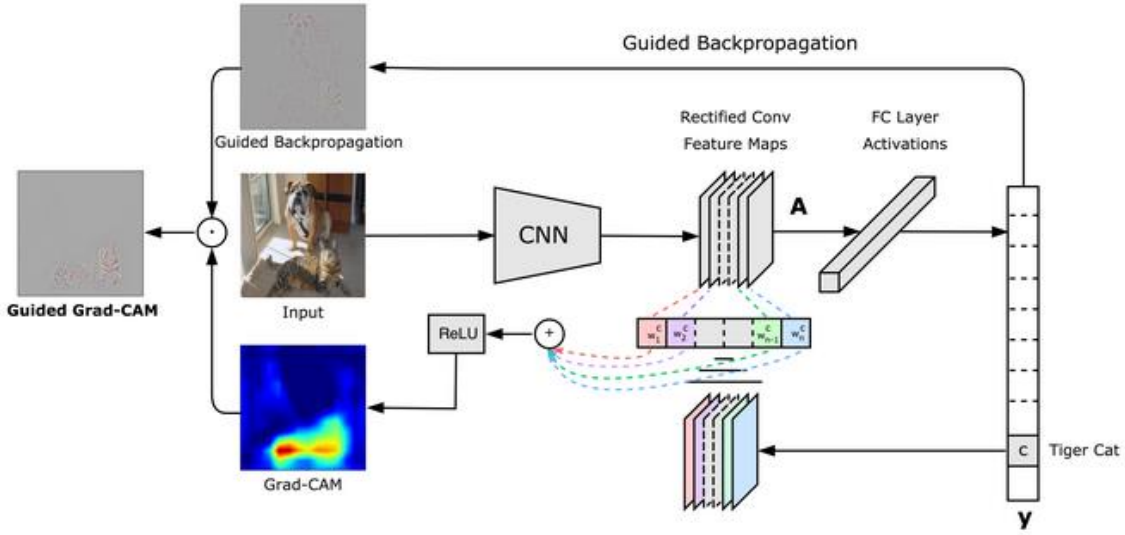

Рис. 2.7. Архітектура Grad-CAM із Guided Backpropagation

Принципи роботи алгоритму Grad-CAM схожі чимось з іншими алгоритмами ШІ, але є свої відмінності:

1. Forward pass. За рахунок використання нейронної мережі виконується передача зображення і отримується передбачення для нього.

2. Обчислення градієнтів. Виконується обчислення щодо активації основного шару градієнтів функції втрат.

3. Зваження активацій. Отримані градієнти з попереднього пункту використовуються для зваження відповідних активацій.

4. Агрегація. В даному етапі підсумовуються отримані результати та формуються карти активацій.

5. Візуалізація. А тепер карти активацій накладаються на вхідне зображення і створюють теплову карту для нього.

Основною задачею даного алгоритму є оцінка і аналіз роботи моделей комп'ютерного зору та часто застосовується в новітніх апаратах КТ для виявлення точок інтересу таких як пухлини для швидкого їх аналізу. Переваг для аналізу зображень даним алгоритмом є багато, а саме простота застосування, також допомагає швидко зрозуміти на які елементи зображення модель звертає увагу у першу чергу. Також даний алгоритм дозволяє діагностувати моделі і знаходити в них помилки. До мінусів можна віднести погане розпізнавання тонких контурів та деталей зображення. Ще одним мінусом даного алгоритму є те, що одночасно карта будується лиш для одного передбаченого класу і при цьому карти не завжди інтуїтивно зрозумілі і мають мало перспективи бути використаними в задачах де є вибірка з великої кількості різних класів.

2.2 Сучасні моделі розпізнавання зображень

Поняття модель дуже обширне поняття у нашому житті. В програмуванні і вчасності в сфері РЗ вони виконують важливу роль де не хватає функціоналу одного алгоритму. Простіше кажучи модель РЗ являє собою певну програмну архітектуру, яка використовує конкретні алгоритми для розв'язання поставлених задач.

Моделі як і звичайні алгоритми можуть базуватися як і на простих алгоритмах так і з використанням глибокого навчання та технологій ШІ залежно від поставлених задач у неї. В даному пункті будуть розглянуті різні моделі для РЗ та буде виконаний короткий їх опис, недоліки, переваги та фактичні сфери використання.

Ще хочеться додати, що різні моделі РЗ дозволяють адаптуватися до змін у вхідних даних, що корисно використовувати для обробки великої кількості інформації, забезпечувати гнучкість у вирішені складних чи критичних ситуаціях.

2.2.1. SVM + HOG

Являється класичним методом розпізнавання зображень, який базується на поєднані методу опорних векторів з алгоритмом гістограм орієнтовних градієнтів, які по іншому називається HOG. Про алгоритм роботи HOG в даній роботі вже був розбір і не потрібно повторюватися, а SVM(support vector machine), а точніше методу опорних векторів – це це метод аналізу даних для класифікації та регресійного аналізу за допомогою моделей з керованим навчанням з пов'язаними алгоритмами навчання, які називаються опорно-векторними машинами.[1]

Загалом робочий процес в даному методі розбитий наступним чином, а саме HOG витягує потрібні ознаки, які описують контури і структуру об’єктів, а SVM допомагає класифікувати зображення.

Основною сферою застосування такої моделі є розпізнавання облич через його простоту реалізації у подібних задачах. Також може використовуватися для нескладних систем безпеки з використанням відеоспостереження чи загалом для аналізу відео.

Недоліків даний метод має не дуже багато бо його фактична специфікація заключається у розпізнавані зображень і не лише зображень і з цією задачею він справляється досить добре. До недоліків можна віднести – це складність у роботі з великими потоками даних та необхідність ручного витягнення ознак із зображень. Але плюси теж доволі значні бо враховуючи основну сферу застосування моделі вона здатна доволі швидко, а саме головне що ще й точно вирішувати прості задачі у сфері РЗ.

Покроковий алгоритм роботи моделі:

1. Попередня обробка зображення, а саме перетворення його у відтінки сірого тону.

2. Обчислення для кожного пікселя напрямку і величини градієнтів.

3. Відбувається розбір зображення на малі блоки і створюється для кожного з них гістограма орієнтацій градієнтів, формується гістограми ознак.

4. Після формування гістограми ознак відбувається їх нормалізація для підвищення їх стійкості в умовах зміни освітлення.

5. І в даному етапі ознаки передаються до SVM, де проходить процес прийняття рішень, до якого класу їх віднесуть.

2.2.2. k-NN + PCA

Модель є доволі цікава і об’єднає в собі властивості алгоритму PCA для зменшення розмірності ознак та збереження важливої інформації про зображення та використання методу k-найближчих сусідів для класифікації отриманих ознак.

Покроковий алгоритм роботи моделі:

1. Підготовка вхідних зображень а точніше перетворення їх у вектори.

2. Виконання обчислень з PCA для пошуку головних компонентів зображення, а правильніше кажучи пошук вектор, який має найбільше номінальне власне значення.

3. Тепер виконується перевірка отриманих зразків з тестовими даними у навчальній вибірці.

4. І на кінець виконується класифікація k найближчих сусідів, що означає що клас буде визначений шляхом «голосування» найближчих сусідів.

Дана модель як і попередня добра справляється із задачами зв’язаними з класифікацією, а саме розпізнавання облич і об’єктів на зображеннях і камерах.

Головні плюси ідентичні попередній моделі, а саме те що вона добре справляється з невеликим наборами даних з дуже непоганою точністю. А з цього виникає значний мінус зв’язаний з складністю обробляти великі обсяги інформації і зв’язано це не в останню чергу через чутливість моделі до вибору k-найближчих.

2.2.3. Random Forest + Haar Features

Дана модель є дуже цікавою з точки зору глибокого навчання бо допомагає рішити деякі питання зв’язані з ним вчасності затухання градієнтів. Вона

використовує залишкові зв’язки для вирішення даної проблеми, що ідеально підходить для тренування дуже глибоких мереж.

Принцип роботи моделі:

1. По перше використовуються хаарові ознаки для опису текстур об’єктів на зображені.

2. В даному етапі створюються і навчаються дерева рішень із випадковими підмножинами даних, при цьому кожне дерево виконує незалежне одне від одного прогнозування.

Задач з якими може впоратись дана модель з великою точністю доволі багато і в основному вони всі базуються на сегментація, класифікація зображень чи розпізнавання об'єктів на них.

Що до плюсів та мінусів то вони обидва значні. Плюсів основних є два, а саме дуже велика точність без виникнення такого терміну, як перенавчання та глибока архітектура, що дозволяє рішати доволі не стандартні задачі зв’язані з РЗ.

2.2.4. LeNet-5

LeNet-5 – це одна з найперших CNN, яка була спеціально розроблена для розпізнавання рукописного тексту. Структура її не дуже унікальна на сьогоднішній день і вона складається з декількох згорткових і повнозв'язних шарів.

Принцип роботи даної моделі наступний:

1. Зображення нормалізується та передається до мережі.

2. Виділяються локальні ознаки такі, як краї чи текстури у згортковому шарі.

3. Після чого зменшується кількість ознак при цьому зберігається важлива інформація, яка потрібна для подальшого аналізу.

4.У даному етапі, а точніше у повновязному шарі відбувається узагальнення інформації для кінцевої класифікації зображення.

Сфера застосування даної моделі є великою і зв’язана з усіма сферами де є текст і потрібно його розпізнавати тобто починаючи від систем розпізнавання знаків на дорогах, закінчуючи самими різними системами оцифрування даних

таких, як старі документи чи книжки для переносу їх в формат який здатний зберігатися не одне століття.

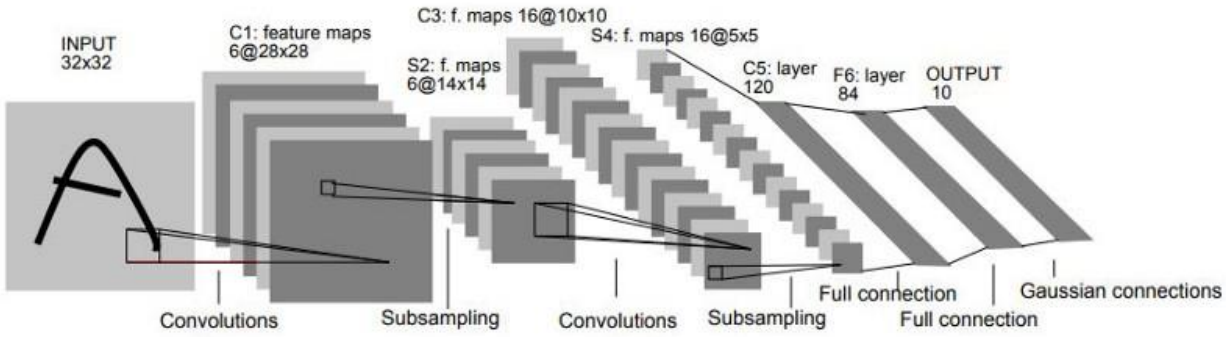

Рис. 2.8. Принцип роботи LeNet-5 для розпізнавання рукописних цифр

Переваги виникають з сфери застосування, а саме те що модель дуже добре розпізнає тексти і цифри. А напевно до недоліків можна віднести, що одночасно вона не здатна апелювати їхньою великою кількістю, що негативно впливає на кінцевий результат.

2.2.5. AlexNet

Дана модель є доволі передовою у своєму роді бо вона одна з найперших на базі CNN яка вміє проводити глибокий аналіз та обробку великої кількості даних з використанням шарів згортки та глибокі структури.

Основною метод і завданням даної моделі було і є – це класифікація зображень у великих наборах даних, для подальшого їх аналізу і використання іншими моделями чи алгоритмами.

Суттєвий недолік у даної моделі лиш один, а саме потреба у великій кількості обчислювальних потужностей. А переваги набагато більші, а саме неймовірна на час свого створення точність, а також передова система(техніка) для боротьби з перенавчанням Dropout.

Принцип техніки полягає в тому, що в процесі навчання із загальної мережі випадковим чином виділяється підмережа, для якої здійснюється навчання. Після

навчання обраної підмережі випадковим чином обирається нова підмережа і навчання продовжується.[2] Також вибір нейронів у підмережах виконується з хаотичною ймовірністю, яку називають коефіцієнтом дропаута.

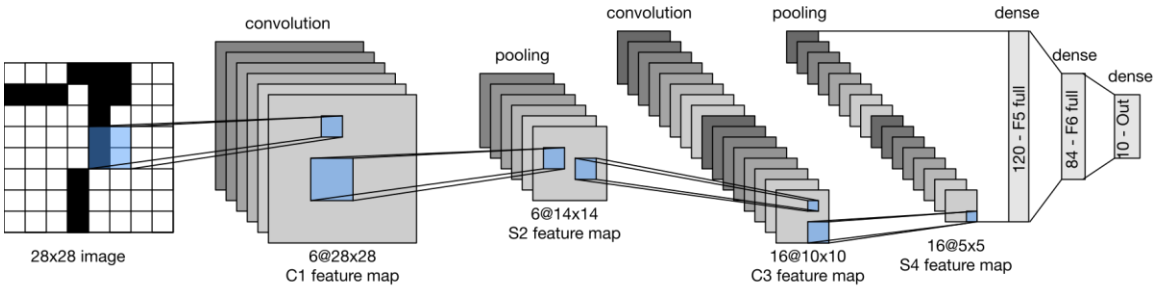

Рис. 2.9. Принцип роботи моделі AlexNet для класифікації об’єктів на зображені

Принципи роботи моделі AlexNet:

1. Введення даних, а точніше зображення для його фіксації та масштабування.

2. У згорткових шарах виділяються ознаки різної складності.

3. На даному етапі шари pooling зменшують розмір, а нормалізація почина процес стабілізації навчання.

4. Після чого відбувається завершення класифікації у повнозв’язному шарі.

2.2.6. Inception (GoogLeNet)

Inception – це доволі ефективна модель з модульною структурою створена для роботи з великими зображеннями та їх класифікація. Принцип роботи наступний спочатку в Inception модулі виконується паралельні згортки зображення різних розмірів. Далі у шарі pooling перед фільтрацією виконується зменшення зображення до стандартного розміру. І в кінці через повнозв’язний шар виконується класифікація зображень. Реальна сфера застосування для такої масштабної моделі одна, а саме класифікація великих за розміром і обсягів

зображень. Ще причиною такого використання даної моделі є те що вона має доволі складну структуру, аби її перенавчити для інакшої задачі.

2.3 Опис, огляд та порівняння існуючих програмних рішень для OCR

Загалом дане питання є дуже цікаве і за свою історію мало не один десяток різних програмних рішень, починаючи від класичних методів, які годинами могли вчити розлічати стандартні друковані цифри та букви, закінчуючи сучасними програмними засобами, котрі здатні розпізнавати рукописні тексти і одночасно обробляти їх великі обсяги. В даному розділі будуть розглянуті деякі існуючі популярні програмні рішення для розпізнавання текстів із зображень.

2.3.1. Tesseract OCR

Це один з найпопулярніших алгоритмів для розпізнавання текстів на фотографія, який широко використовується в багатьох сферах де використовується технології OCR. Перша його версія була створена компанією Hewlett-Packard(HP) ще в далекому 1985 році для своїх принтерів і використовувався він у багатьох їхніх продуктах.

Фактично до 2005 року код алгоритму був закритий і користуватися у відкрити ним не могли, але у цьому ж році він став доступним для спільноти у всьому світі, що дало змогу ентузіастам його розвивати відповідно для своїх задач. З 2006 року підтягнулися більші «гравці» такі, як Google і почали активно виділяти ресурси на його розвиток.

За рахунок цього на сьогоднішній день він доволі сильно розвинувся, що призвело до його популяризації і активного використання у багатьох додатках для розпізнавання тексту, до яких включно належать різні мобільні програми та корпоративні рішення. Наразі даний алгоритм навчений більш ніж на 100 мовах і є одним з найточніших інструментів для OCR. Також на сьогоднішній день Tesseract OCR використовує метод машинного навчання для розпізнавання тексту, а не класичні методи.

Також він здатний бути навчений новим мовам при потребі відповідно до завдання. Даний алгоритм має дуже велику кількість підтримки форматів зображень, а самі популярні – це TIFF, PNG, JPEG, а також формат PDF для документів. А вихідні дані отримуються у результаті вихідного тексту, який може бути переданим не лиш у форматі TXT але і в форматі HOCR (HTML OCR) чи PDF (OCR).

Переваги:

1. Висока точність.

За рахунок використання нейронної мережі LSTM в значній мірі виросла точність розпізнавання текстів.

2. Відкритий код та собівартість.

За рахунок цього не потрібно ніяких ліцензій для отримання доступу для програмного продукту, а також можливі модифікація окремих аспектів за потреби.

3. Підтримка великої кількості форматів та мов.

Як було сказано раніше даний алгоритм може підтримувати дуже велику кількість форматів як вхідних так і вихідних.

4. Можливість довчити модель.

Крім освоєння окремих мов даний алгоритм здатний ще бути навчений для специфічних умови та окремі видів складних шрифтів.

Недоліки:

1. Швидкодія.

Порівняно з більшістю OCR алгоритмами даний є доволі повільним, що стає значно замітнішим при обробці великих за обсягом зображень.

2. Слабка ефективність у складних умовах.

Даний алгоритм має певні недоліки у складних умовах таких, як слабке освітлення, велика кількість шумів, рукописні тексти то алгоритм може стати малоефективним в таких умовах.

3. Складність інтеграції. У процесі інтеграції у мобільні додатки чи десктопні рішення Tesseract може дуже не коректно працювати, про це буде згадуватися у другому розділі бо і у процесі виконання даної магістерської роботи виникли

проблеми з його інтеграцією. Причиною є специфічні умови для форматів даних та зображень для їх обробки.

2.3.2. Adobe Acrobat OCR

Ще доволі цікавим набір інструментів для розпізнавання зображень є Adobe Acrobat сам по собі це не його основний функціонал і правильно його назвати додатком для роботи з документами у форматі PDF. Сам модуль OCR в даний інструмент дув доданий вже давно ще в середині 90-х років, після чого став стандартом для роботи з PDF документами.

Для роботи OCR в Adobe Acrobat використовуються різні моделі машинного навчання та оптичного розпізнавання тексту, які моментами потребують адаптації для коректної роботи з шрифтами та стилями. Однією з особливістю в Adobe Acrobat є те що OCR фактично інтегрована в додаток, що дозволяє на пряму загружати і працювати з файлами у форматі PDF без необхідності їх експортувати з інших джерел.

Переваги:

1. Підтримка мов та шрифтів.

Хоч і OCR розпізнає меншу кількість мов і форматів ніж інші алгоритми, в любому випадку їх підтримка являється необхідністю для любої подібної системи.

2. Інтеграція з PDF.

Як вже було сказано раніше OCR в даному випадку працює в інструменті для редагування документів що допомагає йому бути ефективним у таких справах де потрібно відскакувати чи відредагувати частину документу.

3. Висока точність.

Adobe це близько не мала інді компанія і вони вкладають великі фінансові ресурси у розвиток своїх технологій і всього що зв’язано з нейронними мережами, що надає OCR у інструменті Acrobat великої точності для зчитування текстів.

Недоліки:

1. Собівартість.

На мою думку у час коли існує велика кількість безкоштовних бібліотек, моделей, алгоритмів та багато інших систем для OCR то платити за використання Acrobat може стати значимим фактором аби обрати іншу програму.

2. Нестабільність роботи з рукописними почерками.

Прочитавши велику кількість відгуків зв'язаних з Acrobat стало зрозуміло, що у дуже багатьох випадках він погано а іноді і дуже відлічав почерки і може путати не лиш букви, але і мови.

3. Невелика кількість форматів.

Ще значний недолік, але з точки зору правильної оцінки його теж треба указати, а саме те, що підтримується не значна кількість форматів для конвертації отриманих результатів.

4. Кількість підтримуваних мов(близько 20 мов).

Мінус геть не значний оскільки всі основні мови є, але все ж таки потрібно на це звернути увагу враховуючи, що тут проходить фактичне порівняння OCR інструментів.

2.3.3. Google ML Kit OCR

Google ML Kit – це доволі обширний інструмент для машинного навчання, який Google надає для розвитку та розробки мобільних додатків, які базуються на Android та iOS. Набір його OCR-інструмент був розроблений спеціально для швидкого і доволі точного розпізнавання текстів на мобільних пристроях, що вдалося зробити за рахунок використання нейронних мереж глибокого навчання.

Дана платформа стала доступною для використання в 2018 році і з того часу активно використовується розробниками програмних продуктів для розробки програм на основі OCR програмних рішень. В більшості випадках людей заволікає те що інструмент доволі новий часто оновлюються, а також має доволі просту систему інтеграції порівняно з іншими програмними рішеннями. Попри невелику кількість підтримуваних мов він лишається доволі потужний інструментом за рахунок інтеграції через SDK.

Переваги:

1.Швидкодія.

Не кожен OCR інструмент може похвастатися такою швидкістю, як ML Kit, якої він має за рахунок наперед навчених моделей.

2. Простота в інтеграції.

Даний інструмент здатний підключатися через надання простих API, що є доволі корисним для початківців, які до кінця не розуміються, як правильно підключати OCR інструменти для своїх проектів

3. Добре розпізнавання шрифтів.

Даний інструмент від попередників доволі непогано розпізнає як і людські рукописні шрифти, так і звичайні друковані, що може бути дуже корисним для програм даного типу.

Недоліки:

1.Проблкми при роботі з великими текстами.

Даний інструмент має певні технічні недоліки через які при стандартних сценаріях експлуатації він не здатний коректно обробляти великі обсяги інформації.

2. Залежність певного функціоналу від з’єднання з інтернетом.

Деяка кількість певних функцій не доступна при стандартному сценарії, оскільки вони потребують використання хмарних обчислень для коректного функціонування.

3. Кількість підтримуваних мов(близько 50 мов).

Мінус геть не значний оскільки всі основні мови є, але все ж таки потрібно на це звернути увагу враховуючи, що тут проходить фактичне порівняння OCR інструментів.

2.3.4. Google Vision API (OCR)

Google Vision API (OCR) – це ще один OCR інструмент від Google для аналізу зображень в тому числі і тексту та роботи з ним. Даний хмарний сервіс є частиною Google Cloud Platform і він надає схожий функціонал, що і попередник, але має свої

переваги та недоліки. Google Vision API був доступний користувачам інтернету, ще в далекому 2016 році і швидко здобув популярність завдяки своїй точності та простоті використання. Саме API використовує працює за рахунок натренованих нейронних мереж та алгоритмам машинного навчання, які Google розробила за немалі фінансові ресурси і підтримує по сьогоднішній день.

Переваги:

1. Висока точність.

За вдяки алгоритмам машинного навчання Google Vision API здатна з великою точністю виявляти мову, шрифти та ще багато інших нюансів зв’язаних з точністю та якістю отриманих результатів.

2. Легкість в інтеграції.

Даний API здатний працювати з більшістю додатків різних виділ за умови підтримки HTTP-запити.

3. Підтримка різних форматів.

API підтримує велику кількість різних форматів зображень та документів для роботи з ними, що є дуже суттєвим плюсом для систем розпізнавання текстів на них.

4. Робота з великими обсягами інформації.

Завдяки своїй архітектурі, яка побудована хмарних носіях Google Vision API здатний обробляти дуже великі обсяги інформації та документів за відносно не великі проміжки часу.

Недоліки:

1.Собівартість.

Так як і з Acrobat у даного API є свій значний мінус, а саме ціна використання сервісами Google для розпізнавання текстів, оскільки для використання API в комерційних цілях буде потребувати підписки до одного з платних тарифів бо кількість безкоштовних запитів дуже обмежена.

2. Залежність від інтернету. Є деяка група людей кому не підвиде варіант використовувати в себе в проекті ПЗ яке потребує постійного інтернет з’єднання для правильної роботи. Причина чому воно не працює без інтернет з’єднання я

думаю всім зрозуміла, а саме використання хмарних технологій для обробки тексту на зображеннях.

3. Неякісні зображення. Даний API, як більшість йому подібний не здатний сам редагувати зображення та забирати з нього шуми чи рішати інші проблеми зв’язані з якістю зображення.

2.3.5. Adobe Scan

Adobe Scan – це ще один продукт від Adobe для сканування документів. Він являє собою мобільний додаток, який використовує свою власну унікальну технологію для розпізнавання документів та перетворення їх у редагуймі файли. В світ даний додаток явився в кінці 2017 року і отримав велику популярність через підтримку та інтеграцію інших продуктів від Adobe таких як Adobe Acrobat, що сильно допомогло не лиш заволікти новачків, але також заставило старих користувачів платформи Adobe застосовувати додаток для облегшення буденних завдань.

Переваги:

1. Інтуїтивний інтерфейс.

Даний продукт не являється окремим продуктом для інтеграції в інші платформи, а являється окремим додатком, що робить зручний інтерфейс в ньому значною перевагою для користувача.

2.Інтеграція та підтримка.

Легко співпрацює з іншими продуктами компанії Adobe та забезпечує, облегшує, доповнює їхні функціональні можливості.

3. Підтримка багатьох форматів.

Як і Adobe Acrobat має підтримку великої кількості форматів в які він здатний конвертувати отриманий текст із зображень.

Недоліки:

1.Підтримка мов(близько 20).

Скоріше всього недолік відносний бо усі основні мови підтримуються, а основна претензія у тому, що дуже багато OCR підтримують по 50 а іноді і більше 100 різних мов, і тут воно сказано лиш для подальшого їх порівняння.

2. Собівартість

Через потреби підписки для отримання повного функціоналу додатку виникає питання чи дійсно потрібний додаток в «екосистемі», де для користування навіть базовим функціоналом якогось з додатків хочуть, аби люди платили, і це при такій різноманітності додатків у App і Play Store, і при їхніх «благих» цінах за свої послуги.

3. Точність в поганих умова.

Одна з проблем додатків такого роду – це погане сприйняття шумів та різних відхилень від нормалі у зображеннях.

2.3.6. Microsoft Azure OCR

Microsoft Azure OCR базується на Microsoft Azure Cognitive Services і є частиною великого хмарного середовища для машинного навчання. Інструмент схожий на аналоги від Google по функціоналу, але має і певні відмінності наприклад як можливість розпізнавати таблиць в документах(з хорошою точністю). Дана платформа OCR використовує велику кількість нейронних мереж для забезпечення точних.

Переваги:

1. Точність і швидкодія.

За рахунок використання в Microsoft Azure глибоких нейронних мереж вона здатна доволі швидко та ефективно розпізнавати текст і зображення.

2. Великі обчислювальні навички.

За рахунок того що Azure знаходяться на хмарах, її алгоритми здатні обчислювати велику кількість даних(зображень чи документів) без виникнення серйозних проблем з працездатністю.

3. Вміння аналізувати структурні дані.

За рахунок своїх алгоритмів Microsoft Azure здатний аналізувати не лише простий текст в документах та зображеннях, але і такі дані, як таблиці та інші структуровані дані.

Недоліки:

1. Залежність від інтернету. Як і більшість схожих інструментів які базуються на хмарній будові для правильної роботи алгоритмів і обробки зображень потрібне постійне і стабільне інтернет з'єднання.

2. Собівартість. Хоч і Microsoft Azure має дуже непоганий функціонал і доволі високу швидкодію, але при використанні його в комерційних цілях виникають певні проблеми, а саме висока собівартість.

3. Підтримка мов(близько 25), форматів та шрифтів.

Із форматів інструмент підтримує усі види фото а також PDF- формат документів, що хватає, але не у всіх випадках. Є певні проблеми при розпізнаванні деяких шрифтів, а також при обробці зображень з рукописним текстом.

4. Погана стійкість до складних умов.

В Microsoft Azure, як і в більшості подібних програм є проблема з розпізнаванням текстів у складних умова, таких, як розмиття, шуми чи інші проблеми.

2.4 Висновок до розділу

В даному розділі була виконана основна задача дослідження, а саме аналіз існуючих рішень, огляд існуючих алгоритмів, моделей для РЗ та вчасності OCR. Були обґрунтовано проаналізовані існуючі моделі та програмні продукти для РЗ, а конкретніше в яких саме задачах і чому саме використовують ту чи іншу модель(алгоритм). Також були розглянуті нейронні мережі які розпізнають ті чи інші особливості зображень і вміють з ними взаємодіяти(аналізувати, класифікувати, або конвертувати в текст). В кінці було зроблене коротке порівняння деяких готових програмних продуктів для виявлення існуючих переваг

та недоліків кожної з них. Ще було зазначені їх основні сфери застосування і було аргументовано чому саме вони використовуються для відповідних задач.

РОЗДІЛ 3 УДОСКОНАЛЕННЯ АЛГОРИТМУ ДЛЯ РОЗПІЗНАВАННЯ ЗОБРАЖЕНЬ ТЕКСТІВ ТА ЙОГО КОНВЕРТАЦІЇ В ТЕКСТ

3.1 Опис проблем при розробці додатків розпізнавання зображень та конвертації їх у текст на мобільних пристроях

Попри сьогоднішній прогрес телефонів багато з них на рівні здатні працювати так, як молодші десктопі рішення вони все ж таки мають ряд проблем для проектів на мобільній платформі та для розробників вчасності.

При розробці мобільного додатку на платформі РЗ потрібно врахувати ряд проблем, які можуть, або виникнуть в процесі розробки, а саме:

1. Обмежені ресурси.

Мобільні пристрої не мають такої кількості обчислювальної здатності, оперативної чи постійної пам’яті, як сервери чи інші дестопні рішення, що є доволі суттєвою проблемою при розробці власного проекту.

2. Погана якість зображення та залежність від якісного апаратного забезпечення.

У більшості випадках камери телефонів мають різні матриці та інші технічні відмінності зв’язані з камерою, що може призвести до показу різних результатів у процесі розпізнавання зображень текстів, в тому числі можуть виникати певні артефакти, що може призвести до критичних помилок.

3.Висока затримка обробки.

При розпізнаванні текстів в реальному часі виникають наступна проблема, а саме те, що не можливо «адекватно» створити повноцінну систему розпізнавання тексту на телефону бо будуть виникати великі затримки при обробці зображень, що в свою чергу може призвести до швидкого виходу з ладу пристрою.

4. Енергоефективність.

Алгоритми РЗ вчасності тексту не дуже енергоефективні, що здатне призвести до постійної залежності від джерела енергії, що теж негативно вплине на роботоздатність гаджету.

5. Інтерфейс.

Інтерфейс користувача повинен бути зручним, інтуїтивно зрозумілим і мати лиш виключно ті функції які потребує розпізнавання текстів, бо нагромаджений інтерфейс здатний призвести до негативного досвіту користування додатком, особливо коли йдеться про мобільні додатки.

6. Сумісність.

Проблема сумісності доволі суттєва, вчасності через те, що на сьогоднішній день телефони доволі швидко міняються і разом з цим зникає підтримка старіших моделей треба впевнитися що все буде працювати і на молодших моделях гаджетів.

7. Інтернет з’єднання.

Інтернет є невідмінною частиною нашого життя і для обчислень і передачі інформації між мобільним пристроєм і хмарними середовищами потрібне постійне інтернет з’єднання.

3.2 Вимоги до програмного рішення РЗ та конвертації їх в текст

В даному пункті будуть розглянуті технічні(функціональні) вимоги до програмного продукту, яким він повинен підходити для того, щоб його назвати вдалим рішенням.

Перелік вимог:

1. Розпізнавання тексту.

Як не дивно найголовнішою вимогою для даної роботи є повноцінна підтримка функціоналу розпізнавання текстів на зображеннях та їх конвертація.

2.Передобробка зображення.

Додати можливість зменшити кількість шумів та інших функцій для удосконалення зображень

3. Післяобробка тексту.

Надати можливість користувачу редагувати текст після його обробки для виправлення малих помилок моделі

4. Збереження та експорт.

Додати можливість зберігати відредагований текст у буфері обміну, а також у вигляді PDF чи DOC форматі для подальшого експорту

5. Швидкодія та продуктивність.

Підібрати найліпшу модель для обробки зображення для найліпшого результату та швидкодії.

6. Інтерфейс користувача.

Розробити зручний інтерфейс бел лишніх функцій для найліпшого та найточнішого розпізнавання зображень.

7. Сумісність.

Надати підтримку старіших версій Android версії наприклад 8.0 і вище.

3.3 Створення алгоритму роботи додатку з упором на поліпшення Google ML Kit OCR

Перед тим як починати розробляти алгоритм бажано придумати назву додатку треба ознайомитися з документацією до Google ML Kit OCR[40], та визначитися з назвою додатку. Для облегшення подальшого оголення додатку, щоб кожен раз не казати цей додаток, даю йому назву «RecQuick», що є скороченням від англійських слів «recognize quickly», які дослівно переводяться «швидке розпізнавання». Далі переходжу на офіційний сайт ML Kit де проходить ознайомлення з усією потрібною інформацією для підключення моделі у додаток.

Після ознайомлення з документацією роботи Google ML Kit та додатковим вивченням усіх плюсів та мінусів додатку, починається процес проектування алгоритму додатку, а саме процесів конвертації зображення в текст.

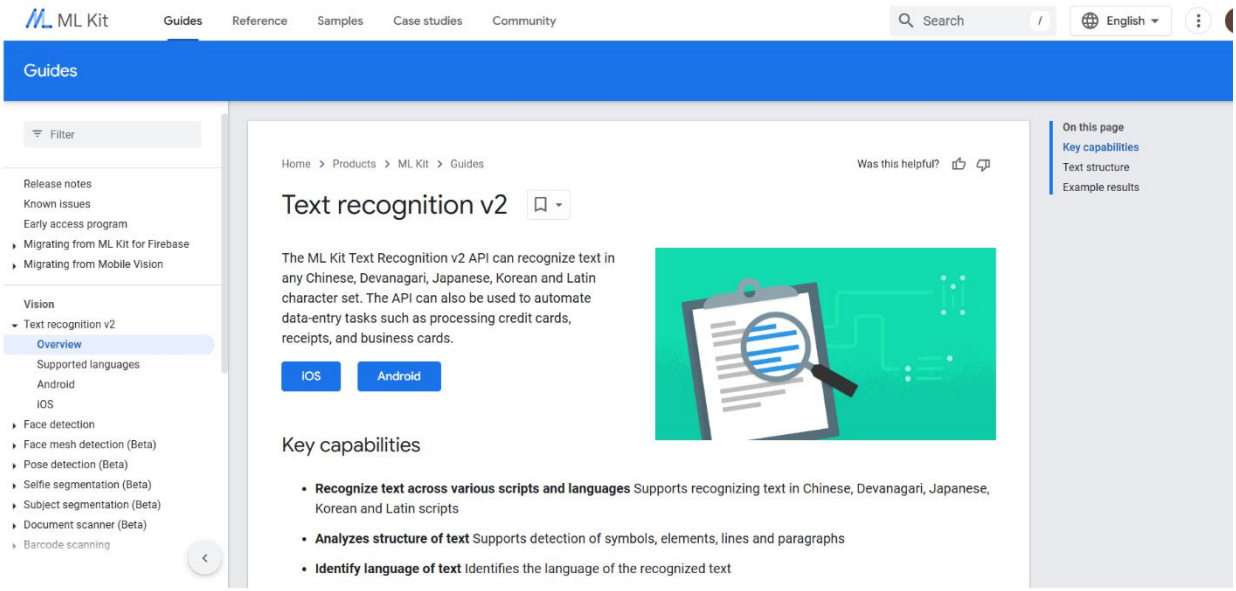

Рис. 3.1. Зображення документації для використання Google ML Kit

1.Користувач заходить в додаток.

Користувач з певною цілю заходить в додаток «RecQuick» для оброблення зображення.

2.Вибір варіанта фото.

В даному етапі користувач вибирає який з варіантів він завантажить фото чи з галереї чи зразу з камери.

3. Попередня обробка.

Використання фільтрів для ліпшого розпізнавання тексту, а саме підвищення різкості чи контрастності для ліпшої видимості тексту на зображені.

4. Обробка зображення Google ML Kit.

Тепер використовується Google ML Kit для розпізнавання тексту на зображення.

5. Оптимізація орфографії.

Тепер виконується процес перевірки написаного тексту. Бо модель не є ідеальною і при розпізнаванні рукописних текстів часто виникають помилки.

6.Перевірка користувачем

Ручна та виправлення помилок користувачем для подальшого використання.

7. Експорт тексту.

Ескорт тексту після обробки користувачем в зручно форматі чи збереження його у буфері обміну.

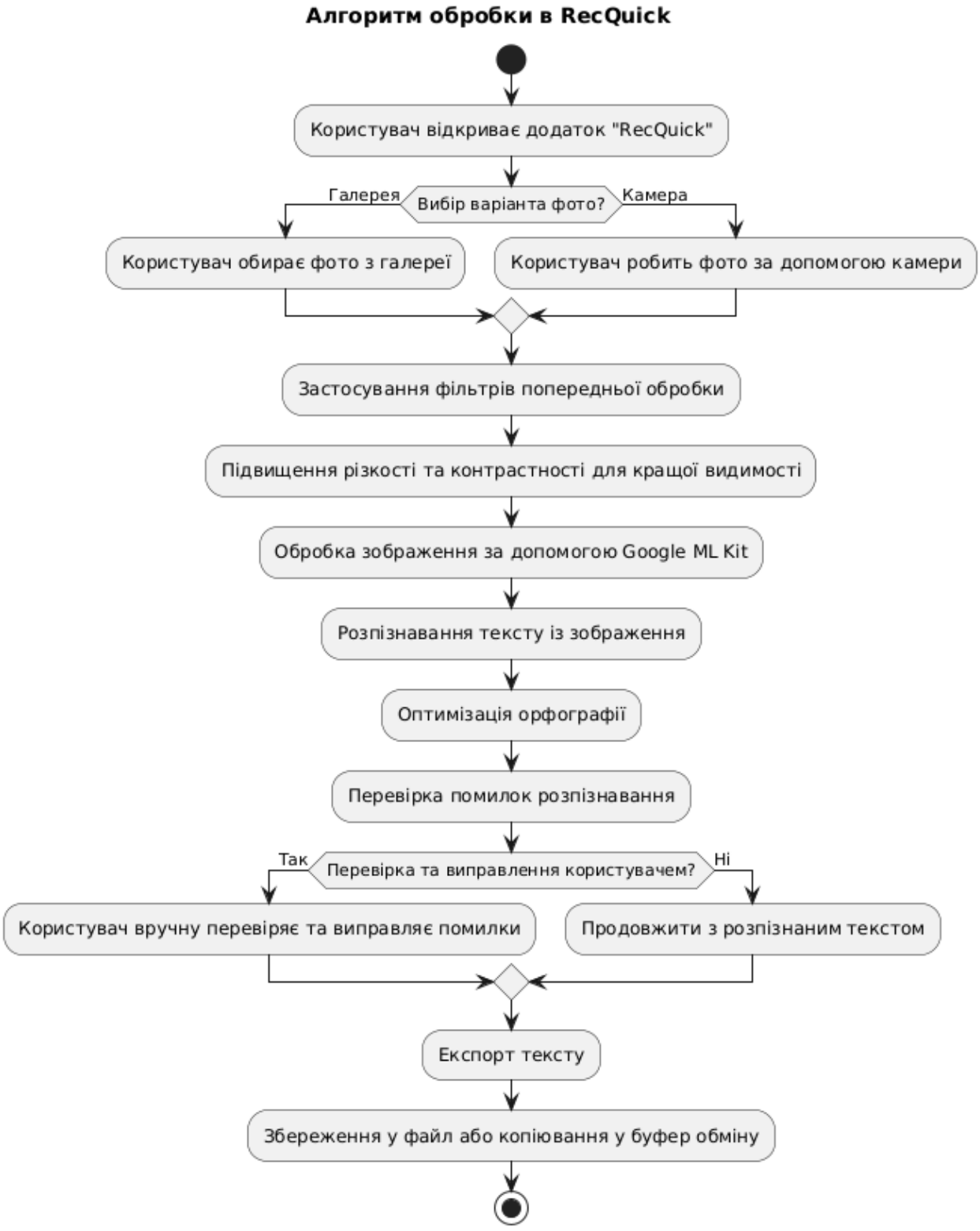

Рис. 3.2. Візуалізація алгоритму роботи додатку RecQuick

3.4 Програмна реалізація функцій додатку РЗ та конвертації його в текст

Для початку створення додатку треба оприділитися з мовою та середовищем розробки. У даному випадку мова для розробки – це Kotlin, а середовищем розробки буде Android Studio.

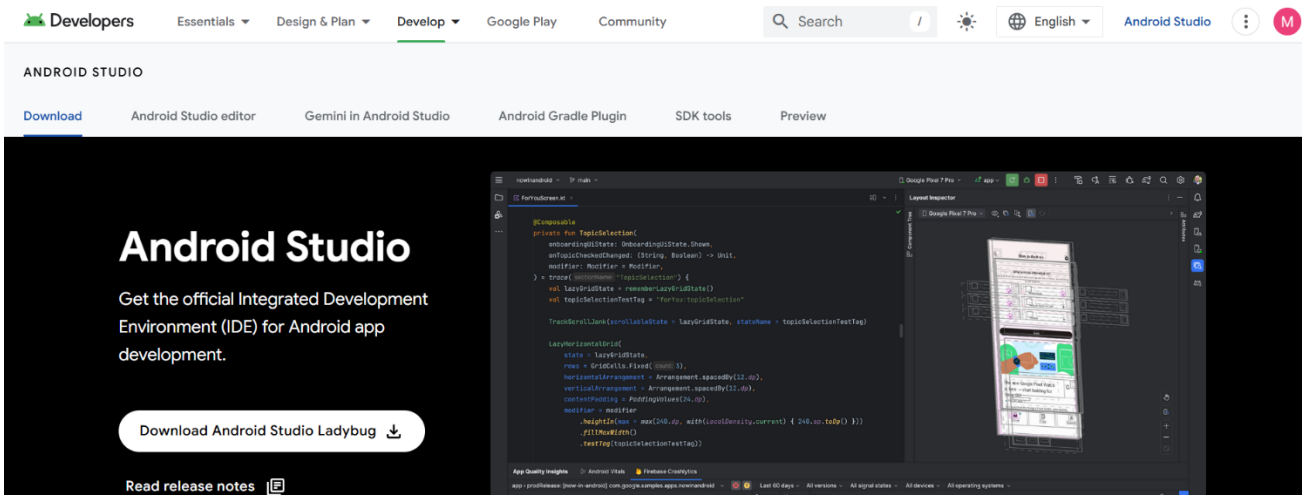

Рис. 3.3. Головне меню сайту Android Studio

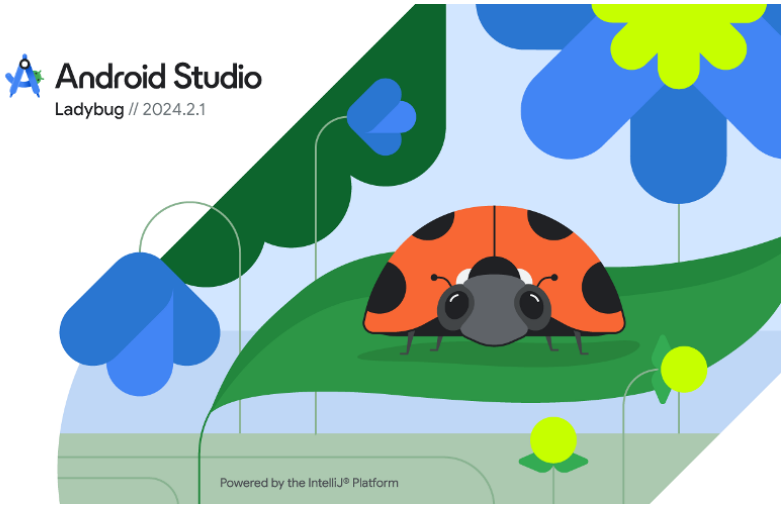

Рис 3.4. Вікно при завантажені Android Studio

Після переходу на сайт скачую середовище і вибираю відповідний макет для створення додатку. В даному етапі запускаються та настроюються усі потрібні

утиліти та бібліотеки для коректної роботи Android Studio та усіх інших потрібних додатків для проекту.

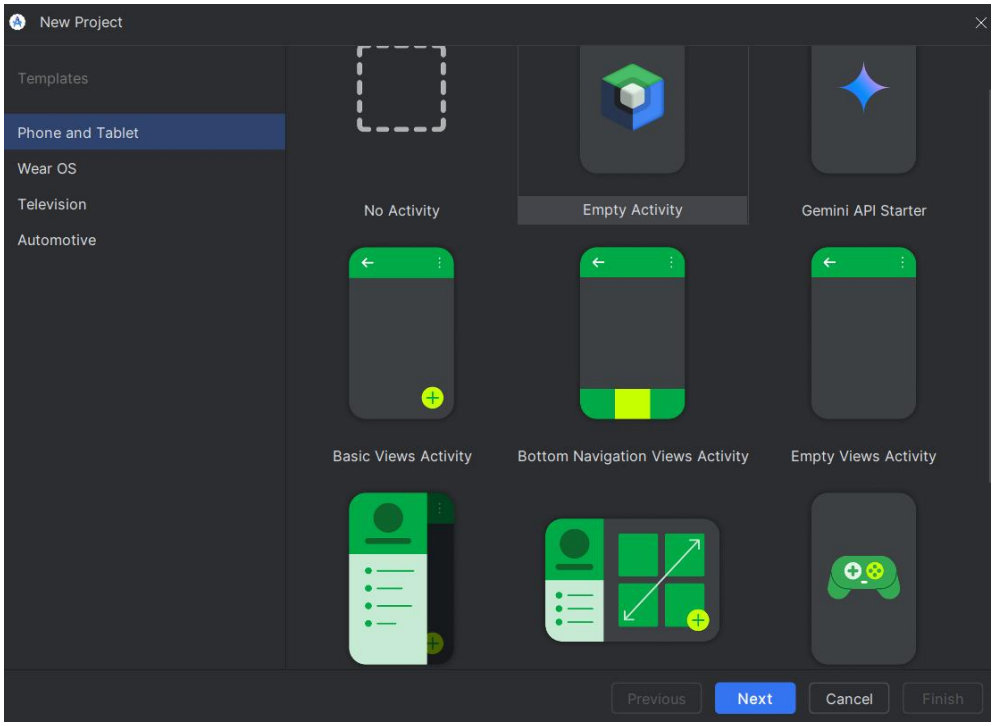

Рис. 3.5. Макети додатків в Android Studio

При створені додатку вибираються оптимальні варіанти відповідно до вимог додатку, а саме підтримка старих версій андроїд.

Для початку створюю інтерфейс користувача для зручного користування додатком та продумую мінімальний, але максимум для зручного користуванням додатку, бо ускладнення інтерфейсу складними анімаціями чи нагромадженим функціоналом може призвести до некоректної роботи на старих на не дуже потужних смартфонах.

У середовищі Android Studio головне меню знаходиться в файлі activity_main.xml та може бути створене, як і з допомогою коду, так і з використанням готових елементів інтерфейсу. В лістингу 3.1 описано процес створення інтерфейсу та зазначено розташування елементів.

Лістинг 3.1.

```
    <RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
```

```
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <!-- Основний вміст сторінки -->
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:orientation="vertical"
        android:padding="16dp"
        android:layout_above="@+id/bottomButtonsLayout">

        <!-- ImageView для відображення вибраного або знятого
зображення -->
        <ImageView
            android:id="@+id/imageView"
            android:layout_width="match_parent"
            android:background="#F0EDED"
            android:layout_height="200dp"
            android:layout_marginBottom="16dp"
            android:contentDescription="Вибране зображення"
            android:scaleType="centerCrop" />

        <!-- Кнопки для вибору фото та роботи з камерою -->
        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:orientation="horizontal"
            android:layout_marginBottom="16dp">

            <Button
                android:id="@+id/btnGallery"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="Вибрати фото з галереї" />

            <Button
                android:id="@+id/btnCamera"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="Відкрити камеру" />…
</LinearLayout>

<!-- EditText для відображення згенерованого тексту -->
<EditText
    android:id="@+id/textViewResult"
    android:layout_width="match_parent"
```

```
 android:layout_height="150dp"
 android:layout_marginTop="16dp"
 android:background="#F0EDED"
 android:padding="8dp"
 android:scrollbars="vertical"
 android:gravity="start|top"
 android:text="Згенерований текст"
 android:overScrollMode="always"
 android:inputType="textMultiLine|textNoSuggestions" />
```

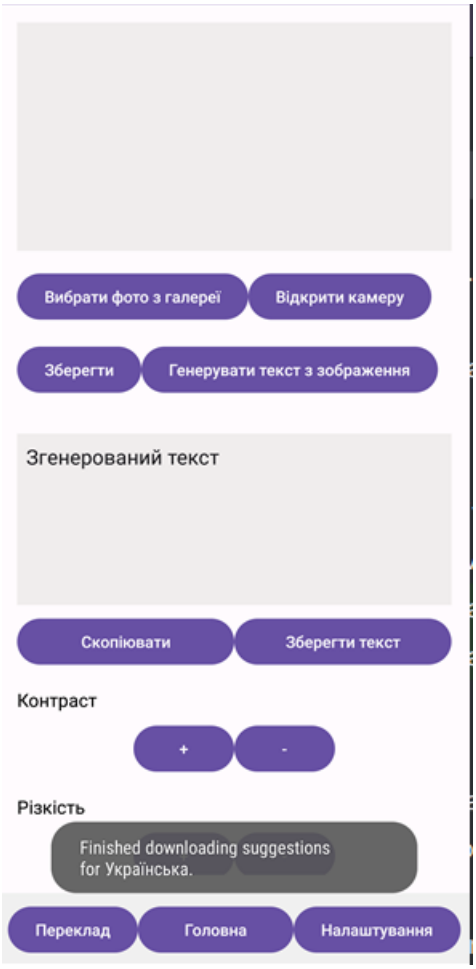

Рис. 3.6. Зовнішній вигляд інтерфейсу користувача

В наступному лістингу 3.2 описані дозволи, які потрібні для роботи з додатком а саме дозвіл до сховищ пристрою та дозвіл до камери. Бо без них не один програмний засіб на телефон не має права втручатися в роботу гаджету без попереднього дозволу від користувача. Що в свою чергу накладає всю відповідальність при використані того чи іншого ПЗ на нього.

Лістинг 3.2

```
private fun checkPermissions() {
if (ContextCompat.checkSelfPermission(this,
Manifest.permission.CAMERA) != PackageManager.PERMISSION_GRANTED) {
ActivityCompat.requestPermissions(this,
arrayOf(Manifest.permission.CAMERA), 0)
}
if (ContextCompat.checkSelfPermission(this,
Manifest.permission.READ_EXTERNAL_STORAGE) !=
PackageManager.PERMISSION_GRANTED) {
ActivityCompat.requestPermissions(this,
arrayOf(Manifest.permission.READ_EXTERNAL_STORAGE), 1)
}
}
```

В лістингу 3.3 описується ініціалізація інтерфейсу та всіх його елементів. Іншими словами виконується процес взаємодії графічної частини додатку з кодом.

Лістинг 3.3

```
// Ініціалізація елементів UI
imageView = findViewById(R.id.imageView)
textViewResult = findViewById(R.id.textViewResult)

// Кнопки для обробки зображень
findViewById<Button>(R.id.btnIncreaseContrast).setOnClickListener {
adjustContrast(1.2f) }
findViewById<Button>(R.id.btnDecreaseContrast).setOnClickListener {
adjustContrast(0.8f)
findViewById<Button>(R.id.btnIncreaseSharpness).setOnClickListener {
adjustSharpness(1.5f) }
findViewById<Button>(R.id.btnDecreaseSharpness).setOnClickListener {
adjustSharpness(0.7f) }


// Кнопки для копіювання та збереження тексту
findViewById<Button>(R.id.btnCopy).setOnClickListener {
copyTextToClipboard() }
findViewById<Button>(R.id.btnSaveText).setOnClickListener {
saveTextToFile() }

// Кнопка для вибору зображення з галереї
findViewById<Button>(R.id.btnGallery).setOnClickListener {
    val galleryIntent = Intent(Intent.ACTION_PICK,
MediaStore.Images.Media.EXTERNAL_CONTENT_URI)
```

```
        startActivityForResult(galleryIntent, GALLERY_REQUEST_CODE)
    }

    // Кнопка для відкриття камери
findViewById<Button>(R.id.btnCamera).setOnClickListener {
        val cameraIntent = Intent(MediaStore.ACTION_IMAGE_CAPTURE)
        startActivityForResult(cameraIntent, CAMERA_REQUEST_CODE)
    }

    // Кнопка для генерування тексту з зображення за допомогою OCR
findViewById<Button>(R.id.btnGenerate).setOnClickListener {
        imageView.drawable?.let { drawable ->
            val bitmap = (drawable as? BitmapDrawable)?.bitmap
            bitmap?.let { processImageForText(it) }
        }
    }
}
```

В лістингу 3.4 показано процес вибору та завантаження зображень з галереї чи камери. У додатку реалізовано лише два способи для завантаження зображень, але їх достатньо для більшої частини користувачів.

Лістинг 3.4

```
    override fun onActivityResult(requestCode: Int, resultCode:
Int, data: Intent?) {
        super.onActivityResult(requestCode, resultCode, data)

        if (resultCode == RESULT_OK) {
            if (requestCode == GALLERY_REQUEST_CODE) {
                val selectedImageUri = data?.data
                imageView.setImageURI(selectedImageUri)
            } else if (requestCode == CAMERA_REQUEST_CODE) {
                val imageBitmap = data?.extras?.get("data") as?
Bitmap
                imageView.setImageBitmap(imageBitmap)
            }
        }
```

Далі для попередньої обробки зображення використовується підвищення фільтрації та контрасті зображенні. Що в подальшому допоможе чіткіше розпізнавати текст на зображенні. Дана реалізація показана в лістингах 3.5(Різкість)

та 3.6(Контраст) і в подальшому покажуть на скільки важлива попередня обробка зображень для таких моделей як Google ML Kit.

Лістинг 3.5

```
private fun adjustSharpness(sharpness: Float) {
if (::currentBitmap.isInitialized) {
val width = currentBitmap.width
val height = currentBitmap.height

val kernel = floatArrayOf(
0f, -sharpness, 0f,
-sharpness, 1 + 4 * sharpness, -sharpness,
0f, -sharpness, 0f
)

val bitmap = Bitmap.createBitmap(width, height,
currentBitmap.config)
val pixels = IntArray(width * height)
val outputPixels = IntArray(width * height)

// Отримуємо пікселі зображення
currentBitmap.getPixels(pixels, 0, width, 0, 0, width, height)

// Проходимося по кожному пікселю
for (y in 1 until height - 1) {
for (x in 1 until width - 1) {
var r = 0f
var g = 0f
var b = 0f

for (ky in -1..1) {
for (kx in -1..1) {
val pixel = pixels[(y + ky) * width + (x + kx)]
val factor = kernel[(ky + 1) * 3 + (kx + 1)]

r += ((pixel shr 16 and 0xFF) * factor)
g += ((pixel shr 8 and 0xFF) * factor)
b += ((pixel and 0xFF) * factor)
}
}

// Обмежуємо значення каналів до 0-255
val red = r.coerceIn(0f, 255f).toInt()
val green = g.coerceIn(0f, 255f).toInt()
val blue = b.coerceIn(0f, 255f).toInt()
```

```
    // Зберігаємо піксель
    outputPixels[y * width + x] = (0xFF shl 24) or (red shl 16) or
(green shl 8) or blue
    }
    }

    // Встановлюємо нові пікселі
    bitmap.setPixels(outputPixels, 0, width, 0, 0, width, height)
    currentBitmap = bitmap

    // Оновлюємо зображення в ImageView
    imageView.setImageBitmap(currentBitmap)
    } else {
    Toast.makeText(this, "Зображення не доступне для обробки",
Toast.LENGTH_SHORT).show()
    }
    }
```

Лістинг 3.6(Зміна контрасту)

```
    private fun adjustContrast(contrast: Float) {
        if (::currentBitmap.isInitialized) {
            val cm = ColorMatrix()
            cm.set(
                floatArrayOf(
                    contrast, 0f, 0f, 0f, 0f,
                    0f, contrast, 0f, 0f, 0f,
                    0f, 0f, contrast, 0f, 0f,
                    0f, 0f, 0f, 1f, 0f
                )
            )

            val paint = Paint().apply { colorFilter =
ColorMatrixColorFilter(cm) }
            val bitmap = Bitmap.createBitmap(currentBitmap.width,
currentBitmap.height, currentBitmap.config)
            val canvas = Canvas(bitmap)
            canvas.drawBitmap(currentBitmap, 0f, 0f, paint)
            currentBitmap = bitmap
            imageView.setImageBitmap(currentBitmap)
        } else {
            Toast.makeText(this, "Зображення не доступне для
обробки", Toast.LENGTH_SHORT).show()
        }
    }
```

Лишається три основних модуля для реалізація, а найголовніший розпізнавання тексту на зображеннях та його компіляція. В наступному лістингу 3.7 буда реалізована логіка розпізнавання тексту з зображення з використанням Google ML Kit.

Лістинг 3.7

```
private fun processImageForText(bitmap: Bitmap) {
    val inputImage = InputImage.fromBitmap(bitmap, 0)
    val recognizer =
TextRecognition.getClient(TextRecognizerOptions.DEFAULT_OPTIONS)

    recognizer.process(inputImage)
        .addOnSuccessListener { visionText ->
displayExtractedText(visionText) }
        .addOnFailureListener { e ->
            Toast.makeText(this, "Не вдалося розпізнати текст:
${e.message}", Toast.LENGTH_SHORT).show()
        }
}

private fun displayExtractedText(visionText: Text) {
    textViewResult.setText(visionText.text)
}
```

Далі буде розглянуто перевірку правопису слів. Для цього в лістингу 3.8 було створе певний алгоритм для перевірки на правопис. Правда він не точний і не годен виправити усі помилки зв’язці з написанням текстів, але основні, такі як великі букви посеред слова він здатний виправити. Тепер замість «ПриВіт» програма пише «Привіт».

Лістинг 3.8

```
private fun autoCorrectText(input: String): String {
// Розбиваємо текст на речення
val sentences = input.split(". ", "! ", "? ")

// Обробляємо кожне речення
val correctedSentences = sentences.map { sentence ->
    sentence.lowercase().replaceFirstChar { it.uppercase() }
}
```

```
    // Об'єднуємо речення назад у текст
    return correctedSentences.joinToString(". ") { it.trim() } + if
(input.endsWith(".") || input.endsWith("!") || input.endsWith("?"))
"" else "."
}
// Виведення розпізнаного тексту та функія автокорекції
private fun displayExtractedText(visionText: Text) {
    val rawText = visionText.text
    val correctedText = autoCorrectText(rawText)
    textViewResult.setText(visionText.text)
}
```

Лишається лише одна функція яку потрібно реалізувати в даному додатку, а саме збереження файлу та швидке його копіювання у буфер обміну для подальшого його використання. Для цього в лістингу 4.9 буде представлена сама проста реалізація даних функцій для найменшого навантаження на телефон.

Лістинг 3.9

```
    // Функція для копіювання тексту в буфер обміну
private fun copyTextToClipboard() {
    val text = textViewResult.text.toString()
    val clipboard = getSystemService(CLIPBOARD_SERVICE) as
ClipboardManager
    val clip = ClipData.newPlainText("Розпізнаний текст", text)
    clipboard.setPrimaryClip(clip)
    Toast.makeText(this, "Текст скопійовано в буфер обміну",
Toast.LENGTH_SHORT).show()
}

// Функція для збереження тексту в .txt файл
private fun saveTextToFile() {
    val text = textViewResult.text.toString()

    // Отримуємо каталог для збереження файлів
    val directory = getExternalFilesDir(null)

    // Знаходимо найбільший номер серед файлів "fototototextX.txt"
    val existingFiles = directory?.listFiles { _, name ->
name.startsWith("fototototext") && name.endsWith(".txt") }
    val nextNumber = if (existingFiles.isNullOrEmpty()) {
        1
    } else {
        existingFiles.mapNotNull {
```

```
it.name.removePrefix("fototototext").removeSuffix(".txt").toIntOrNull(
)
        }.maxOrNull()?.plus(1) ?: 1
    }

    // Формуємо нове ім'я файлу
    val fileName = "fototototext$nextNumber.txt"
    val file = File(directory, fileName)

    // Записуємо текст у файл
    FileOutputStream(file).use {
        it.write(text.toByteArray())
        it.flush()
    }
```

У результаті вийшло створити додаток з відповідним до покладених вимог функціоналом, що має простий але зрозумілий інтерфейс, а також здатний добре працювати на доволі старих девайсах, що буде перевірено в наступному пункті а саме тестування.

3.5 Перевірка на роботоздатності та швидкодію додатку.

Для тестування додатків в Android Studio є можливість створити віртуальну машину для перевірки функціоналу своїх додатків. В якості піддослідного був вибраний наступний тестовий стенд, а саме телефон з середньою діагоналлю та підтримкою Android 8.0.

Рис. 3.7. Тестовий стенд для перевірки додатку

Далі запускається смартфон та додаток на ньому для подальшої перевірки функціоналу. Дана функція є дуже корисною і допомагає перевірити роботоздатність додатку ще до початку реальної експлуатації користувачами

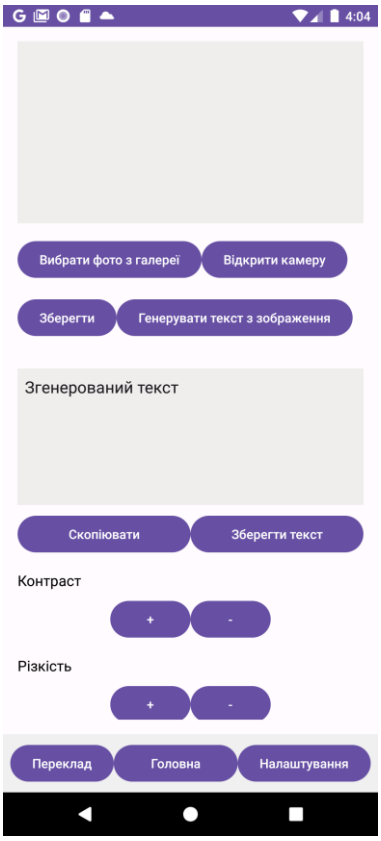

Рис. 3.8. Зовнішній вигляд додатку «RecQuick»

Рис. 3.9 Приклад використання фільтрів в додатку «RecQuick»

Далі перевіряються такі функції, як камера та галерея, які теж успішно працюють. Також були перевірені функції для збільшення контрастності та різкості зображення. Можливо варто трохи переробити алгоритм їх дії для більшого ефекту, але вони доволі якісно справляються і без цього. Загалом дані функції сильно допомагають при розпізнаванні складних текстів рукописного походження, або фотографій, які були зроблені у не дуже добру погоду чи при несприятливих умовах таких як темрява.

3.6 Висновок до розділу

В даному розділі було проведено аналізу найважливіших проблем для розробки систем РЗ на мобільних платформах, а також була представлений один з алгоритмів роботи одного з таких додатків в якому було представлено взаємодії користувача з компонентами у системі. Ще було реалізовано програмне рішення для розпізнавання текстів на зображеннях. В результаті був отриманий додаток м мінімально потрібним функціоналом для розпізнавання зображень, при цьому за рахунок того, що більшість обчислень проходять на хмарному сховищі додаток здатний адекватно працювати і коректно виконувати свої функції на доволі слабких телефонах системах наприклад Android 8.0 чи старших. Переваг у додатку наступні – це швидкодія, якість результатів, можливість їх збереження. Недоліки наступні: відсутність анімацій інтерфейсу(не можливо реалізувати для плавної роботи на старих девайсах), до кінця правильне розпізнавання кирилиці час від часу виникають проблеми з неправильним визначенням літер, наприклад може спутати «3» та цифру «три».

ВИСНОВОК

В даній магістерській роботі була проведена робота з вивченням, аналізом існуючих програмних рішень, алгоритмів та моделей для розпізнавання зображень. Були вивчені такі поняття, як комп'ютерний зір його аспекти, методи та принципи роботи. Було коротко розглянуто історію технології розпізнавання зображень її основні моменти періоду, як вона розвинулася з забавки для вчених до повноцінної технології, яка інтегрована у багатьох аспектах теперішнього життя.

Також у якості прикладу використання технології було детально розглянуто різні існуючі програмні продукти, алгоритми та моделі, як класичні, так і основані на ШІ та машинному навчанні.

В якості практичної частини для магістерської роботи було реалізований додаток, який оснований на використанні ML Kit для розпізнавання тексту. В ньому були реалізовані основні, аспекти для поліпшення результату розпізнавання тексту, а також були зазначені проблеми моделі ML Kit для розпізнавання різних мов, що ще раз підкреслює, що попри те що технологія РЗ є дуже корисною і широко використовується її досі є куди розвивати, бо в даний момент виникає дилема між якістю отриманих результатів, часом потрібних для їх обчислення та кількістю обчислювальних ресурсів для отримання оптимального результату. І на мою думку, як автора даної магістерської дана технологія ще має дуже великі перспективи розвитку у вчасності за рахунок швидкого оновлення нейронних моделей та усіх технологій зв'язаних з ШІ.

Правда окрім розвитку загальних аспектів для удосконалення програмного забезпечення та систем зв'язаних з розпізнаванням зображень варто не забувати про розвиток в таких галузях, що зв'язані з виробництвом графічних прискорювачів та центральних процесорів. Основний приріст на сьогоднішній день в продуктивності процесорів та графічних прискорювачів заключається у стоншені техпроцесу. Але дана «гонка» за легкою продуктивністю може дуже скоро зайти в глухий кут через те, що кремній вже наблизився до товщини 3 нм., і з кожним роком його стоншення і удосконалення стає все дорожчим і нестабільним. Так, що для

подальшого розвитку технології РЗ все ж таки потрібно, або знаходити нову альтернативу кремнію, або оптимізувати та удосконалювати ті технології, які допоможуть оптимально і економна використовувати ресурси сучасних і майбутніх комп’ютерних систем.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Берман М., Сапіро Г. Математичні методи в реконструкції зображень. – SIAM, 2017. – 450 с.
2. Бішоп К.М. Розпізнавання патернів і машинне навчання. – К.: Springer, 2006. – 738 с.
3. Ван Чж., Ма Л. Сегментація зображень: огляд. – Journal of Computer Science and Technology, 2016. – 12 с.
4. Гонсалес Р.С., Вудс Р.Е. Цифрова обробка зображень. – К.: Pearson Prentice Hall, 2008. – 1036 с.
5. Джейн А.К., Фаррохнія Ф. Несупервізоване сегментування текстур з використанням фільтрів Габора. – Pattern Recognition, 1991. – 6 с.
6. Джейдерберг М., Симонян К., Зіссерман А., Кавукчуоглу К. Просторові трансформери мережі. – NeurIPS, 2015. – 8 с.
7. Дапогні А., Ліндеберг Т. Теорія просторових масштабів в комп'ютерному зорі. – Springer, 2002. – 450 с.
8. Далал Н., Тріггс Б. Гістограми орієнтованих градієнтів для виявлення людей. – IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2005. – 7 с.
9. Зейлер М.Д., Фергюс Р. Візуалізація та розуміння згорткових мереж. – Європейська конференція з комп'ютерного зору (ECCV), 2014. – 10 с.
10. Кастл А., Мур Д. Посібник з обробки зображень. – К.: CRC Press, 2011. – 684 с.
11. Кохлі П., Ротер К. Текстони, контури та області: основа для низькорівневого зору. – IEEE Transactions on Pattern Analysis and Machine Intelligence, 2009. – 15 с.
12. Кріжевський А., Суцкевер І., Хінтон Г.Е. Класифікація ImageNet за допомогою глибоких згорткових нейронних мереж. – NeurIPS, 2012. – 10 с.
13. ЛеКун Й., Бенджіо Й., Хінтон Г. Глибоке навчання. – Nature, 2015. – 436 с.
14. Лекун Й., Бенджіо Й., Хінтон Г. Глибоке навчання. – К.: MIT Press, 2016. – 800 с.
15. Лін М., Чен Ц., Ян С. Мережі в мережах. – arXiv, 2014. – 6 с.
16. Міколов Т., Чен К., Коррадо Г., Дін Дж. Ефективна оцінка представлень слів у векторному просторі. – arXiv, 2013. – 8 с.

17. О'Рурк Дж. Обчислювальна геометрія в С. – К.: Cambridge University Press, 1998. – 580 с.
18. Поггіо Т., Брадскі Г. Алгоритми та застосування комп'ютерного зору. – К.: Springer, 2010. – 652 с.
19. Рассел С., Норвіг П. Штучний інтелект: сучасний підхід. – К.: Pearson, 2016. – 1136 с.
20. Різенгубер М., Поггіо Т. Ієрархічні моделі розпізнавання об'єктів у корі. – Nature Neuroscience, 1999. – 15 с.
21. Роннебергер О., Фішер П., Брокс Т. U-Net: Згорткові мережі для сегментації біомедичних зображень. – MICCAI, 2015. – 8 с.
22. Саттон Р.С., Барто А.Г. Навчання з підкріпленням: введення. – К.: MIT Press, 1998. – 552 с.
23. Сі Дж., Міан А. Згорткові нейронні мережі для візуального розпізнавання. – Springer, 2018. – 280 с.
24. Симонян К., Зіссерман А. Дуже глибокі згорткові мережі для масштабного розпізнавання зображень. – arXiv, 2014. – 8 с.
25. Сонка М., Хлавак В., Бойл Р. Обробка зображень, аналіз та машинне бачення. – К.: Cengage Learning, 2014. – 944 с.
26. Сжеліскі Р. Алгоритми та застосування комп'ютерного зору. – К.: Springer, 2010. – 652 с.
27. Фергус Р., Перона П., Зіссерман А. Розпізнавання об'єктів за допомогою багатошкального навчального алгоритму. – IEEE Transactions on Pattern Analysis and Machine Intelligence, 2003. – 8 с.
28. Фукушима К. Неокогнітрон: самонавчальна нейронна мережа для механізму розпізнавання патернів, нечутливого до зміщення позиції. – Biological Cybernetics, 1980. – 10 с.
29. Хан С., Анвар С. Алгоритми глибокого навчання для розпізнавання та класифікації зображень. – Springer, 2020. – 410 с.
30. Хан С., Хаяд М. Методи розпізнавання зображень: алгоритми та застосування. – Springer, 2021. – 450 с.

31. Харалік Р.М., Шанмугам К., Дінштейн І. Текстурні ознаки для класифікації зображень. – IEEE Transactions on Systems, Man, and Cybernetics, 1973. – 6 с.
32. Чен І., Сун Х. Огляд сегментації зображень: алгоритми та застосування. – Springer, 2016. – 290 с.
33. Чжоу Б., Хосла А. Навчання глибоких ознак для дискримінативної локалізації. – IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015. – 7 с.
34. Шапіро Л.Г., Стокман Дж.Т. Комп'ютерний зір. – К.: Prentice Hall, 2001. – 552 с.
35. Янг Л., Шао М., Юн Л. Глибоке навчання для розпізнавання зображень: нові методи та алгоритми. – IEEE Transactions on Pattern Analysis and Machine Intelligence, 2019. – 10 с.
36. Лі М., Зу Л., Лі П. Моделі згорткових нейронних мереж для розпізнавання зображень у реальному часі. – Journal of Machine Learning Research, 2018. – 12 с.
37. Лін Ц., Ву Л., Хо Дж. Алгоритми для класифікації зображень на основі глибокого навчання. – IEEE Transactions on Image Processing, 2020. – 8 с.
38. Сун С., Чжан Х., Лі Ш., Цай С. Алгоритми для високопродуктивного розпізнавання зображень. – Journal of Artificial Intelligence Research, 2022. – 12 с.
39. Тань Я., Лі Л., Ванг С. Комп'ютерне розпізнавання та його застосування до візуальних даних. – Springer, 2019. – 300 с.
40. Google ML Kit Електронний ресурс . - Режим доступу: https://developers.google.com/ml-kit/vision/text-recognition/v2/android?hl=ru

ДОДАТКИ

Додаток А

Частина основного коду додатку

```
package com.example.myapp

import android.Manifest
import android.content.ClipData
import android.content.ClipboardManager
import android.content.Intent
import android.content.pm.PackageManager
import android.graphics.Bitmap
import android.graphics.Canvas
import android.graphics.ColorMatrix
import android.graphics.ColorMatrixColorFilter
import android.graphics.Paint
import android.graphics.drawable.BitmapDrawable
import android.os.Bundle
import android.os.Parcel
import android.os.Parcelable
import android.provider.MediaStore
import android.widget.Button
import android.widget.EditText
import android.widget.ImageView
import android.widget.Toast
import androidx.appcompat.app.AppCompatActivity
import androidx.core.app.ActivityCompat
import androidx.core.content.ContextCompat
import com.example.myapplication12.R
import com.google.mlkit.vision.common.InputImage
import com.google.mlkit.vision.text.Text
```

```
import com.google.mlkit.vision.text.TextRecognition
import
com.google.mlkit.vision.text.latin.TextRecognizerOptions
import java.io.File
import java.io.FileOutputStream

class MainActivity() : AppCompatActivity(), Parcelable {

    private val GALLERY_REQUEST_CODE = 100
    private val CAMERA_REQUEST_CODE = 101

    private lateinit var imageView: ImageView
    private lateinit var textViewResult: EditText
    private lateinit var currentBitmap: Bitmap

    constructor(parcel: Parcel) : this() {}

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)

        // Ініціалізація елементів UI
        imageView = findViewById(R.id.imageView)
        textViewResult = findViewById(R.id.textViewResult)

        // Кнопки для обробки зображень

findViewById<Button>(R.id.btnIncreaseContrast).setOnClickLi
stener { adjustContrast(1.2f) }
```

```
findViewById<Button>(R.id.btnDecreaseContrast).setOnClickLi
stener { adjustContrast(0.8f) }

findViewById<Button>(R.id.btnIncreaseSharpness).setOnClickL
istener { adjustSharpness(1.5f) }

findViewById<Button>(R.id.btnDecreaseSharpness).setOnClickL
istener { adjustSharpness(0.7f) }

        // Кнопки для копіювання та збереження тексту

findViewById<Button>(R.id.btnCopy).setOnClickListener {
copyTextToClipboard() }

findViewById<Button>(R.id.btnSaveText).setOnClickListener {
saveTextToFile() }

        // Кнопка для вибору зображення з галереї

findViewById<Button>(R.id.btnGallery).setOnClickListener {
            val galleryIntent = Intent(Intent.ACTION_PICK,
MediaStore.Images.Media.EXTERNAL_CONTENT_URI)
            startActivityForResult(galleryIntent,
GALLERY_REQUEST_CODE)
        }

        // Кнопка для відкриття камери

findViewById<Button>(R.id.btnCamera).setOnClickListener {
            val cameraIntent =
```

```
Intent(MediaStore.ACTION_IMAGE_CAPTURE)
            startActivityForResult(cameraIntent,
CAMERA_REQUEST_CODE)
        }

        // Кнопка для генерування тексту з зображення за
допомогою OCR

findViewById<Button>(R.id.btnGenerate).setOnClickListener {
            imageView.drawable?.let { drawable ->
                val bitmap = (drawable as?
BitmapDrawable)?.bitmap
                bitmap?.let { processImageForText(it) }
            }
        }
    }

    // Функція для налаштування контрасту
    private fun adjustContrast(contrast: Float) {
        if (::currentBitmap.isInitialized) {
            val cm = ColorMatrix()
            cm.set(arrayOf(
                contrast, 0f, 0f, 0f, 0f,
                0f, contrast, 0f, 0f, 0f,
                0f, 0f, contrast, 0f, 0f,
                0f, 0f, 0f, 1f, 0f
            ).toFloatArray())

            val paint = Paint()
            paint.colorFilter = ColorMatrixColorFilter(cm)
```

```
            // Створюємо новий Bitmap для змін
            val bitmap =
Bitmap.createBitmap(currentBitmap.width,
currentBitmap.height, currentBitmap.config)
            val canvas = android.graphics.Canvas(bitmap)
            canvas.drawBitmap(currentBitmap, 0f, 0f, paint)
            currentBitmap = bitmap

            // Оновлюємо зображення в ImageView
            imageView.setImageBitmap(currentBitmap)
        } else {
            Toast.makeText(this, "Зображення не доступне
для обробки", Toast.LENGTH_SHORT).show()
        }
    }


    // Функція для зміни різкості (не реалізована)
    private fun adjustSharpness(sharpness: Float) {
        // Implement a sharpening filter logic here
        Toast.makeText(this, "Різкість змінена на
$sharpness", Toast.LENGTH_SHORT).show()
    }

    // Функція для копіювання тексту в буфер обміну
    private fun copyTextToClipboard() {
        val text = textViewResult.text.toString()
        val clipboard = getSystemService(CLIPBOARD_SERVICE)
as ClipboardManager
```

```
        val clip = ClipData.newPlainText("Розпізнаний
текст", text)
        clipboard.setPrimaryClip(clip)
        Toast.makeText(this, "Текст скопійовано в буфер
обміну", Toast.LENGTH_SHORT).show()
    }

    // Функція для збереження тексту в .txt файл
    private fun saveTextToFile() {
        val text = textViewResult.text.toString()

        // Отримуємо каталог для збереження файлів
        val directory = getExternalFilesDir(null)

        // Знаходимо найбільший номер серед файлів
"fototototextX.txt"
        val existingFiles = directory?.listFiles { _, name
-> name.startsWith("fototototext") && name.endsWith(".txt") }
        val nextNumber = if (existingFiles.isNullOrEmpty())
{
            1
        } else {
            existingFiles.mapNotNull {

it.name.removePrefix("fototototext").removeSuffix(".txt").toI
ntOrNull()
            }.maxOrNull()?.plus(1) ?: 1
        }

        // Формуємо нове ім'я файлу
```

```
        val fileName = "fototototext$nextNumber.txt"
        val file = File(directory, fileName)

        // Записуємо текст у файл
        FileOutputStream(file).use {
            it.write(text.toByteArray())
            it.flush()
        }

        // Показуємо повідомлення користувачу
        Toast.makeText(this, "Текст збережено у файл
$fileName", Toast.LENGTH_SHORT).show()
    }
```