

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет комп'ютерних наук і технологій

(повне найменування факультету)

Кафедра програмних засобів

(повне найменування кафедри)

Пояснювальна записка

до дипломного проєкту (роботи)

магістр

(ступінь вищої освіти)

на тему ДОСЛІДЖЕННЯ ТА ПРОГРАМНА РЕАЛІЗАЦІЯ
НЕЙРОМЕРЕЖЕВИХ ЗАСОБІВ РОЗПІЗНАВАННЯ ОБ'ЄКТІВ
RESEARCH AND SOFTWARE IMPLEMENTATION OF
NEURAL NETWORK TOOLS FOR OBJECT RECOGNITION

Виконав(ла): студент(ка) 2 курсу, групи КНТ-213М
Спеціальності 122 Комп'ютерні науки
(код і найменування спеціальності)

Освітня програма (спеціалізація)
Системи штучного інтелекту

КОЛБАСА Д.О.

(ПРИЗВИЩЕ та ініціали)

Керівник ОЛІЙНИК А.О.

(ПРИЗВИЩЕ та ініціали)

Рецензент ІЛЛЯШЕНКО М.Б.

(ПРИЗВИЩЕ та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет КНТ
Кафедра програмних засобів
Ступінь вищої освіти магістр
Спеціальність 122 Комп'ютерні науки
(код і найменування)

Освітня програма (спеціалізація) Системи штучного інтелекту
(назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮ

Завідувач кафедри ПЗ, д.т.н, проф.
Сергій СУББОТІН
“ ” 2024 року

З А В Д А Н Н Я

НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА(КИ)

КОЛБАСИ Дмитра Олександровича

(ПРИЗВИЩЕ, ім'я, по батькові)

1. Тема проєкту (роботи) Дослідження та програмна реалізація нейромережевих засобів розпізнавання об'єктів. Research and Software Implementation of Neural Network Tools for Object Recognition

керівник проєкту (роботи) д.т.н., професор, ОЛІЙНИК Андрій Олександрович,
(науковий ступінь, вчене звання, ПРИЗВИЩЕ, ім'я, по батькові)

затверджені наказом закладу вищої освіти від “07” жовтня 2024 року № 398

2. Строк подання студентом проєкту (роботи) 02 грудня 2024 року

3. Вихідні дані до проєкту (роботи) рекомендована література, технічне завдання

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1. Аналіз предметної області. 2. Матеріали і методи. 3. Опис програми. 4. Експлуатація, тестування та експериментальне дослідження програми.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, кількість слайдів, плакатів)

Слайди презентації

6. Консультанти розділів проєкту (роботи)

Розділ	ПРИЗВИЩЕ, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-4 Основна частина	ОЛІЙНИК А.О., професор		
Нормоконтроль	ДЕЙНЕГА Л.Ю., ст. викладач		

7. Дата видачі завдання “9” вересня 2024 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
1	Постановка завдання роботи.	1 тиждень	Завдання, ТЗ
2	Аналіз предметної області.	1 тиждень	Розділ 1
3	Вибір мови програмування та інших технологій розробки.	2 тиждень	Розділ 2
4	Розробка структури програми.	3 тиждень	Розділ 3
5	Розробка програми.	4-7 тижні	Розділи 3,4
6	Тестування та експериментальне дослідження програмного забезпечення.	8 тиждень	Розділ 4
7	Оформлення пояснювальної записки та документів до неї.	9-10 тижні	Додатки
8	Нормоконтроль та рецензування.	11 тиждень	
9	Захист роботи.	12 тиждень	

Студент(ка)

_____ Дмитро КОЛБАСА
(підпис) (Ім'я ПРИЗВИЩЕ)

Керівник проєкту (роботи)

_____ Андрій ОЛІЙНИК
(підпис) (Ім'я ПРИЗВИЩЕ)

РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи магістра:
84с., 3 табл., 28 рис., 3 дод., 27 джерел.

PYTHON, TENSORFLOW, МОВА ПРОГРАМУВАННЯ, НЕЙРОННА
МЕРЕЖА, ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ, РОЗПІЗНАВАННЯ ОБ'ЄКТІВ.

Об'єкт дослідження – процес розробки програмного забезпечення для розпізнавання об'єктів.

Предмет дослідження – програмні засоби для підтримки розпізнавання об'єктів.

Мета роботи – дослідження та розробка програмного забезпечення для розпізнавання об'єктів на основі нейромережових технологій.

Матеріали, методи та технічні засоби: мова програмування Python, середовище розробки Sublime Text, бібліотека Tensorflow.

Результати. Розроблено програмне забезпечення для розпізнавання об'єктів на основі нейромережових технологій за допомогою мови програмування Python та середовища розробки Sublime Text.

Практична цінність роботи полягає у розробці програмного забезпечення, що забезпечує розпізнавання об'єктів на основі нейромережових технологій у.

Висновки. Виконано проєктування програмного забезпечення для розпізнавання об'єктів на основі нейромережових технологій. Розроблено програмне забезпечення для розпізнавання об'єктів на основі нейромережових технологій. Здійснено тестування розробленого програмного забезпечення для розпізнавання об'єктів на основі нейромережових технологій.

Галузь використання – розпізнавання образів.

ABSTRACT

Explanatory note to the diploma qualifying work of the master: 84 pages, 3 tables, 28 figures, 3 appendixes, 27 sources.

PYTHON, TENSORFLOW, PROGRAMMING LANGUAGE, NEURAL NETWORK, SOFTWARE, OBJECT RECOGNITION.

The object of research is the process of developing software for object recognition.

The subject of the research is software for supporting the object recognition process.

The purpose of this work is to research and develop the software for object recognition based on neural network technologies.

Materials, methods and technical tools: Python programming language, Sublime Text development environment, Tensorflow library.

Results. The software tools to support the object recognition based on neural network technologies using the Python programming language and the Sublime Text development environment have been developed.

The practical value of the work lies in the development of software that provides object recognition based on neural network technologies.

Conclusions. The software to support the object recognition process has been designed. The software to support the object recognition process has been developed. The developed software to support the object recognition process has been tested.

Scope of use – pattern recognition.

ЗМІСТ

	С.
Перелік скорочень та умовних позначок	8
Вступ.....	9
1 Аналіз предметної області.....	11
1.1 Системи розпізнавання образів	11
1.2 Аналіз програмного забезпечення для підтримки процесу розпізнавання об'єктів.....	21
1.3 Висновки за розділом 1	30
2 Матеріали і методи	32
2.1 Вибір мови програмування	32
2.2 Вибір середовища розробки для створення програмного забезпечення для розпізнавання об'єктів	34
2.3 Розробка моделі взаємодії користувача з програмним забезпеченням для розпізнавання об'єктів	36
2.4 Висновки за розділом 2	38
3 Опис програми	39
3.1 Структура програмного забезпечення для розпізнавання об'єктів.....	39
3.2 Функціонування програмного забезпечення для розпізнавання об'єктів.....	40
3.3 Підготовка вхідних даних для програмного забезпечення для розпізнавання об'єктів	42
3.4 Опис процесу синтезу розпізнавальних моделей при створенні програми для розпізнавання об'єктів	45
3.5 Висновки за розділом 3	48
4 Експлуатація, тестування та експериментальне дослідження програми.....	50
4.1 Призначення й умови застосування програми	50
4.2 Характеристики програми для розпізнавання об'єктів	51
4.3 Інструкція по експлуатації програми.....	51
4.3.1 Звернення до програми.....	51
4.3.2 Вхідні й вихідні дані	52

4.3.3 Повідомлення.....	52
4.4 Виконання програмного забезпечення для підтримки процесу розпізнавання об'єктів	52
4.5 Тестування програмного забезпечення для підтримки процесу розпізнавання об'єктів	55
4.6 Висновки за розділом 4	56
Висновки	57
Перелік джерел посилання	60
Додаток А Технічне завдання	63
Додаток Б Фрагмент тексту програми	67
Додаток В Слайди презентації	77

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

НМ – нейронна мережа;

РО – розпізнавання об'єктів;

ПЗ – програмне забезпечення;

ШІ – штучний інтелект;

ШНМ – штучна нейронна мережа.

ВСТУП

Актуальність програмного забезпечення для розпізнавання образів надзвичайно висока в сучасному середовищі. Програмні засоби для розпізнавання об'єктів використовуються в системах відеоспостереження для автоматичного виявлення ідентифікації людей, транспортних засобів та інших об'єктів, що підвищує рівень безпеки [1]-[6].

У промисловості програмні засоби розпізнавання об'єктів допомагає автоматизувати процеси контролю якості, упаковки та управління складськими запасами. У медицині методи та технології розпізнавання об'єктів допомагають в аналізі медичних зображень, таких як рентгенівські знімки, для діагностики захворювань та відстеження прогресу лікування. Програмні засоби розпізнавання об'єктів дозволяють покращити обслуговування клієнтів, автоматизувати інвентаризацію і впроваджувати нові технології для покупок без каси [1]-[6].

У транспорті та робототехніці програмні засоби розпізнавання об'єктів дозволяють створювати автономні транспортні засоби, які можуть орієнтуватися в навколишньому середовищі, уникати перешкод і взаємодіяти з об'єктами [1]-[6].

Проте деякі програмні засоби для розпізнавання об'єктів можуть неправильно ідентифікувати об'єкти. Це може статися через використання неефективних алгоритмів розпізнавання образів, або обмеження у використовуваних алгоритмах, недостатню якість зображення або складність середовища. Інші програмні засоби для розпізнавання об'єктів можуть вимагати значної обчислювальної потужності і великої кількості пам'яті, що може бути витратним і обмежити використання в деяких умовах. Впровадження та підтримка інших програмних систем розпізнавання об'єктів можуть бути дорогими, особливо для малих і середніх підприємств.

Тому актуальною є розробка програмного забезпечення для розпізнавання об'єктів. У дипломній кваліфікаційній роботі магістра

розв'язується актуальне завдання дослідження та розробки програмного забезпечення для розпізнавання об'єктів на основі нейромережових технологій.

Для досягнення поставленої мети у кваліфікаційній роботі магістра необхідно розв'язати такі задачі:

- виконати аналіз предметної області та програмних засобів для розпізнавання об'єктів;
- розробити модель взаємодії користувача з програмним забезпеченням для розпізнавання об'єктів на основі нейромережових технологій;
- здійснити проектування програмного забезпечення для розпізнавання об'єктів на основі нейромережових технологій;
- створити програмне забезпечення для розпізнавання об'єктів на основі нейромережових технологій;
- виконати тестування розробленого програмного забезпечення для розпізнавання об'єктів на основі нейромережових технологій.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Системи розпізнавання образів

Розпізнавання об'єктів (РО) – це процес, за допомогою якого програмне забезпечення виявляє шаблони в даних і використовує їх для прогнозування або класифікації за допомогою обчислювальних алгоритмів. Це важлива складова сучасних систем штучного інтелекту (ШІ) [1]-[7].

Процес розпізнавання об'єктів включає аналіз вхідних даних для виявлення певних шаблонів, які можуть бути використані для прогнозування, сегментації даних і прийняття обґрунтованих рішень [1]-[7].

Хоча розпізнавання об'єктів часто зосереджується на виявленні шаблонів у даних, наука про розпізнавання шаблонів починається з класифікації невизначених шаблонів. Це забезпечує можливість як для розпізнавання образів, так і для інших методів, що базуються на природних закономірностях і їх розвитку [1]-[7].

Таким чином, розпізнавання об'єктів є частиною більш широкого набору дисциплін, які охоплюють різні аспекти науки і техніки. Для створення інтелектуальних систем часто необхідні можливості для ефективного визначення і класифікації шаблонів [1]-[7].

Системи розпізнавання образів використовують комп'ютерні алгоритми для аналізу та інтерпретації різних типів даних. Вхідні дані можуть бути представлені у формі тексту, зображень або звуків. Це схоже на розвинений комп'ютерний зір, який фокусується на розпізнаванні образів [1]-[7].

Автоматизація, машинне навчання шаблонів, прогнозування, класифікація та інтеграція є важливими аспектами в техніці і науці, включаючи біологію, психологію, медицину, маркетинг, комп'ютерний зір і штучний інтелект [1]-[7].

Шаблони обробки можуть повторюватися в різних типах даних. Наприклад, зображенням може бути відбиток пальця, підпис, обличчя чи

запис голосу. Моделі можуть бути спостережені фізично, через зображення та відео, або математично, використовуючи геометричні формули [1]-[7].

Методи розпізнавання можуть виконувати одну з двох основних функцій: ідентифікація, що визначає вхідні дані як частину попередньо визначеного класу, та оцінка, яка дозволяє визначити клас, який ще не був встановлений [1]-[7].

Зазвичай проблеми розпізнавання формують класифікаційні завдання. Класифікації можуть бути базовані на природних середовищах (під повним наглядом) або на узгодженості даних (на основі попередніх знань). Розпізнавання образів продовжує вдосконалюватися завдяки розвитку складних і обчислювально інтенсивних інструментів [1]-[7].

Теорія розпізнавання образів виходить з припущення, що людський фактор якимось пов'язаний з процесом розпізнавання. Наприклад, вибір наступного ходу в шахах залежить від поточного розташування фігур на дошці, а рішення про купівлю або продаж акцій ґрунтується на складному наборі фінансових даних. Таким чином, мета розпізнавання образів полягає в розумінні цих складних процесів прийняття рішень і автоматизації їх за допомогою комп'ютерних систем [1]-[7].

Розпізнавання шаблонів можна визначити як здатність машини за допомогою відповідного програмного забезпечення спостерігати за навколишнім середовищем, навчатися розрізняти різні елементи з різних точок зору і приймати логічні рішення на основі розпізнаних шаблонів. Оцінка цих шаблонів передбачає їх класифікацію у відповідні групи [1]-[7].

Оскільки область розпізнавання образів є надзвичайно великою і різноманітною, існує безліч підходів до її пояснення [1]-[7].

Інше визначення розпізнавання образів розглядає це як когнітивну дисципліну, що має на меті класифікацію об'єктів у різні категорії або групи. Розпізнавання образів є також важливим елементом системи пам'яті комп'ютера, яка використовується для прийняття рішень [1]-[7].

В загальному розпізнавання образів можна описати як процес спрощення даних, відображення інформації або інтелектуального аналізу даних. В контексті інформатики це передбачає порівняння нових вхідних даних з уже збереженими в базі даних на основі їхніх характеристик [1]-[7].

Штучний інтелект включає в себе імітацію людської когнітивної діяльності, де машини запрограмовані думати і діяти подібно до людей. Наприклад, системи глибокого розпізнавання облич є частиною цієї технології. У цьому контексті розпізнавання образів стає важливою частиною психології і штучного інтелекту [1]-[7].

У сучасну еру штучного інтелекту розпізнавання образів та машинне навчання активно використовуються для створення моделей, які здатні швидко ідентифікувати та знаходити специфічні шаблони в даних. Розпізнавання образів має велике значення у різних програмах, зокрема в числовому аналізі та обробці зображень [1]-[7].

Більшість сучасних систем розпізнавання образів побудовані на методах штучного інтелекту. До популярних додатків відносяться системи розпізнавання мовлення, текстур, обличчя, жестів, візуальної аналітики та медичних зображень у медичній сфері [1]-[7].

Історично існували два основних підходи до розпізнавання образів: когнітивне судження (або системи суджень) та знання моделей (або структурні словники). Третій підхід, що отримав велику популярність, використовує штучні нейронні мережі (ШНМ) [1]-[7].

Застосування нейронних мереж для розпізнавання образів не завжди є оптимальним рішенням для всіх проблем. Як правило, використовуються комбінації цих трьох підходів для досягнення найкращих результатів у розпізнаванні образів [1]-[7].

Серед традиційних методів розпізнавання образів числові методи були найбільш дослідженими і широко використовувалися ще до впровадження сучасних технологій штучного інтелекту [1]-[7].

У статистичному аналізі зразки організовуються відповідно до їх рядів, при цьому кількість рядків визначає, як точка зразка інтерпретується у багатовимірному просторі. Ці об'єкти вибираються таким чином, щоб формувати візерунки, які відрізняються в різних просторах [1]-[7].

Метод працює шляхом відбору характеристик, що сприяють формуванню кластерів. Машина навчається і адаптується, як очікується, а потім використовує виявлені шаблони для виконання різних завдань. Статистичний аналіз спрямований на вибір функцій таким чином, щоб вектор зразка належав до різних груп у просторі лінійних розмірів [1]-[7].

Процес розпізнавання застосовується до завдань, що включають складні шаблони, які можна вирішити за допомогою ієрархічної геометрії. Семантична модель базується на початкових словах (наприклад, буквах алфавіту) і визначається відповідно до їх пріоритету. Таке взаємодія пов'язане зі словами та реченнями, де навчальні приклади допомагають зрозуміти граматичні правила і "вивчити" формування речень [1]-[7].

Окрім класифікації, розпізнавання структури також описує організацію шаблону на основі вихідних зразків. Цей метод застосовується до зразків, де візерунки мають дискретну структуру, яку можна зафіксувати за встановленими критеріями, такими як форми зображення тканин [1]-[7].

Обчислювальний підхід може призводити до відхилень у розробці ймовірностей, що вимагає тривалого навчання і зусиль для розрахунків [1]-[7]. Зіставлення шаблонів є одним із найосновніших та перших етапів у розпізнаванні шаблонів. Цей процес полягає у визначенні схожості між об'єктами одного типу в двох різних типах [1]-[7].

Таким чином, зіставлення шаблонів намагається виявити кореляції в шаблонах на основі статистичних коефіцієнтів. Чисельні методи часто використовуються в цифровій обробці зображень для визначення невеликих ділянок зображення, які відповідають певній сітці. Глобальні застосування включають медичну візуалізацію, контроль якості виробництва, навігацію роботів і розпізнавання облич [1]-[7].

Розпізнавання образів у обробці природної мови. Обробка природної мови є науковою областю, що займається комп'ютерним розумінням людської мови. Вона зосереджена на розпізнаванні шаблонів у текстах для виконання завдань, таких як машинний переклад, видобуток інформації та класифікація тексту [1]-[7].

Останніми роками обробка природної мови досягла значних успіхів завдяки збільшенню обсягу доступних даних і розвитку машинного навчання. Наприклад, Google Translate тепер використовує технологію нейронних мереж Google Neural Translation Machine, яка забезпечує точніші результати порівняно з попередніми методами [1]-[7].

Розпізнавання образів за допомогою нейронних мереж є сьогодні одним з найпопулярніших підходів у цій сфері. Нейронні мережі ґрунтуються на паралельних структурах, які імітують процес прийняття рішень людиною. Вони функціонують як високо паралельні комп'ютерні системи, що складаються з безлічі простих елементів, з'єднаних між собою [1]-[7].

Найбільш відомим і ефективним методом машинного навчання з використанням нейронних мереж є глибоке навчання. Воно використовує глибокі нейронні мережі для виконання завдань класифікації, що показало себе дуже успішним [1]-[7].

Сучасне розпізнавання образів за допомогою нейронних мереж є більш ефективним у порівнянні з іншими методами завдяки можливості багаторазово змінювати ваги в процесі навчання. В останні роки глибоке навчання досягло значних успіхів у наукових дослідженнях [1]-[7].

Аналіз літературних оглядів [1]-[7] показує, що жоден окремий алгоритм не забезпечує оптимальні результати у всіх випадках. Тому поєднання різних методів машинного навчання та розпізнавання образів зазвичай призводить до покращення ефективності і продуктивності систем розпізнавання [1]-[7].

Багато програмних засобів у сфері розпізнавання образів використовують гібридні розпізнавальні моделі для підвищення ефективності

розпізнавання образів, адаптуючи їх до специфіки практичних завдань і доступних даних [1]-[7].

Методи глибокого навчання демонструють чудові результати, але потребують значних обчислювальних ресурсів. У той час як "м'які" обчислювальні методи можуть часто бути більш ефективними. Також важливою є попередня обробка даних перед застосуванням штучного інтелекту для розпізнавання образів [1]-[7].

Застосування гібридних моделей дозволяє покращити пропускну здатність програм і систем для виявлення образів [1]-[7].

Розробка програмних систем розпізнавання образів включає три ключових етапи: збір та попередня обробка даних, представлення даних та прийняття рішень [1]-[7].

Процес розпізнавання образів можна розділити на такі етапи. Збір цифрових даних: збір необроблених даних або сигналів для подальшого аналізу. Фільтрація інформації: вилучення важливих характеристик із загальної маси даних. Аналіз даних: визначення ключових особливостей або шаблонів у даних. Категоризація: сортування даних на основі їхніх характеристик. Взаємодія: зв'язування даних для подальшого розуміння. Інтелектуальне оброблення: використання отриманих даних для прийняття обґрунтованих рішень [1]-[7].

Проаналізуємо компоненти системи розпізнавання образів. Система розпізнавання образів включає кілька основних компонентів. Компонент для отримання даних підтримує процес збору необроблених даних для аналізу. Модуль виділення ознак призначений для виявлення найбільш значущих характеристик або шаблонів у даних. Компонент для класифікації призначений для присвоєння міток даним на основі виявлених ознак. Модулі подальшої обробки забезпечують дії, необхідні для поліпшення точності та зменшення помилок у результатах [1]-[7].

Проаналізуємо принципи розроблення програмних систем розпізнавання образів [1]-[7]. При створенні системи розпізнавання образів

важливо враховувати кілька ключових аспектів [1]-[7]. Одним з основних принципів є використання модульної архітектури, яка дозволяє легко інтегрувати різні компоненти та алгоритми. Це є критичним, оскільки перехід між машинним навчанням і штучним інтелектом може відбутися в короткі терміни, іноді навіть за рік. Наприклад, з кожним роком точність і ефективність алгоритмів розпізнавання об'єктів зростає експоненційно. Модульність дозволяє оновлювати та тестувати нові алгоритми, що сприяє покращенню продуктивності системи з часом [1]-[7].

Крім того, важливо правильно вибрати методи навчання відповідно до завдання та доступних даних. Найпопулярнішими методами є контрольоване, напівконтрольоване та вбудоване навчання [1]-[7].

У сфері комп'ютерного зору, наприклад, контрольоване навчання використовує марковані дані (наприклад, зображення з позначками). Це дозволяє системі розпізнавання образів вивчати шаблони і робити точні прогнози на основі отриманої інформації. Для детальнішої інформації ознайомтеся з нашими матеріалами про контрольоване і неконтрольоване навчання.

Проаналізуємо приклади застосувань програмних засобів для розпізнавання об'єктів.

Один з найбільш поширених варіантів використання програмних засобів розпізнавання образів є розпізнавання зображень. У цьому випадку система комп'ютерного зору навчається ідентифікувати зображення, що містять певні шаблони. Це може включати завдання, такі як розпізнавання облич на фотографіях, класифікація об'єктів, визначення меж, а також розпізнавання позицій тіла [1]-[7].

Розпізнавання відео є категорією розпізнавання зображень і зосереджено на виявленні об'єктів, подій або дій у відеозаписах. Відео аналітика може використовуватися для таких задач, як розпізнавання осіб, виявлення вторгнень, виявлення руху, реальний моніторинг об'єктів та їх відстеження [1]-[7].

Використання розпізнавання об'єктів для прогнозування фондового ринку є класичним, але складним завданням. Це включає оцінку майбутньої вартості акцій або інших активів компанії. Лінійні та машинні методи навчання вивчалися протягом десятиліть, але нещодавно з'явилися новітні моделі, які швидко набирають популярність [1]-[7].

Оптичне розпізнавання символів є процесом, що дозволяє визначити оптичні візерунки на цифрових зображеннях. Процес включає в себе сегментацію зображення, виділення особливостей і класифікацію [1]-[7].

Машинне навчання, засноване на розпізнаванні образів, використовується для аналізу та обробки тексту. Моделі застосовуються для розуміння людської мови і створення текстових повідомлень. У цьому контексті розпізнавання шаблонів допомагає у сегментації документів і автоматичному виявленні важливих частин тексту, що робить його корисним для виявлення шахрайства у фінансовій та страховій сферах [1]-[7].

Процес рецензування рукописів включає порівняння шаблонів для виявлення узорів, що стосуються хірургічних операцій або іншого типу операцій. Для розпізнавання рукописного тексту використовуються різні програми, але це завдання є досить складним через унікальні і різноманітні стилі почерку [1]-[7].

Алгоритми розпізнавання зображень призначені для ідентифікації шаблонів у візуальних зображеннях з метою виявлення об'єктів. Одним з поширених завдань є класифікація зображень, де нейронні мережі використовуються для маркування зображень або їх частин відповідно до представлених даних. Це дозволяє створити візуальні пошукові системи, які допомагають користувачам легко знаходити і порівнювати зображення [1]-[7].

Технологія розпізнавання голосу дозволяє пристроям виконувати голосові команди і взаємодіяти з користувачем шляхом розпізнавання усної мови. Цей процес є частиною машинного навчання для обробки і розпізнавання звукових сигналів [1]-[7].

Використовуючи розпізнавання образів, машинне навчання аналізує зображення та відео для прогнозування поведінки людей в різних ситуаціях. Мета цього підходу – виявлення емоцій, думок і намірів аудиторії або клієнтів. Використання систем розпізнавання образів включає в себе глибоке навчання для розпізнавання виразів обличчя та жестів, що дозволяє поліпшити маркетингові кампанії і взаємодію з клієнтами [1]-[7].

Проаналізуємо переваги використання програмного забезпечення для розпізнавання образів [1]-[7].

Технології розпізнавання образів пропонують численні переваги, які залежать від конкретних застосувань. Зазвичай виявлення шаблонів у даних дозволяє аналізувати та прогнозувати майбутні тренди або розробляти стратегії на основі виявлених індикаторів [1]-[7]. Основні переваги полягають у розв'язанні таких практичних завдань [1]-[7].

Ідентифікація об'єктів: розпізнавання образів може допомогти в ідентифікації об'єктів, навіть під різними кутами та відстанями (наприклад, у відео на основі глибокого навчання) або виявленні небезпечних ситуацій. Зокрема, ця технологія може використовуватися для розпізнавання осіб за допомогою системи розпізнавання обличчя або аналізу руху. Нещодавно з'явилися нові системи штучного інтелекту, які можуть ідентифікувати людей за характеристиками водіння [1]-[7].

Виявлення нових шаблонів: алгоритми розпізнавання шаблонів дозволяють мислити нестандартно і виявляти шаблони, які люди можуть не помітити. Такі алгоритмічні моделі здатні знаходити тонкі зміни в даних або зв'язки між різними факторами в великих обсягах інформації. Це особливо корисно в медицині, наприклад, для діагностики за допомогою магнітно-резонансної терапії з використанням глибокого навчання [1]-[7].

Інформаційна безпека: розпізнавання образів використовується в інформаційній безпеці та інформаційних технологіях для виявлення вторгнень через системи моніторингу мереж або комп'ютерних систем на наявність зловмисної активності чи атак [1]-[7].

Прогнозування майбутніх подій: прогнозування майбутніх подій є важливою частиною багатьох систем розпізнавання образів. Це включає передбачення ринкових цін, інвестиційних можливостей або тенденцій у ланцюгу постачання [1]-[7].

Сучасні методи машинного навчання забезпечують високоякісні дані та зразки в режимі реального часу. Це дозволяє приймати обґрунтовані рішення на основі точних даних. Важливим аспектом є швидкість нових технологій розпізнавання образів, які можуть перевершити традиційні методи і забезпечити нові можливості. Наприклад, виявлення лікувальних шаблонів і визначення факторів ризику в медичних базах даних можуть швидко надати лікарям критично важливу інформацію [1]-[7].

Нейронні мережі дозволяють виявляти закономірності в великих обсягах даних, що неможливо зробити за допомогою звичайних статистичних методів. Розпізнавання образів є особливо важливим у медицині, зокрема в криміналістичному аналізі [1]-[7].

Алгоритми розпізнавання образів можна застосовувати до різних цифрових носіїв, таких як зображення, текст або відео. Ці методи забезпечують класифікацію даних для прийняття більш обґрунтованих рішень і можуть бути використані повністю автоматично для вирішення складних аналітичних завдань [1]-[7].

Визначено, що розпізнавання об'єктів є процесом, за допомогою якого програмне забезпечення виявляє шаблони в даних і використовує їх для прогнозування або класифікації за допомогою обчислювальних алгоритмів. Системи розпізнавання образів використовують комп'ютерні алгоритми для аналізу та інтерпретації різних типів даних, вхідні дані можуть бути представлені у формі тексту, зображень або звуків. Проаналізовано процес розробки програмних систем розпізнавання образів. Проаналізовано компоненти та принципи розробки програмних систем розпізнавання образів. Проаналізовано приклади застосувань програмних засобів для розпізнавання об'єктів. Відзначено, що одним з найбільш поширених варіантів використання

програмних засобів розпізнавання образів є розпізнавання зображень. Відзначено, що розпізнавання образів за допомогою нейронних мереж є сьогодні широко використовуваним та ефективним підходом. Нейронні мережі ґрунтуються на паралельних структурах, які імітують процес прийняття рішень людиною. Показано, що жоден алгоритм не забезпечує оптимальні результати у всіх випадках. Тому доцільно розробити програмне забезпечення, що дозволяє використовувати різні моделі на основі штучних нейронних мереж для розпізнавання образів, що дозволить покращити ефективність програмних систем розпізнавання образів.

1.2 Аналіз програмного забезпечення для підтримки процесу розпізнавання об'єктів

Проаналізуємо програмне забезпечення для підтримки процесу розпізнавання об'єктів [8]-[14].

Програмне забезпечення Clarifai (рис. 1.1) спеціалізується на комп'ютерному зорі та розпізнаванні звуків. Програмне забезпечення Clarifai використовує моделі ШІ для пошуку та класифікації даних, допомагаючи організаціям створювати ефективні інтелектуальні рішення [8]-[12]. Програмне забезпечення Clarifai може обробляти понад 20 мільйонів відео, зображень, аудіофайлів та текстів, а також відфільтровувати небажаний контент. Програмне забезпечення Clarifai використовує обробку природної мови для аналізу тексту на предмет тематики. Програмне забезпечення Clarifai підтримує популярні формати зображень, такі як JPEG, PNG, TIFF і WEBP. Програмне забезпечення Clarifai для розпізнавання зображень дозволяє знаходити зображення на основі інших зображень. Наприклад, можна знайти зображення за їхньою схожістю, що може бути важче зробити за допомогою текстових описів [8]-[12].

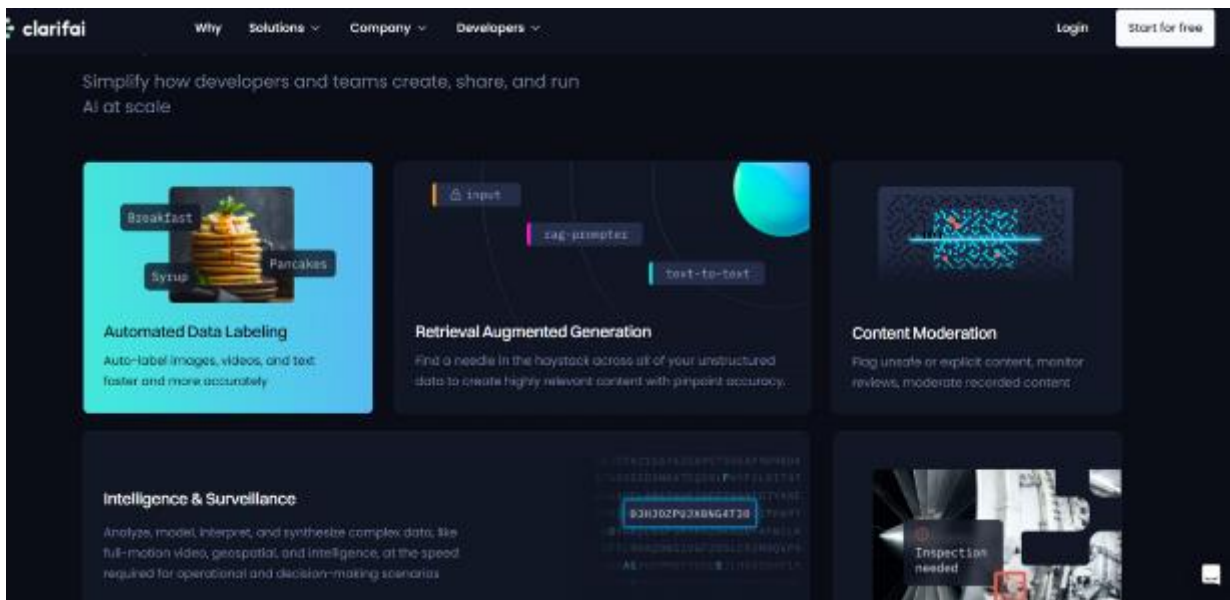


Рисунок 1.1 – Програмне забезпечення Clarifai [12]

Програмне забезпечення Clarifai допомагає знайти схожі зображення, просто вибравши фотографії та вказавши параметри пошуку [8]-[12].

Програмне забезпечення Clarifai є безкоштовним програмним інтерфейсом для розпізнавання зображень, який здатний маркувати, організовувати та інтерпретувати зображення і відео за допомогою машинного навчання і штучного інтелекту. Завдяки технології комп'ютерного зору це програмне забезпечення допомагає федеральним і комерційним організаціям визначати вміст даних, спрощуючи рішення їхніх задач [8]-[12].

Програмне забезпечення для розпізнавання зображень Clarifai дозволяє програмам розуміти зображення та відео. Воно приймає зображення як вхідні дані, а алгоритми комп'ютерного зору видають результати, такі як мітки чи обмежувальні рамки. Серед інших функцій програми Clarifai для розпізнавання зображень є відновлення зображень, розпізнавання об'єктів і реконструкція сцен. Ці можливості часто інтегруються в інтелектуальні програми. Таке програмне забезпечення може використовуватися як дослідниками даних для навчання моделей розпізнавання зображень, так і розробниками, які хочуть додати функції розпізнавання зображень до інших програм [8]-[12].

Доступ до програмного забезпечення Clarifai може здійснюватися через бібліотеки машинного навчання, фреймворки або через комплексні платформи, в залежності від потреб користувача [8]-[12]. Програмне забезпечення Clarifai пропонує широкий спектр можливостей для розпізнавання різних типів зображень і об'єктів [8]-[12].

Основними характеристиками програмного забезпечення Clarifai для розпізнавання зображень є такі [8]-[12]:

- модульність: програмне забезпечення Clarifai пропонує різні модулі для обробки зображень, відео та аудіо. Це дозволяє користувачам вибирати та комбінувати функції відповідно до конкретних потреб задачі [8]-[12];

- швидкість обробки: програмне забезпечення Clarifai забезпечує швидкий час відгуку, зокрема обробка зображень і відео може виконуватись в межах долі секунди, що дозволяє ефективно справлятися з великими обсягами даних [8]-[12];

- потужні засоби розпізнавання зображень: програмне забезпечення Clarifai використовує потужні моделі для ідентифікації та категоризації об'єктів у зображеннях. Це включає можливість пошуку зображень на основі схожості з іншими зображеннями [8]-[12];

- засоби обробки відео: крім зображень, програмне забезпечення Clarifai може обробляти відео, що дозволяє виявляти та класифікувати об'єкти, події або дії в реальному часі [8]-[12];

- висока масштабованість: програмне забезпечення Clarifai може обробляти величезні обсяги даних, включаючи понад відео, зображення, аудіофайли і тексти [8]-[12];

- програмне забезпечення Clarifai пропонує зручний доступ через прикладні програмні інтерфейси та фреймворки, що дозволяє інтегрувати його можливості в різноманітні додатки та платформи [8]-[12];

- підтримка різних форматів: програмне забезпечення Clarifai підтримує популярні формати файлів зображень, зокрема, JPEG, PNG, TIFF і WEBP [8]-[12];

- глибоке навчання: використання технологій глибокого навчання дозволяє точніше виявляти та класифікувати об'єкти завдяки вдосконаленим нейронним мережам [8]-[12];

- інтуїтивно зрозумілий інтерфейс: програмне забезпечення Clarifai пропонує легкий у використанні інтерфейс, який дозволяє швидко налаштовувати та запускати моделі [8]-[12];

- точність: програмне забезпечення Clarifai підтримує високий рівень точності у розпізнаванні та класифікації зображень [8]-[12];

- аналіз даних: програмне забезпечення Clarifai підтримує можливість проводити глибокий аналіз і виявлення трендів у даних [8]-[12].

Серед недоліків програмного забезпечення Clarifai можна відзначити залежність від хмари для обробки зображень, що може викликати питання про конфіденційність і безпеку даних, особливо для чутливих або приватних даних. Іншим недоліком є те, що процес налаштування та інтеграції з прикладним програмним інтерфейсом є досить складним, особливо якщо у користувачів немає досвіду в роботі з подібними інструментами. Крім того, розпізнавальні моделі, що використовуються у програмному забезпеченні Clarifai, можуть потребувати регулярного оновлення для підтримки високого рівня точності і відповідності новим даним та трендам [8]-[12].

Програмне забезпечення Imagga (рис. 1.2) представляє собою програмне рішення для розпізнавання зображень, яке призначене для розробників і компаній, які хочуть інтегрувати функціонал розпізнавання зображень у свої програми. Програмне забезпечення Imagga забезпечує підтримку різних технологій, зокрема, обчислень в хмарі, що дозволяє використовувати це ПЗ у різних галузях, зокрема, медіа та розваг, комерції і роздрібної торгівлі, реклами, нерухомості, Інтернету речей. Програмне забезпечення Imagga пропонує кілька автоматичних функцій, включаючи сортування, упорядкування і маркування зображень за категоріями, кольорами, тегами або індивідуальними особливостями [8]-[11], [13].

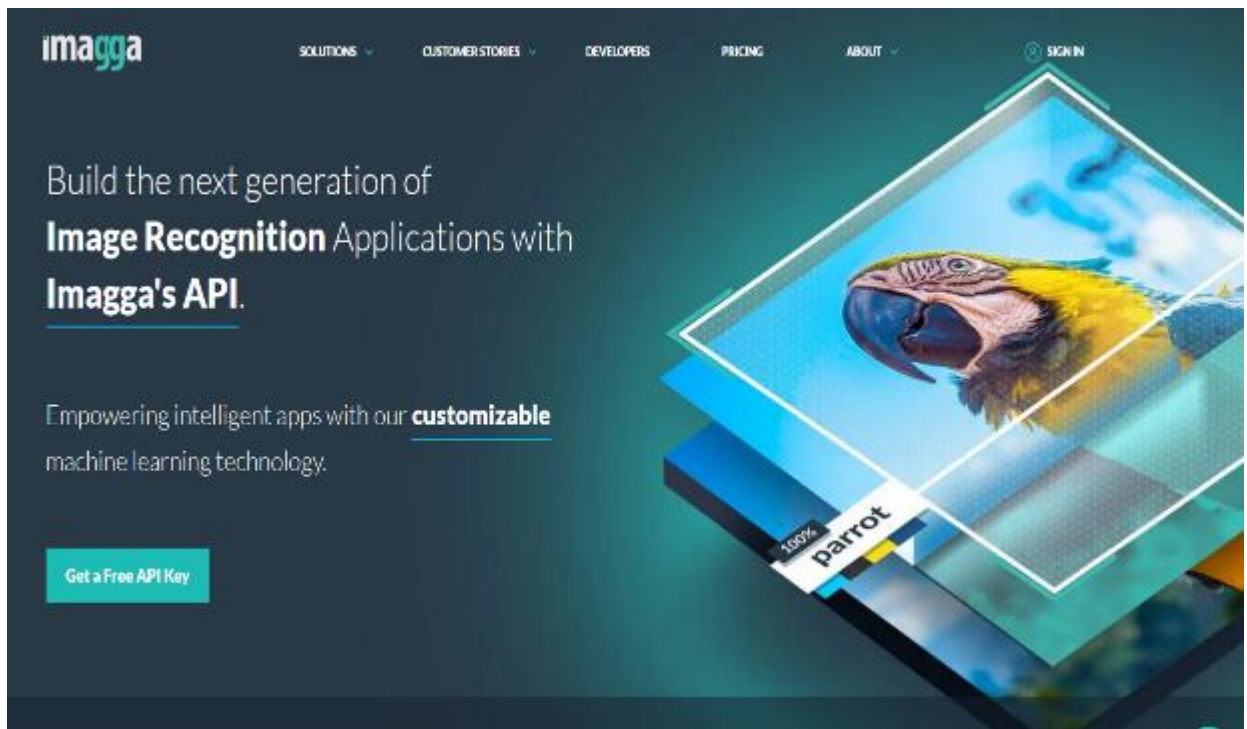


Рисунок 1.2 – Програмне забезпечення Imagga [13]

Програмне забезпечення Imagga дозволяє використовувати вбудовані можливості або адаптувати їх під свої конкретні потреби. Гнучкість Imagga дозволяє йому відповідати різним вимогам і рівням навичок [8]-[11], [13].

Програмне забезпечення Imagga являє собою автоматизоване рішення для додавання тегів до зображень і управління категоріями, яке розроблене для роботи з великими обсягами зображень. Воно призначене для розпізнавання цифрових зображень і включає бібліотеку для класифікації активів і управління метаданими. Інструменти пошуку і фільтрації допомагають знайти та управляти мультимедійними файлами [8]-[11], [13].

Засоби прикладного інтерфейсу програмного забезпечення Imagga також дозволяють створювати звіти та проводити аналіз. В порівнянні з іншими безкоштовними прикладними інтерфейсами для розпізнавання зображень, Imagga відрізняється своєю доступністю [8]-[11], [13].

Програмне забезпечення Imagga надає ефективні інструменти для розпізнавання і управління зображеннями, що робить його корисним як для великих підприємств, так і для невеликих компаній [8]-[11], [13].

Програмне забезпечення Imagga має такі особливості [8]-[11], [13]:

- засоби автоматичного тегування: програмне забезпечення Imagga автоматично додає теги до зображень, що полегшує їх організацію та пошук відповідних зображень [8]-[11], [13];

- засоби класифікації та сортування: програмне забезпечення Imagga дозволяє класифікувати і сортувати зображення за категоріями, кольорами та іншими критеріями [8]-[11], [13];

- фільтрація контенту: програмне забезпечення Imagga надає вбудовані інструменти для фільтрації зображень та іншого контенту, що допомагають контролювати наявність небажаного або неприйняттого контенту серед заданого набору зображень [8]-[11], [13];

- пошук за зображеннями: програмне забезпечення Imagga забезпечує можливість здійснювати пошук зображень на основі їх схожості, а не лише за текстовими тегами [8]-[11], [13];

- засоби прикладного програмного інтерфейсу для інтеграції, що дозволяє інтегрувати функції розпізнавання зображень в інші програми та системи [8]-[11], [13];

- управління метаданими: програмне забезпечення Imagga надає інструменти для управління і організації метаданих зображень, що полегшують їх подальше використання і аналіз [8]-[11], [13];

- підтримка обробки великих обсягів зображень для масштабованих рішень [8]-[11], [13];

- засоби аналізу та звітності: програмне забезпечення Imagga надає можливість створювати звіти та проводити аналіз даних зображень [8]-[11];

- масштабованість: програмне забезпечення Imagga підходить для розробки великих програмних продуктів завдяки можливості обробки значних обсягів зображень [8]-[11], [13];

- гнучкість: програмне забезпечення Imagga надає можливість налаштування та розширення функцій відповідно до специфічних потреб користувачів [8]-[11], [13];

– інтуїтивний інтерфейс: програмне забезпечення Imagga надає простий у використанні інтерфейс, який дозволяє розробникам легко інтегрувати розпізнавання зображень у свої програми [8]-[11], [13];

Серед недоліків програмного забезпечення Imagga можна відзначити залежність від хмари для обробки зображень, що може викликати питання про конфіденційність і безпеку даних, особливо для чутливих або приватних даних. Іншим недоліком є те, що процес налаштування та інтеграції з прикладним програмним інтерфейсом є досить складним, особливо якщо у користувачів немає досвіду в роботі з подібними інструментами. Крім того, розпізнавальні моделі, що використовуються у програмному забезпеченні Imagga, можуть потребувати регулярного оновлення для підтримки високого рівня точності і відповідності новим даним та трендам [8]-[11], [13].

Програмне забезпечення Hive (рис. 1.3) є це хмарною програмною платформою на основі штучного інтелекту, призначеною для пошуку, розуміння, класифікації та виявлення вмісту, зокрема, зображень, як в Інтернеті, так і в спеціалізованих базах даних [8]-[11], [14].

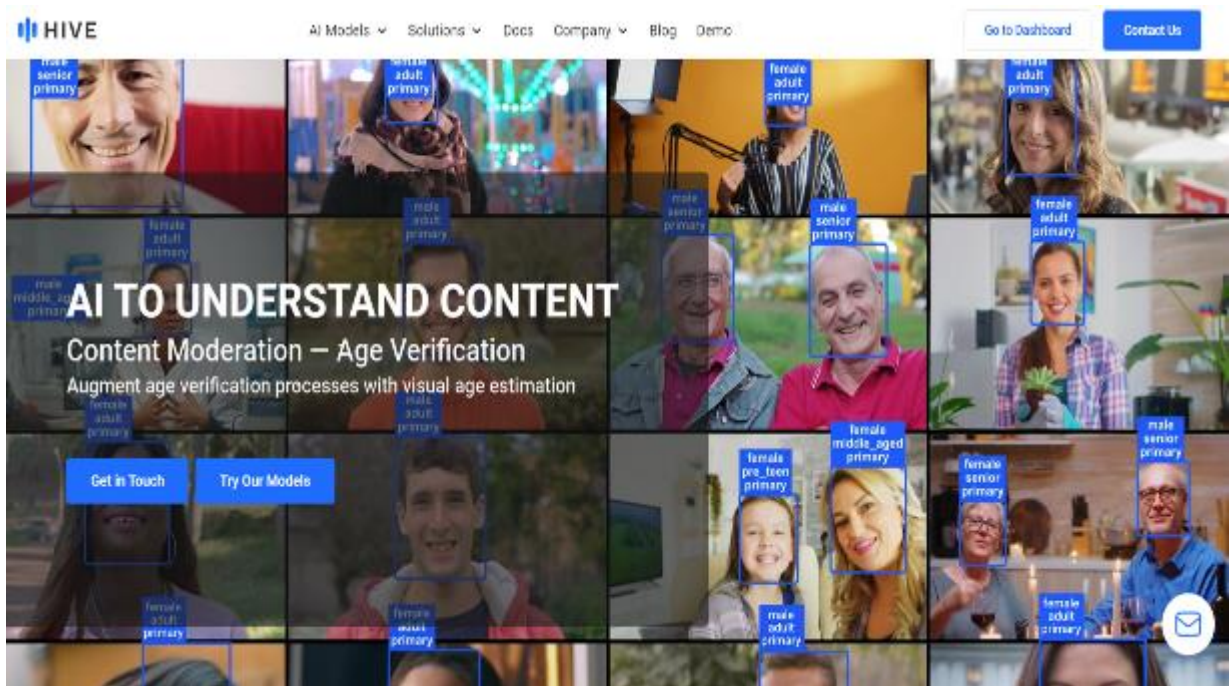


Рисунок 1.3 – Програмне забезпечення Hive [14]

Програмне забезпечення HIVE здатне генерувати зображення на основі текстових підказок і надає готові рішення для різних типів організацій, включаючи програми для знайомств, Інтернет спільноти, ринків та платформ. Програмне забезпечення HIVE особливо підходить для компаній і агентств, які займаються моніторингом репутації бренду, а також для бізнесів, які потребують контролю за безпекою контенту, наприклад, програмного забезпечення для знайомств [8]-[11], [14].

Програмне забезпечення HIVE є потужним хмарним рішенням для аналізу візуального контенту, яке забезпечує миттєву обробку зображень і відео, виявляючи десятки категорій. Воно також здійснює дослідження тексту на зображеннях, відрізняючи спам та інші небезпечні елементи. HIVE визначає об'єкти, сцени та події на візуальному контенті, а також витягує текст на різних мовах через обробку зображень. Платформа транскрибує аудіо в режимі реального часу, виявляє демографічні атрибути, розпізнає логотипи та їх розташування, а також перекладає текст на різні мови для кращого доступу та взаємодії. Крім того, програмне забезпечення HIVE може класифікувати контент, створений штучним інтелектом, для виявлення фальсифікацій і застосування фільтрів обличчя [8]-[11], [14].

Програмне забезпечення HIVE має такі особливості [8]-[11], [14]:

- візуальне дослідження зображень: програмне забезпечення HIVE досліджує зображення та відео, виявляючи різні категорії вмісту з високою швидкістю обробки [8]-[11], [14];
- можливість розпізнавання загальних об'єктів, деталей та подій на зображеннях і відео [8]-[11], [14];
- оптичне розпізнавання символів: програмне забезпечення HIVE витягує накладення та текст із зображень і відео на різних мовах [8]-[11], [14];
- визначає ключові демографічні характеристики, такі як вік і стать, для застосування вікових обмежень і націлювання рекламного контенту програмного забезпечення [8]-[11], [14];

– засоби розпізнавання логотипів: програмне забезпечення Hive ідентифікує тисячі логотипів на зображеннях і відео для оцінки ефективності розміщення продукту [8]-[11], [14];

– класифікація контенту, створеного штучним інтелектом: програмне забезпечення Hive виявляє контент, створений штучним інтелектом, зокрема глибокі підробки та використання фільтрів для обличчя, для точної ідентифікації фальсифікацій [8]-[11], [14].

Серед недоліків програмного забезпечення Hive можна відзначити залежність від хмари для обробки зображень, що може викликати питання про конфіденційність і безпеку даних, особливо для чутливих або приватних даних. Іншим недоліком є те, що процес налаштування та інтеграції з прикладним програмним інтерфейсом є досить складним, особливо якщо у користувачів немає досвіду в роботі з подібними інструментами. Крім того, розпізнавальні моделі, що використовуються у програмному забезпеченні Hive, можуть потребувати регулярного оновлення для підтримки високого рівня точності і відповідності новим даним та трендам [8]-[11], [14].

Порівняльну характеристику програмного забезпечення для розпізнавання об'єктів наведено у таблиці 1.1.

Таблиця 1.1 – Порівняння програмного забезпечення для розпізнавання об'єктів

Критерій порівняння	Clarifai [12]	Imagga [13]	Hive [14]
Інтуїтивно зрозумілий інтерфейс	+	+—	—
Потужні засоби розпізнавання зображень	+—	+	+—
Модульність	+	+—	—
Висока швидкість обробки зображень	+	+—	—
Висока масштабованість	+—	+	—

За результатами проведеного аналізу можна зробити висновок, що у наш час існує досить багато програмних засобів для розпізнавання об'єктів. Проте деякі програмні засоби для розпізнавання об'єктів можуть неправильно ідентифікувати об'єкти. Це може статися через використання неефективних алгоритмів розпізнавання образів, або обмеження у використовуваних алгоритмах, недостатню якість зображення або складність середовища. Інші програмні засоби для розпізнавання об'єктів можуть вимагати значної обчислювальної потужності і великої кількості пам'яті, що може бути витратним і обмежити використання в деяких умовах. Впровадження та підтримка інших програмних систем розпізнавання об'єктів можуть бути дорогими, особливо для малих і середніх підприємств. Тому актуальною є розробка програмного забезпечення для розпізнавання об'єктів.

При розробці програмного забезпечення для підтримки процесу розпізнавання об'єктів необхідно забезпечити такі функціональні вимоги:

- підтримка можливості завантаження зображення для ідентифікації об'єктів;
- підтримка можливості пакетного завантаження набору зображень для виявлення об'єктів на них;
- можливість вибору розпізнавальної моделі для ідентифікації об'єктів;
- можливість розпізнавання об'єктів на зображеннях;
- можливість збереження вихідних даних з розпізнаними об'єктами.

1.3 Висновки за розділом 1

Визначено, що розпізнавання об'єктів є процесом, за допомогою якого програмне забезпечення виявляє шаблони в даних і використовує їх для прогнозування або класифікації за допомогою обчислювальних алгоритмів. Системи розпізнавання образів використовують комп'ютерні алгоритми для аналізу та інтерпретації різних типів даних, вхідні дані можуть бути представлені у формі тексту, зображень або звуків. Проаналізовано процес

розробки програмних систем розпізнавання образів. Проаналізовано компоненти та принципи розробки програмних систем розпізнавання образів. Проаналізовано приклади застосувань програмних засобів для розпізнавання об'єктів. Відзначено, що одним з найбільш поширених варіантів використання програмних засобів розпізнавання образів є розпізнавання зображень. Відзначено, що розпізнавання образів за допомогою нейронних мереж є сьогодні широко використовуваним та ефективним підходом. Нейронні мережі ґрунтуються на паралельних структурах, які імітують процес прийняття рішень людиною. Показано, що жоден алгоритм не забезпечує оптимальні результати у всіх випадках. Тому доцільно розробити програмне забезпечення, що дозволяє використовувати різні моделі на основі штучних нейронних мереж для розпізнавання образів, що дозволить покращити ефективність програмних систем розпізнавання образів.

За результатами проведеного аналізу зроблено висновок, що у наш час існує досить багато програмних засобів для розпізнавання об'єктів. Проте деякі програмні засоби для розпізнавання об'єктів можуть неправильно ідентифікувати об'єкти. Це може статися через використання неефективних алгоритмів розпізнавання образів, або обмеження у використовуваних алгоритмах, недостатню якість зображення або складність середовища. Інші програмні засоби для розпізнавання об'єктів можуть вимагати значної обчислювальної потужності і великої кількості пам'яті, що може бути витратним і обмежити використання в деяких умовах. Впровадження та підтримка інших програмних систем розпізнавання об'єктів можуть бути дорогими, особливо для малих і середніх підприємств. Тому актуальною є розробка програмного забезпечення для розпізнавання об'єктів.

Сформульовано функціональні вимоги до програмного забезпечення для розпізнавання об'єктів.

2 МАТЕРІАЛИ І МЕТОДИ

2.1 Вибір мови програмування

При розробці програмного забезпечення для розпізнавання об'єктів було використано мову програмування Python [15], [16].

Python – це об'єктно-орієнтована мова з динамічною типізацією, яка вирізняється своєю простотою і універсальністю. Вона активно використовується в аналізі даних, машинному навчанні, чисельних обчисленнях і веброзробці [15], [16].

Python часто вважається кращим вибором для технічних завдань завдяки легкості написання і читання коду в порівнянні з іншими мовами. Python використовує простий і зрозумілий синтаксис, що робить код легким для читання і написання. Це допомагає новачкам швидше освоювати мову та знижує ймовірність помилок у коді [15], [16].

Мова Python має велику стандартну бібліотеку, що забезпечує широкий спектр функціональних можливостей, таких як робота з файлами, Інтернет протоколами, системними командами тощо [15], [16].

Python має кілька ключових переваг: його легко освоїти і навчання не є дорогим; розвинена екосистема включає численні бібліотеки та ресурси, а також можливість інтеграції з кодом на C; мова підтримує багато платформ, що робить її універсальною у використанні; вона також підтримує об'єктно-орієнтоване програмування, включаючи поліморфізм і успадкування [15], [16].

Мова програмування Python є однією з основних мов для машинного навчання завдяки бібліотекам, таким як TensorFlow, Keras і PyTorch, які надають потужні інструменти для створення і навчання моделей. Мова програмування Python часто використовується для автоматизації рутинних завдань, написання скриптів і управління системами завдяки своїй простоті і універсальності [15], [16].

Python використовують у наукових дослідженнях і обчисленнях завдяки бібліотекам для чисельних обчислень і моделювання. Python є дуже універсальною і потужною мовою програмування, яка підходить для широкого спектра завдань. Його простота, підтримка різних парадигм програмування і велика кількість бібліотек роблять його ідеальним вибором для багатьох типів проектів [15], [16].

Обґрунтування вибору мови програмування для розробки програмного забезпечення для розпізнавання об'єктів наведено у таблиці 2.1.

Таблиця 2.1 – Обґрунтування вибору мови програмування для розробки програмного забезпечення для розпізнавання об'єктів

Критерій порівняння мов програмування	Мова програмування		
	Python	R	Matlab
Легкість вивчення	+	+—	—
Бібліотеки для машинного навчання	+	+—	+
Зручність для створення ПЗ для розпізнавання об'єктів	+	—	—
Інтеграція з іншими засобами	+	+—	+—
Простота та зрозумілість коду	+	+—	+—

Отже, для реалізації програмного забезпечення для підтримки процесу розпізнавання об'єктів обрано мову програмування Python, яка має багатий набір бібліотек та фреймворків для розпізнавання об'єктів і машинного навчання, що спрощують реалізацію складних алгоритмів і моделювання, а також забезпечують високий рівень абстракції для швидкого розроблення прототипів програмного забезпечення. Крім того, Python легко інтегрується з іншими мовами програмування і технологіями, такими як C++ для високопродуктивних обчислень або JavaScript для розроблення

вебінтерфейсів. Це дозволяє створювати комплексні рішення, які можуть включати різні компоненти і технології.

2.2 Вибір середовища розробки для створення програмного забезпечення для розпізнавання об'єктів

В якості середовища розробки для створення програмного забезпечення для підтримки процесу розпізнавання об'єктів було обрано середовище Sublime Text [17], [18].

Середовище розробки Sublime Text є редактором, який вирізняється своєю швидкістю, простотою та гнучкістю. Він забезпечує швидку роботу і низьке споживання системних ресурсів, що робить його ідеальним для роботи з великими файлами та проектами [17], [18].

Редактор має простий і мінімалістичний інтерфейс, що дозволяє сконцентруватися на написанні коду без відволікань [17], [18].

Середовище Sublime Text підтримує редагування кількох рядків коду одночасно завдяки різним курсорам і пропонує ефективні інструменти для пошуку та заміни тексту. Sublime Text підтримує численні плагіни та розширення, які можна встановити через менеджер пакетів, що дозволяє додати нові можливості, такі як синтаксичне підсвічування для різних мов або інтеграцію з системами контролю версій [17], [18].

Редактор Sublime Text підтримує широкий спектр мов програмування і може бути налаштований для роботи з новими мовами завдяки плагінам. Також він пропонує функції для швидкого переходу до символів, файлів або рядків, що спрощує переміщення по великим кодовим базам [17], [18].

Sublime Text дозволяє користувачам налаштовувати кольорові теми, вибираючи з доступних варіантів або створюючи власні, а можливість запису макросів допомагає автоматизувати повторювані завдання, підвищуючи ефективність роботи. Тому Sublime Text підходить як для новачків, так і для досвідчених розробників завдяки своїй простоті використання та багатству функцій [17], [18].

Також для розробки використано бібліотеку Tensorflow. TensorFlow є потужною платформою з відкритим кодом для машинного навчання, яка надає широкий набір інструментів, бібліотек і ресурсів спільноти. Ця гнучка екосистема дозволяє дослідникам просувати інноваційні досягнення в галузі машинного навчання, а розробникам спрощує створення і впровадження програм, що використовують машинне навчання [19].

Обґрунтування вибору середовища розробки для створення програмного забезпечення для розпізнавання об'єктів наведено у таблиці 2.2.

Таблиця 2.2 – Обґрунтування вибору середовища розробки для створення програмного забезпечення для розпізнавання об'єктів

Критерій порівняння середовищ розробки	Середовища розробки		
	Sublime Text	Visual Studio	Atom
Зручність інтерфейсу	+	+	—
Підтримка різних мов програмування	+	+	+—
Функціональність	+	+	+—
Підтримка плагінів	+	+—	+—
Мінімалістичний інтерфейс	+	—	+

Таким чином, для створення програмного забезпечення для розпізнавання об'єктів обрано середовище розробки Sublime Text, що є зручним редактором та дозволяє легко редагувати файли без значного споживання ресурсів системи. Це особливо корисно при роботі з великими наборами програмних кодів або проектами, що включають велику кількість файлів. Крім того, Sublime Text підтримує велику кількість плагінів і налаштувань, які можна адаптувати під специфічні потреби проекту. Це дозволяє розширювати функціональність редактора, наприклад, для підтримки

специфічних синтаксичних підсвіток або інтеграції з системами контролю версій.

2.3 Розробка моделі взаємодії користувача з програмним забезпеченням для розпізнавання об'єктів

З метою можливості розробки програмного забезпечення для розпізнавання об'єктів створимо відповідну модель у вигляді діаграми прецедентів.

При розробленні моделі взаємодії користувача з програмним забезпеченням для розпізнавання об'єктів враховуємо те, що користувачі програми мають можливості завантаження зображення для ідентифікації об'єктів; пакетного завантаження набору зображень для виявлення об'єктів на них; вибору розпізнавальної моделі для ідентифікації об'єктів; розпізнавання об'єктів на зображеннях; збереження вихідних даних з розпізнаними об'єктами.

Основний успішний сценарій користувача при роботі з програмним забезпеченням для розпізнавання об'єктів можна описати таким чином:

- користувач програмного забезпечення для розпізнавання об'єктів запускає програму;
- після запуск програмного забезпечення для розпізнавання об'єктів користувачу відображається повідомлення зі списком можливих дій;
- користувач бачить список можливих дій та обирає вибір зображення для ідентифікації об'єктів;
- користувач обирає зображення, повертається до головного меню;
- користувач обирає модель для розпізнавання зображень;
- користувач виконує розпізнавання об'єктів на обраному зображенні, обираючи відповідну дію в меню програми;
- користувач завершує роботу з програмним забезпеченням для розпізнавання об'єктів.

Враховуючи сценарії роботи користувачів при роботі з програмою, модель взаємодії користувача з програмним забезпеченням для розпізнавання об'єктів подамо у вигляді діаграми прецедентів, зображеної на рис. 2.1.

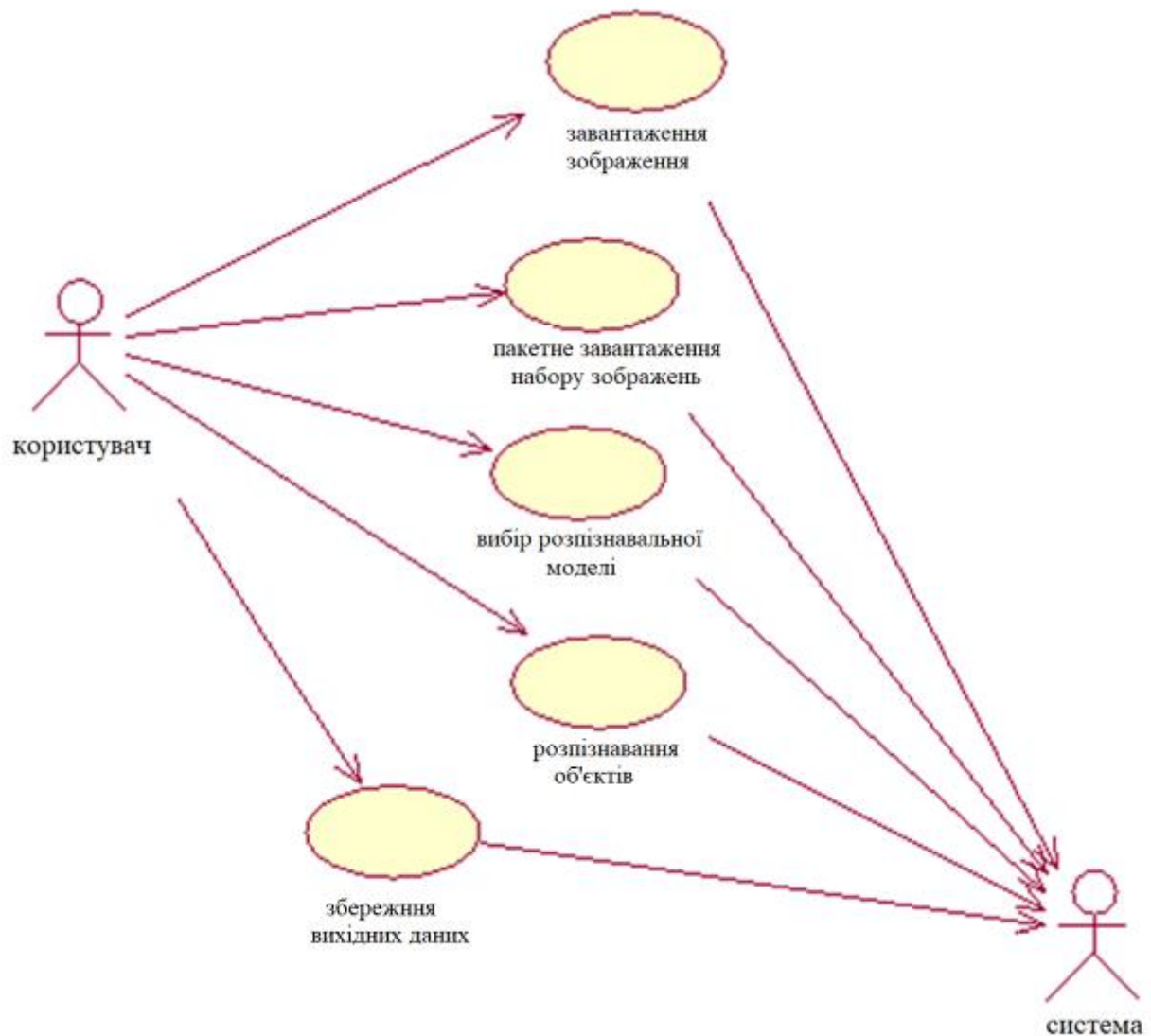


Рисунок 2.1 – Модель взаємодії користувача з програмним забезпеченням для розпізнавання об'єктів

Таким чином, створено модель взаємодії користувача з програмним забезпеченням для розпізнавання об'єктів, яка подана у вигляді діаграми прецедентів, та дозволяє розроблювати програмне забезпечення для розпізнавання об'єктів.

2.4 Висновки за розділом 2

Для реалізації програмного забезпечення для підтримки процесу розпізнавання об'єктів обрано мову програмування Python, яка має великий набір бібліотек та фреймворків для розпізнавання об'єктів і машинного навчання, що спрощують реалізацію складних алгоритмів і моделювання, а також забезпечують високий рівень абстракції для швидкого розроблення прототипів програмного забезпечення. Крім того, Python легко інтегрується з іншими мовами програмування і технологіями, такими як C++ для високопродуктивних обчислень або JavaScript для розроблення вебінтерфейсів. Це дозволяє створювати комплексні рішення, які можуть включати різні компоненти і технології.

Для створення програмного забезпечення для розпізнавання об'єктів обрано середовище розробки Sublime Text, що є зручним редактором та дозволяє легко редагувати файли без значного споживання ресурсів системи. Це особливо корисно при роботі з великими наборами програмних кодів або проектами, що включають велику кількість файлів. Крім того, Sublime Text підтримує велику кількість плагінів і налаштувань, які можна адаптувати під специфічні потреби проекту. Це дозволяє розширювати функціональність редактора, наприклад, для підтримки специфічних синтаксичних підсвіток або інтеграції з системами контролю версій.

Створено модель взаємодії користувача з програмним забезпеченням для розпізнавання об'єктів, яка подана у вигляді діаграми прецедентів, та дозволяє розроблювати програмне забезпечення для розпізнавання об'єктів.

3 ОПИС ПРОГРАМИ

3.1 Структура програмного забезпечення для розпізнавання об'єктів

Структуру програмного забезпечення для розпізнавання об'єктів наведено на рис. 3.1.



Рисунок 3.1 – Структура програмного забезпечення для розпізнавання об'єктів

Як видно з рис. 3.1, програмне забезпечення для розпізнавання об'єктів складається з трьох компонентів.

Програмні модулі для синтезу розпізнавальних моделей призначені для створення моделей на основі нейромережевих технологій. Синтезовані моделі у подальшому можуть використовуватися для розпізнавання об'єктів, подібним до тих, що використовувалися у процесі синтезу відповідної моделі.

Інтерфейсна частина програмного забезпечення для розпізнавання об'єктів забезпечує взаємодію користувача з системою. Враховуючи те, що розроблену програму у майбутньому можна використовувати як частину більш складної системи, прийнято рішення про використання як базису консольного режиму взаємодії. Таким чином, програма забезпечує взаємодію з користувачем через консоль та стандартні термінальні засоби операційної системи.

Програмні модулі для роботи з наборами даних забезпечують завантаження файлів з даними, що містять зображення для обробки, а також засоби попередньої обробки даних та формування вибірок для синтезу розпізнавальних моделей. Ці ж модулі забезпечують збереження результатів розпізнавання об'єктів у вихідних файлах.

3.2 Функціонування програмного забезпечення для розпізнавання об'єктів

Функціонування програмного забезпечення для розпізнавання об'єктів подамо за допомогою схеми, зображеної на рис. 3.2.

На початку процесу функціонування програмного забезпечення для розпізнавання об'єктів необхідно встановити програмне середовище, включаючи вибір мови програмування, фреймворків та бібліотек. Потім слідє попередня підготовка даних, які використовуватимуться для навчання та тестування розпізнавальних моделей на основі нейронних мереж. Після цього обираються та налаштовуються моделі (наприклад, SSD ResNetXX, Faster R-CNN ResNetXX або Inception ResNet V2) і проводиться їх навчання на підготовлених даних. Після навчання моделей виконується тестування для оцінювання їх ефективності. Якщо результати оцінювання ефективності моделей задовільні і похибка розпізнавання є прийнятною, процес навчання завершується, і розпізнавальну модель можна використовувати для ідентифікації об'єктів. У випадку незадовільних результатів, необхідно

коригувати набір даних (якщо є проблеми з виділенням областей або помилки) або вибрати інший тип нейромережевої розпізнавальної моделі.

Після цього розпізнавальна модель може використовуватися як частина програмного забезпечення для розпізнавання об'єктів.

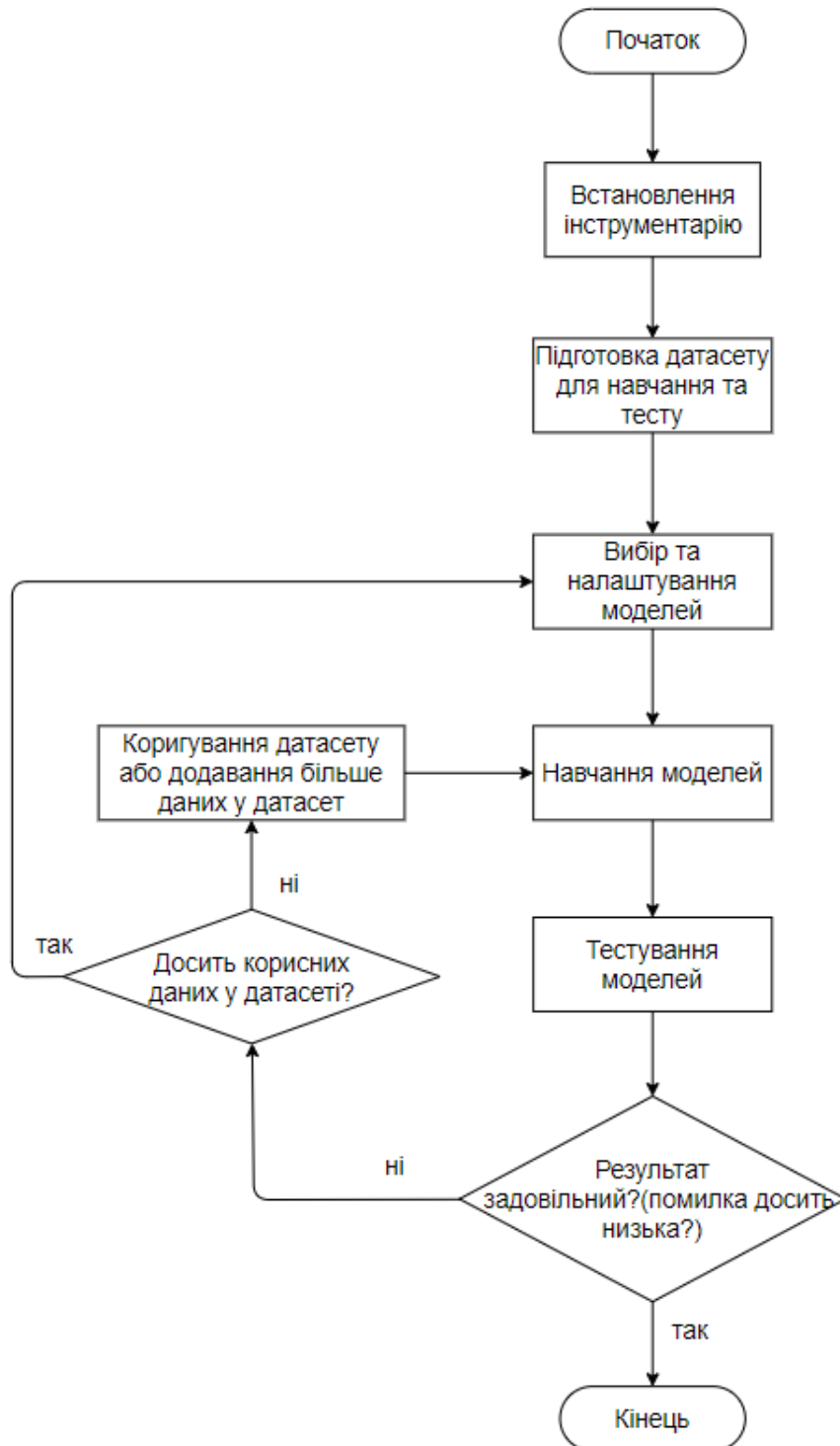


Рисунок 3.2 – Схема функціонування програмного забезпечення для розпізнавання об'єктів

3.3 Підготовка вхідних даних для програмного забезпечення для розпізнавання об'єктів

Як зазначено вище та відображено на рис.3.2, важливим етапом при створенні розпізнавальної моделі для програмного забезпечення для розпізнавання об'єктів є підготовка вхідних даних.

На цьому етапі спочатку необхідно зібрати велику кількість зображень або відео, що містять об'єкти, які потрібно розпізнати. Дані можуть бути зібрані з різних джерел, таких як камери, Інтернет ресурси, або існуючі бази даних [20]-[22].

Після збору даних їх потрібно анотувати, тобто позначити області з об'єктами на зображеннях. Це може включати створення рамок (bounding boxes), масок для сегментації або інших міток, що визначають розташування і тип об'єктів [20]-[22].

Далі необхідно перевірити і очистити дані від шуму або помилок. Це може включати видалення зображень низької якості, що не містять корисної інформації, або виправлення анотацій, що мають помилки [20]-[22].

Дані слід розділити на три основні набори: навчальний (training), валідаційний (validation) та тестовий (test) набори. Навчальний набір використовується для тренування моделей, валідаційний для налаштування гіперпараметрів і перевірки моделі на нових даних, тестовий використовується для оцінки ефективності моделі [20]-[22].

Для покращення якості моделі і уникнення неякісних результатів навчання моделі можна використовувати техніки перетворення даних, такі як обертання, масштабування, зміна яскравості або контрасту зображень. Це допомагає моделі краще справлятися з варіаціями в даних [20]-[22].

Дані можуть потребувати нормалізації (наприклад, масштабування пікселів до діапазону від 0 до 1) або інших видів попередньої обробки, щоб забезпечити стабільність і ефективність навчання моделі [20]-[22].

Регулярна перевірка якості даних і їх відповідності вимогам моделі є важливою для забезпечення надійних результатів. Ці етапи допомагають створити якісний і репрезентативний набір даних, необхідний для успішного навчання і тестування систем розпізнавання об'єктів [20]-[22].

Для навчання розпізнавальних моделей на основі нейронних мереж у задачі виявлення об'єктів на зображеннях можна використати спеціалізований набір даних. Якість моделей залежить від розміру набору даних: чим більше даних, тим точніше результати роботи моделі. Як приклад таких даних можна використати ImageNet, який містить 400 тисяч зображень з двохста класів, з 350 000 розміченими рамками, що обмежують об'єкти. Іншим набором даних може бути Google Open Images, що включає більше мільйона зображень, на яких розмічено приблизно 16 мільйонів рамок. Набір даних MS COCO складається зі 150 тисяч зображень, розділених на вісімдесят класів, з відповідними рамками.

Для навчання нейронної мережі було використано набір даних CIFAR-10 (Canadian Institute for Advanced Research, 10 classes) [23], з якого було виділено зображення, що містили об'єкти двох класів – клас літаків та клас автомобілів [23],

Приклад зображень, що використовувалися як вхідні дані для навчання нейромережевої розпізнавальної моделі, наведено на рис. 3.3.

Для навчання та тестування моделей базовий набір даних необхідно підготувати для його використання, зокрема, необхідно виконати виділення об'єктів на зображеннях за допомогою рамок і додавання відповідних міток (провести процедуру анотування даних). Це ручне виділення необхідне для того, щоб моделі могли навчитися класифікувати об'єкти в межах визначених рамок.

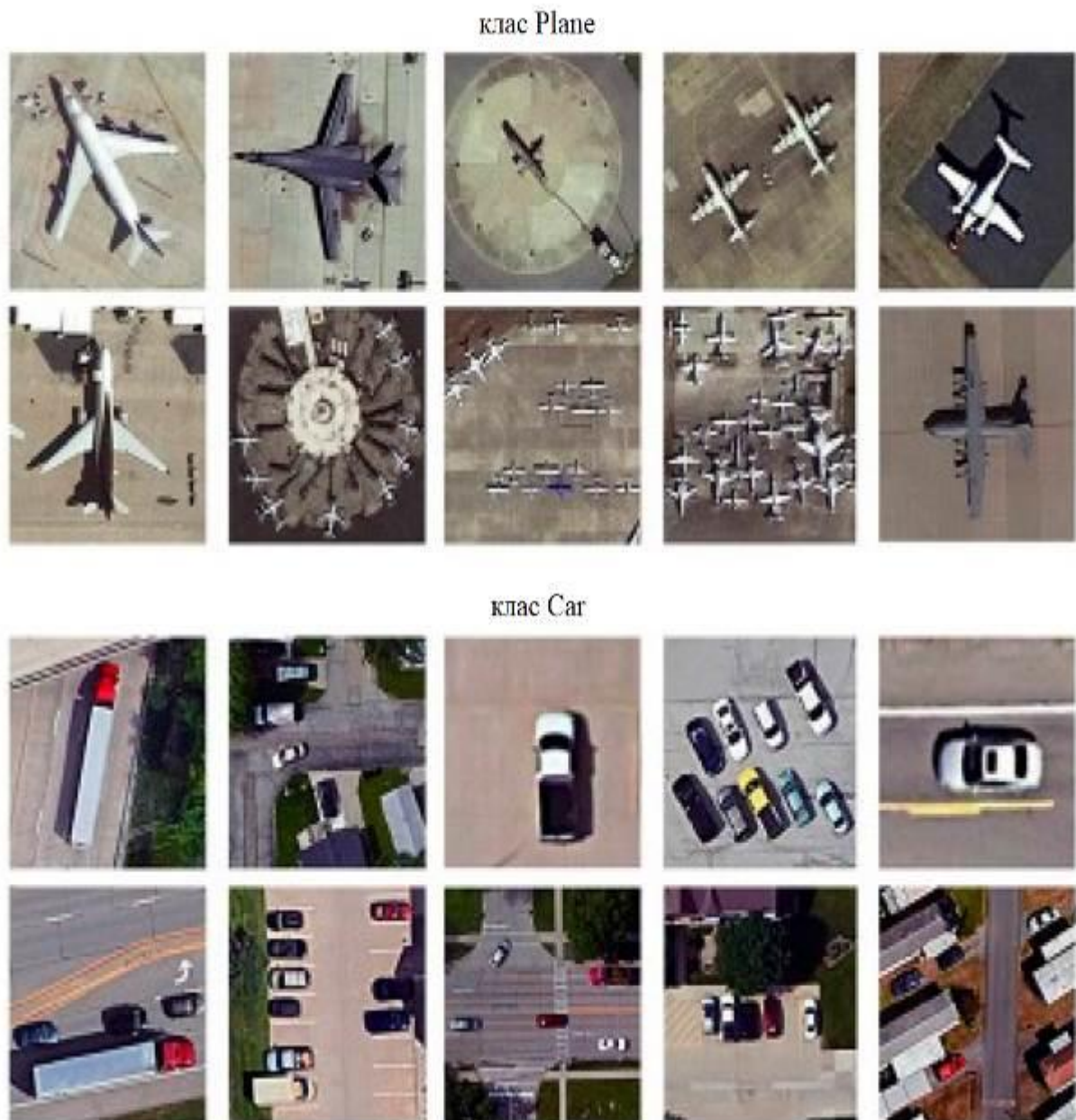


Рисунок 3.3 – Набір вхідних даних для навчання нейромережевої
розпізнавальної моделі

Для анотування об'єктів на зображеннях використовується утиліта LabelImg. Після її установки та запуску відкривається вікно (рис. 3.4), де потрібно вручну виділити об'єкти на кожному зображенні. Після завершення цього процесу зберігаються XML-файли, які мають таке ж ім'я, як і відповідні зображення, і поміщаються в одну папку.

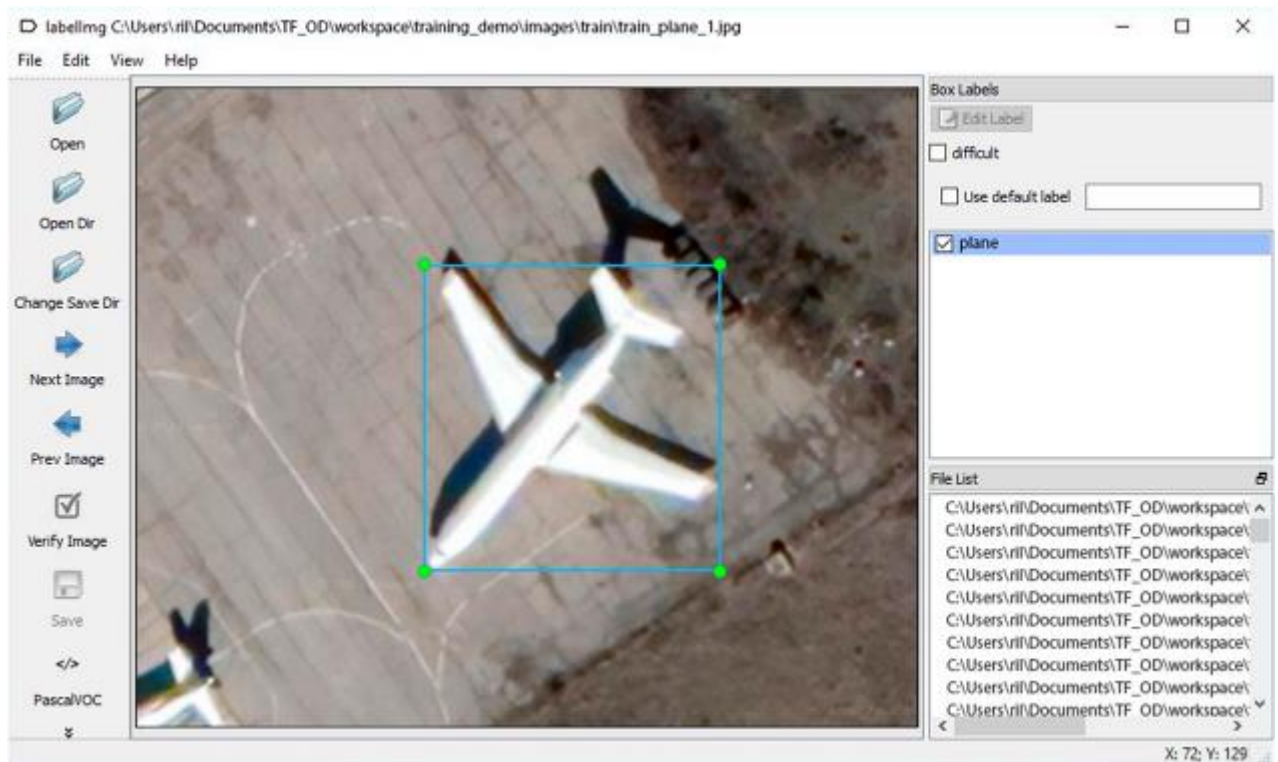


Рисунок 3.4 – Анотація зображень у процесі підготовки вхідних даних

Після підготовки вхідних даних їх можна використовувати для синтезу розпізнавальних моделей на основі нейромережових технологій.

3.4 Опис процесу синтезу розпізнавальних моделей при створенні програми для розпізнавання об'єктів

Процес синтезу розпізнавальних моделей для програми розпізнавання об'єктів охоплює кілька важливих етапів. Спочатку здійснюється збір та підготовка даних: зображення, що містять об'єкти, які потрібно розпізнати, анотуюються шляхом виділення об'єктів і додавання міток або рамок, а також очищаються від шуму і попередньо обробляються для покращення якості навчання. Після цього вибирається відповідна архітектура нейронної мережі, така як SSD, Faster R-CNN або YOLO, і налаштовуються її параметри, включаючи кількість шарів та розміри фільтрів, для досягнення найкращих результатів [24]-[26].

На етапі навчання нейромережева модель тренується на підготовлених даних за допомогою алгоритмів оптимізації, що включає обробку навчальних зображень і коригування ваг нейронної мережі для мінімізації помилок. Паралельно проводиться валідація, щоб перевірити, як добре модель узагальнює знання на нових, не бачених раніше даних. Після завершення навчання модель тестується на окремому тестовому наборі даних, який не використовувався під час навчання, для оцінки її точності і ефективності. Оцінюються різні показники якості нейромережевої моделі, такі як точність, повнота та інші метрики, щоб визначити, наскільки добре модель виконує завдання розпізнавання об'єктів [24]-[26].

На основі результатів тестування можуть бути внесені зміни в архітектуру нейромережевої моделі або процес навчання для покращення результатів. Для підвищення стійкості моделі до різних умов можуть використовуватися техніки перетворення даних, такі як обертання, масштабування та зміна яскравості зображень. Після цього модель інтегрується в програмне забезпечення для розпізнавання об'єктів, де вона буде використовуватися для реального розпізнавання об'єктів. Важливим етапом є тестування моделі в реальних умовах, щоб перевірити її функціональність у практичних сценаріях і здійснити додаткові налаштування при необхідності [24]-[26].

Останній етап включає постійний моніторинг роботи моделі у програмному забезпеченні для розпізнавання об'єктів для виявлення можливих проблем або зниження точності, а також регулярне оновлення моделі з новими даними або вдосконаленням архітектури. Це забезпечує підтримання її ефективності і точності, а також адаптацію до нових умов і вимог, що може виникнути в процесі її використання [24]-[26].

У процесі розробки програмного забезпечення для розпізнавання об'єктів після ініціалізації та попередньої обробки даних було виконано навчання та тестування нейромережевих моделей. Моделі вибиралися з

бібліотеки TensorFlow 2 Detection Model Zoo [27] (рис. 3.5) та зберігалися в папці pre-trained-models.

TensorFlow 2 Detection Model Zoo

TensorFlow 2.2 Python 3.6

We provide a collection of detection models pre-trained on the [COCO 2017 dataset](#). These models can be useful for out-of-the-box inference if you are interested in categories already in those datasets. You can try it in our inference [colab](#).

They are also useful for initializing your models when training on novel datasets. You can try this out on our few-shot training [colab](#).

Please look at [this guide](#) for mobile inference.

Finally, if you would like to train these models from scratch, you can find the model configs in this [directory](#) (also in the linked `tar.gz`).

Model name	Speed (ms)	COCO mAP	Outputs
CenterNet HourGlass104 512x512	70	41.9	Boxes
CenterNet HourGlass104 Keypoints 512x512	76	40.0/61.4	Boxes/Keypoints
CenterNet HourGlass104 1024x1024	197	44.5	Boxes
CenterNet HourGlass104 Keypoints 1024x1024	211	42.8/64.5	Boxes/Keypoints
CenterNet Resnet50 V1 FPN 512x512	27	31.2	Boxes
CenterNet Resnet50 V1 FPN Keypoints 512x512	30	29.3/50.7	Boxes/Keypoints
CenterNet Resnet101 V1 FPN 512x512	34	34.2	Boxes
CenterNet Resnet50 V2 512x512	27	29.5	Boxes
CenterNet Resnet50 V2 Keypoints 512x512	30	27.6/48.2	Boxes/Keypoints
CenterNet MobileNetV2 FPN 512x512	6	23.4	Boxes
CenterNet MobileNetV2 FPN Keypoints 512x512	6	41.7	Keypoints
EfficientDet D0 512x512	39	33.6	Boxes
EfficientDet D1 640x640	54	38.4	Boxes

Рисунок 3.5 – Моделі бібліотеки TensorFlow 2 Detection Model Zoo

Для використання у програмному забезпеченні для розпізнавання об'єктів було обрано, зокрема, такі моделі:

- SSD MobileNet v2 320x320;
- SSD ResNet50 V1 FPN 640x640;
- Faster R-CNN ResNet101 V1 640x640;
- Faster R-CNN ResNet152 V1 640x640.

Оцінку ефективності навчених нейромережевих розпізнавальних моделей наведено на рис. 3.6.

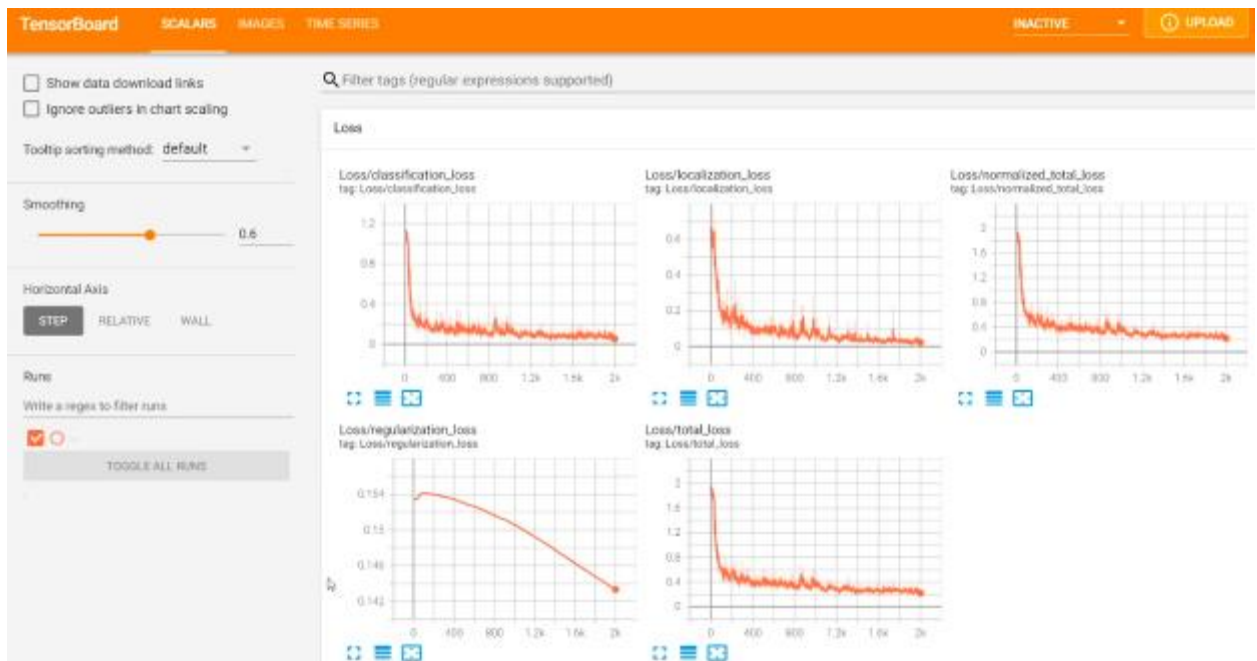


Рисунок 3.6 – Результати оцінювання ефективності нейромережових моделей у процесі їх синтезу

Графіки, наведені на рис. 3.6, показують, що з кожною епохою помилка розпізнавання поступово зменшується, наближаючись до нуля, що свідчить про успішне навчання моделі та її готовність до використання для розпізнавання об'єктів на зображеннях.

Після синтезу розпізнавальних нейромережових моделей навчені моделі було експортовано (збережено у вигляді відповідних файлів) для подальшого використання в програмі для розпізнавання об'єктів на зображеннях.

3.5 Висновки за розділом 3

Розроблено програмне забезпечення для розпізнавання об'єктів.

Запропоновано структуру програмного забезпечення для підтримки процесу розпізнавання об'єктів. Визначено, що програмне забезпечення для розпізнавання об'єктів складається з трьох компонентів. Програмні модулі для синтезу розпізнавальних моделей призначені для створення моделей на основі

нейромережових технологій. Інтерфейсна частина програмного забезпечення для розпізнавання об'єктів забезпечує взаємодію користувача з системою (програма забезпечує взаємодію з користувачем через консоль та стандартні термінальні засоби операційної системи). Програмні модулі для роботи з наборами даних забезпечують завантаження файлів з даними, що містять зображення для обробки, а також засоби попередньої обробки даних та формування вибірок для синтезу розпізнавальних моделей.

Описано процеси підготовки вхідних даних та синтезу розпізнавальних моделей при створенні програмного забезпечення для розпізнавання об'єктів.

Описано особливості реалізації програмного забезпечення для розпізнавання об'єктів.

4 ЕКСПЛУАТАЦІЯ, ТЕСТУВАННЯ ТА ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ ПРОГРАМИ

4.1 Призначення й умови застосування програми

Програма призначена для підтримки процесу розпізнавання об'єктів. Програма написана на мові Python, яка має великий набір бібліотек та фреймворків для розпізнавання об'єктів і машинного навчання, що спрощують реалізацію складних алгоритмів і моделювання, а також забезпечують високий рівень абстракції для швидкого розроблення прототипів програмного забезпечення. В якості середовища розробки для створення програмного забезпечення для розпізнавання об'єктів обрано Sublime Text, що є зручним редактором та дозволяє легко редагувати файли без значного споживання ресурсів системи.

Програмне забезпечення для підтримки процесу розпізнавання об'єктів забезпечує виконання таких функцій:

- підтримка можливості завантаження зображення для ідентифікації об'єктів;
- підтримка можливості пакетного завантаження набору зображень для виявлення об'єктів на них;
- можливість вибору розпізнавальної моделі для ідентифікації об'єктів;
- можливість розпізнавання об'єктів на зображеннях;
- можливість збереження вихідних даних з розпізнаними об'єктами.

Для експлуатації програмного забезпечення для підтримки процесу розпізнавання об'єктів необхідно таке апаратне та програмне забезпечення:

- оперативна пам'ять об'ємом чотири гігабайти або вище;
- не менше 4 Гб вільного простору на запам'ятовуючому пристрою апаратного засобу;
- процесор із тактовою частотою 2 ГГц або вище;
- графічний адаптер;

– встановлена операційна система.

Функціональні характеристики розробленого програмного забезпечення для розпізнавання об'єктів наведено у технічному завданні (додаток А). Фрагмент тексту програмного забезпечення для розпізнавання об'єктів наведено у додатку Б.

4.2 Характеристики програми для розпізнавання об'єктів

Програмне забезпечення для розпізнавання об'єктів складається з трьох компонентів. Програмні модулі для синтезу розпізнавальних моделей призначені для створення моделей на основі нейромережевих технологій. Синтезовані моделі у подальшому можуть використовуватися для розпізнавання об'єктів, подібним до тих, що використовувалися у процесі синтезу відповідної моделі.

Частина з інтерфейсом користувача забезпечує взаємодію користувача з системою. Враховуючи те, що розроблену програму у майбутньому можна використовувати як частину більш складної системи, прийнято рішення про використання як базису консольного режиму взаємодії. Таким чином, програма забезпечує взаємодію з користувачем через консоль та стандартні термінальні засоби операційної системи.

Програмні модулі для роботи з наборами даних забезпечують завантаження файлів з даними, що містять зображення для обробки, а також засоби попередньої обробки даних та формування вибірок для синтезу розпізнавальних моделей. Ці ж модулі забезпечують збереження результатів розпізнавання об'єктів у вихідних файлах.

4.3 Інструкція по експлуатації програми

4.3.1 Звернення до програми

Для запуску програмного забезпечення для розпізнавання об'єктів необхідно запустити відповідний ру-файл.

4.3.2 Вхідні й вихідні дані

Вхідними даними до програмного забезпечення для розпізнавання об'єктів є зображення або набір зображень, на яких необхідно виявити об'єкти, а також розпізнавальна модель, яка використовується для ідентифікації об'єктів.

Вихідними даними програмного забезпечення для розпізнавання об'єктів є виявлені на зображеннях об'єкти.

4.3.3 Повідомлення

Користувач програмного забезпечення для підтримки процесу розпізнавання об'єктів може отримати різні повідомлення, зокрема, повідомлення про завершення процесу розпізнавання, повідомлення про некоректний формат зображення, повідомлення про некоректний шлях до зображення, повідомлення про неможливість виконання операції розпізнавання об'єктів та ін.

4.4 Виконання програмного забезпечення для підтримки процесу розпізнавання об'єктів

Програма забезпечує взаємодію з користувачем через консоль та стандартні термінальні засоби операційної системи. Після запуску програмного забезпечення для підтримки процесу розпізнавання об'єктів відображається головне меню (рис. 4.1), де користувач шляхом введення відповідної цифри може запустити відповідну функцію програми (вибір зображення або набору зображень для розпізнавання об'єктів; вибір розпізнавальної моделі та ін.).

```

-----Object Detection-----
1 - choose model (chosen: SSD ResNet50 V1 FPN 640x640).
2 - select a photo for detection...
3 - select a folder with images to detect objects...
Enter task number:

```

Рисунок 4.1 – Головне меню програмного забезпечення
для розпізнавання об'єктів

Після вибору номеру відповідної функції програми користувачу аналогічним чином необхідно ввести додаткові параметри, наприклад, ввести номер, що відповідає розпізнавальній моделі (рис. 4.2).

```

-----Object Detection-----
1 - choose model (chosen: SSD ResNet50 V1 FPN 640x640).
2 - select a photo for detection...
3 - select a folder with images to detect objects...
Enter task number: 1

-----Choose model-----
1 - SSD MobileNet v2 320x320
2 - SSD ResNet50 V1 FPN 640x640
3 - Faster R-CNN ResNet101 V1 640x640
4 - Faster R-CNN ResNet152 V1 640x640
Enter model number:

```

Рисунок 4.2 – Вибір типу розпізнавальної моделі

При виборі пункту меню стосовно завантаження зображення або набору зображень, користувач, використовуючи стандартні засоби операційної системи, повинен вказати шлях до зображення або папки з зображеннями для подальшого розпізнавання об'єктів (рис. 4.3).

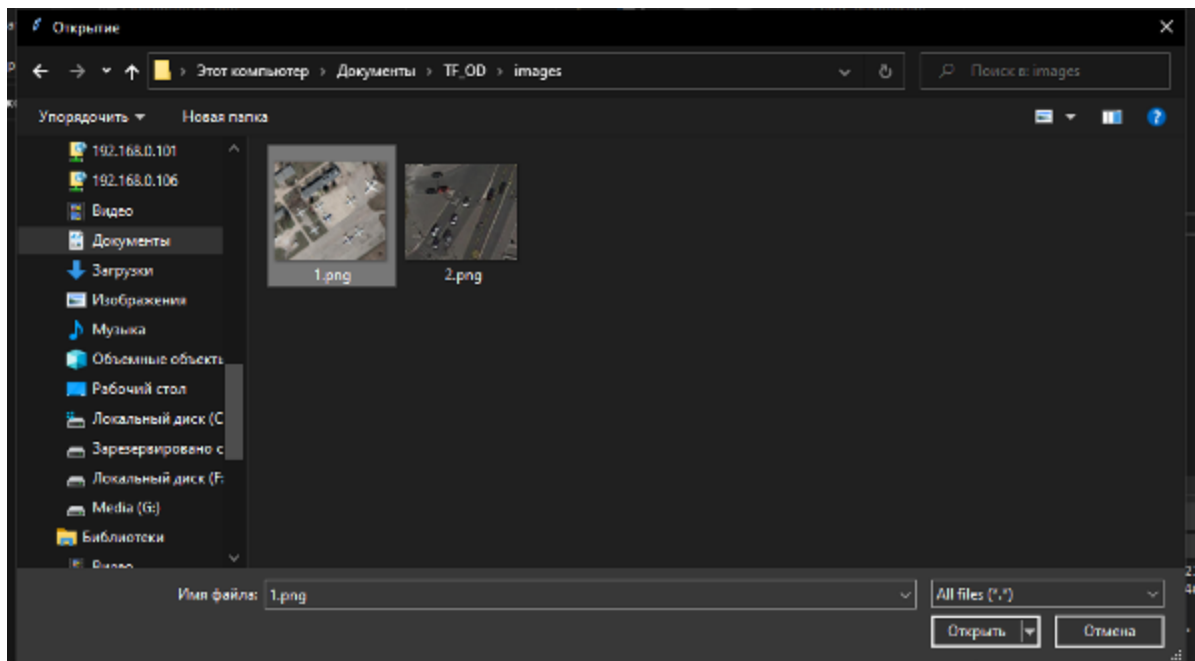


Рисунок 4.3 – Вибір зображення для розпізнавання

Після розпізнавання об'єктів програма зберігає зображення з розпізнаними об'єктами у відповідному файлі. На вихідному зображенні виділяються в рамки розпізнані об'єкти. Над рамками підписується клас об'єкту та відсоток ймовірності віднесення до відповідного класу. На рис. 4.4 наведено результати розпізнавання літаків на зображенні.



Рисунок 4.4 – Результати розпізнавання літаків на зображенні

На рис. 4.5 наведено результати розпізнавання автомобілів на зображенні.

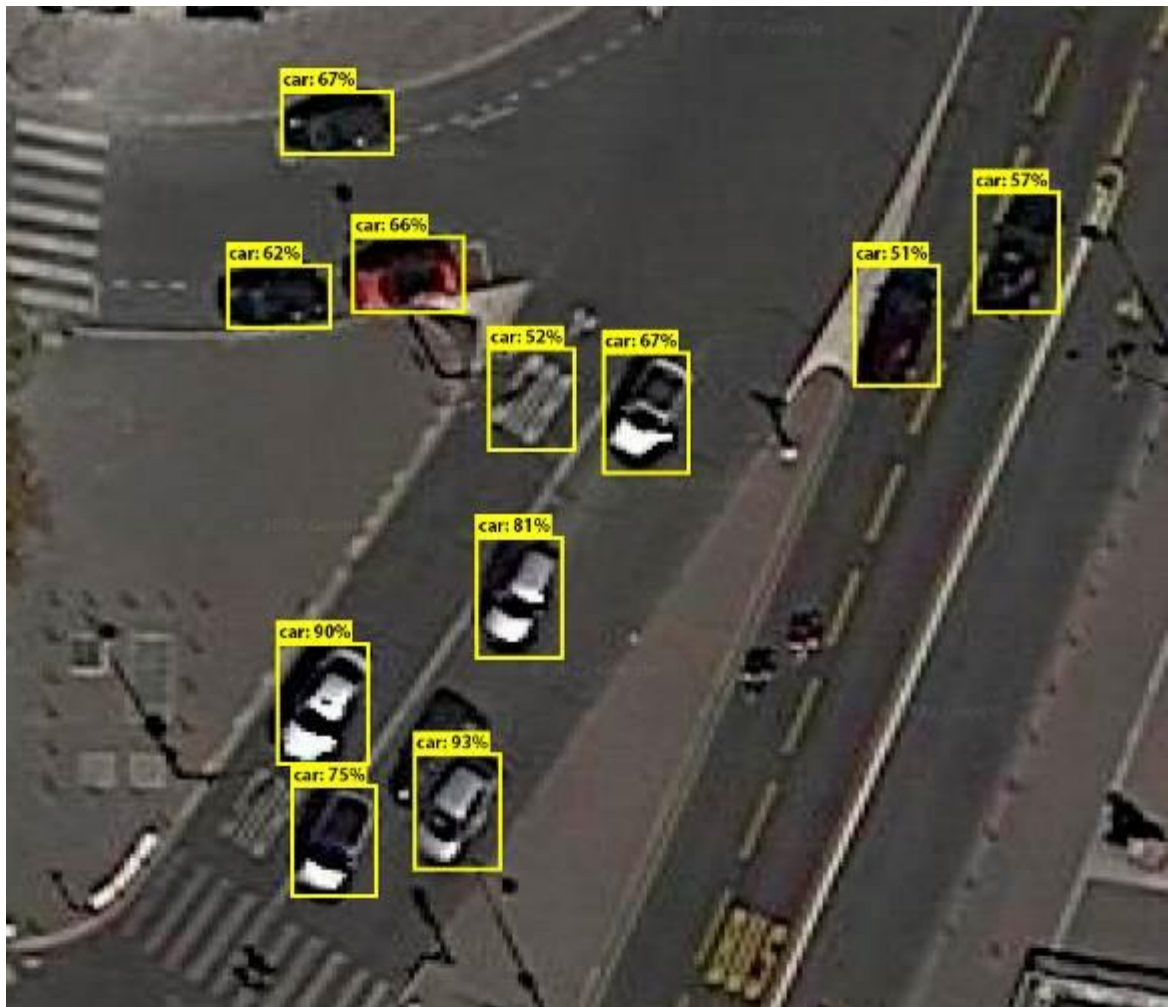


Рисунок 4.5 – Результати розпізнавання автомобілів на зображенні

4.5 Тестування програмного забезпечення для підтримки процесу розпізнавання об'єктів

Виконано тестування програмного забезпечення для підтримки процесу розпізнавання об'єктів.

Виявлено, що програма для підтримки процесу розпізнавання об'єктів функціонує правильно та злагоджено. Вона реалізує всі функціональні вимоги та успішно виконує свою основну задачу підтримки процесу розпізнавання об'єктів.

Розроблене програмне забезпечення дозволяє забезпечувати автоматизацію процесів, пов'язаних з підтримкою процесу розпізнавання об'єктів.

4.6 Висновки за розділом 4

Описано програмне забезпечення для підтримки процесу розпізнавання об'єктів. Виконано тестування розробленого програмного забезпечення для розпізнавання об'єктів. Результати тестування програмного забезпечення показали, що розроблена програма дозволяє забезпечити автоматизацію процесів, пов'язаних з підтримкою процесу розпізнавання об'єктів.

ВИСНОВКИ

В ході виконання дипломної кваліфікаційної роботи магістра було проаналізовано та досліджено процес розробки програмного забезпечення для розпізнавання об'єктів.

Визначено, що розпізнавання об'єктів є процесом, за допомогою якого програмне забезпечення виявляє шаблони в даних і використовує їх для прогнозування або класифікації за допомогою обчислювальних алгоритмів. Системи розпізнавання образів використовують комп'ютерні алгоритми для аналізу та інтерпретації різних типів даних, вхідні дані можуть бути представлені у формі тексту, зображень або звуків. Проаналізовано процес розробки програмних систем розпізнавання образів. Проаналізовано компоненти та принципи розробки програмних систем розпізнавання образів. Проаналізовано приклади застосувань програмних засобів для розпізнавання об'єктів. Відзначено, що одним з найбільш поширених варіантів використання програмних засобів розпізнавання образів є розпізнавання зображень. Відзначено, що розпізнавання образів за допомогою нейронних мереж є сьогодні широко використовуваним та ефективним підходом. Нейронні мережі ґрунтуються на паралельних структурах, які імітують процес прийняття рішень людиною. Показано, що жоден алгоритм не забезпечує оптимальні результати у всіх випадках. Тому доцільно розробити програмне забезпечення, що дозволяє використовувати різні моделі на основі штучних нейронних мереж для розпізнавання образів, що дозволить покращити ефективність програмних систем розпізнавання образів.

За результатами проведеного аналізу зроблено висновок, що у наш час існує досить багато програмних засобів для розпізнавання об'єктів. Проте деякі програмні засоби для розпізнавання об'єктів можуть неправильно ідентифікувати об'єкти. Це може статися через використання неефективних алгоритмів розпізнавання образів, або обмеження у використовуваних

алгоритмах, недостатню якість зображення або складність середовища. Інші програмні засоби для розпізнавання об'єктів можуть вимагати значної обчислювальної потужності і великої кількості пам'яті, що може бути витратним і обмежити використання в деяких умовах. Впровадження та підтримка інших програмних систем розпізнавання об'єктів можуть бути дорогими, особливо для малих і середніх підприємств. Тому актуальною є розробка програмного забезпечення для розпізнавання об'єктів.

Сформульовано функціональні вимоги до програмного забезпечення для розпізнавання об'єктів.

Для реалізації програмного забезпечення для підтримки процесу розпізнавання об'єктів обрано мову програмування Python, яка має великий набір бібліотек та фреймворків для розпізнавання об'єктів і машинного навчання, що спрощують реалізацію складних алгоритмів і моделювання, а також забезпечують високий рівень абстракції для швидкого розроблення прототипів програмного забезпечення. Крім того, Python легко інтегрується з іншими мовами програмування і технологіями, такими як C++ для високопродуктивних обчислень або JavaScript для розроблення вебінтерфейсів. Це дозволяє створювати комплексні рішення, які можуть включати різні компоненти і технології.

Для створення програмного забезпечення для розпізнавання об'єктів обрано середовище розробки Sublime Text, що є зручним редактором та дозволяє легко редагувати файли без значного споживання ресурсів системи. Це особливо корисно при роботі з великими наборами програмних кодів або проектами, що включають велику кількість файлів. Крім того, Sublime Text підтримує велику кількість плагінів і налаштувань, які можна адаптувати під специфічні потреби проекту. Це дозволяє розширювати функціональність редактора, наприклад, для підтримки специфічних синтаксичних підсвіток або інтеграції з системами контролю версій.

Новизна роботи полягає у тому, що створено модель взаємодії користувача з програмним забезпеченням для розпізнавання об'єктів, яка

подана у вигляді діаграми прецедентів, та дозволяє розроблювати програмне забезпечення для розпізнавання об'єктів.

Практична цінність результатів роботи полягає у тому, що розроблено програмне забезпечення для розпізнавання об'єктів.

Запропоновано структуру програмного забезпечення для підтримки процесу розпізнавання об'єктів. Визначено, що програмне забезпечення для розпізнавання об'єктів складається з трьох компонентів. Програмні модулі для синтезу розпізнавальних моделей призначені для створення моделей на основі нейромережових технологій. Інтерфейсна частина програмного забезпечення для розпізнавання об'єктів забезпечує взаємодію користувача з системою (програма забезпечує взаємодію з користувачем через консоль та стандартні термінальні засоби операційної системи). Програмні модулі для роботи з наборами даних забезпечують завантаження файлів з даними, що містять зображення для обробки, а також засоби попередньої обробки даних та формування вибірок для синтезу розпізнавальних моделей.

Описано процеси підготовки вхідних даних та синтезу розпізнавальних моделей при створенні програмного забезпечення для розпізнавання об'єктів.

Описано особливості реалізації програмного забезпечення для розпізнавання об'єктів.

Виконано тестування розробленого програмного забезпечення для розпізнавання об'єктів. Результати тестування програмного забезпечення показали, що розроблена програма дозволяє забезпечити автоматизацію процесів, пов'язаних з підтримкою процесу розпізнавання об'єктів.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. What is Pattern Recognition? A Gentle Introduction [Electronic resource]. – Access mode: <https://viso.ai/deep-learning/pattern-recognition/>.
2. Pattern Recognition. Introduction [Electronic resource]. – Access mode: <https://www.geeksforgeeks.org/pattern-recognition-introduction/>.
3. Pattern Recognition in AI: Technologies and Applications [Electronic resource]. – Access mode: <https://libraria.ai/blog/pattern-recognition-in-ai-technologies-and-applications/>.
4. Breaking Down Pattern Recognition: How We Teach Machines to See and Understand [Electronic resource]. – Access mode: <https://pareto.ai/blog/pattern-recognition-in-machine-learning>.
5. Pattern Recognition in Machine Learning [Electronic resource]. – Access mode: <https://www.v7labs.com/blog/pattern-recognition-guide>.
6. Pattern Recognition: Unlocking the Power of AI [Electronic resource]. – Access mode: <https://machinemindmatters.com/posts/pattern-recognition-unlocking-power-ai/>.
7. Types of Pattern Recognition Algorithms [Electronic resource]. – Access mode: <https://www.globaltechcouncil.org/machine-learning/types-of-pattern-recognition-algorithms/>.
8. Best AI Image Recognition Software [Electronic resource]. – Access mode: <https://fritz.ai/best-ai-image-recognition-software/>.
9. Best Image Recognition Software [Electronic resource]. – Access mode: <https://www.g2.com/categories/image-recognition>.
10. Try The 13 Best Image Recognition Tools [Electronic resource]. – Access mode: <https://www.talkwalker.com/blog/best-image-recognition-tools>.
11. Top 9 Best Image Recognition Software [Electronic resource]. – Access mode: <https://fixthephoto.com/best-image-recognition-software.html>.
12. Clarifai [Electronic resource]. – Access mode: <https://www.clarifai.com/>.

13. Imagga [Electronic resource]. – Access mode: <https://imagga.com/>.
14. Hive [Electronic resource]. – Access mode: <https://thehive.ai/>.
15. Python documentation [Electronic resource]. – Access mode: https://tulip.labri.fr/Documentation/3_8/tulip-python/html/gettingstarted.html.
16. Python pour la data science [Electronic resource]. – Access mode: <https://pythonds.linogaliana.fr/>.
17. Sublime Text [Electronic resource]. – Access mode: <https://sublime-text.en.softonic.com/>.
18. Sublime Text : Un éditeur de texte et de code multiplateforme pour des projets performants [Electronic resource]. – Access mode: <https://hubfi.fr/sublime-text/>.
19. Tensorflow [Electronic resource]. – Access mode: <https://www.tensorflow.org/>.
20. What is data preparation? An in-depth guide [Electronic resource]. – Access mode: <https://www.techtarget.com/searchbusinessanalytics/definition/data-preparation>.
21. What Is Data Preparation? Steps For Effective Data Prep [Electronic resource]. – Access mode: <https://www.astera.com/type/blog/data-preparation/>.
22. How to Prepare your Data [Electronic resource]. – Access mode: <https://towardsdatascience.com/the-basics-of-data-prep-7bb5f3af77ac>.
23. CIFAR-10 (Canadian Institute for Advanced Research, 10 classes) [Electronic resource]. – Access mode: <https://paperswithcode.com/dataset/cifar-10>.
24. How Does a Neural Network Recognize Images [Electronic resource]. – Access mode: <https://www.baeldung.com/cs/neural-networks-image-recognition/>.
25. Image Recognition with Deep Neural Networks and its Use Cases [Electronic resource]. – Access mode: <https://www.altexsoft.com/blog/image-recognition-neural-networks-use-cases/>.
26. An Analysis Of Convolutional Neural Networks For Image Classification [Electronic resource]. – Access mode: <https://www.sciencedirect.com/science/article/pii/S1877050918309335>.

27. TensorFlow 2 Detection Model Zoo [Electronic resource]. – Access mode:

https://github.com/tensorflow/models/blob/master/research/object_detection/g3doc/tf2_detection_zoo.md.

ДОДАТОК А
Технічне завдання

Вступ

Програмне забезпечення може використовуватися для підтримки процесу розпізнавання об'єктів.

A.1 Підстава для розробки

Підставою для розробки є завдання на дипломну кваліфікаційну роботу на тему «Дослідження та програмна реалізація нейромережових засобів розпізнавання об'єктів», затверджене наказом Національного університету «Запорізька політехніка» № 398 від 07 жовтня 2024 р.

A.2 Призначення розробки

Програмний продукт призначений для розпізнавання об'єктів.

A.3 Основні вимоги до програми, що розробляється

A.3.1 Вимоги до функціональних характеристик

Програмне забезпечення для підтримки процесу розпізнавання об'єктів забезпечує виконання таких функцій:

- підтримка можливості завантаження зображення для ідентифікації об'єктів;
- підтримка можливості пакетного завантаження набору зображень для виявлення об'єктів на них;
- можливість вибору розпізнавальної моделі для ідентифікації об'єктів;
- можливість розпізнавання об'єктів на зображеннях;
- можливість збереження вихідних даних з розпізнаними об'єктами.

А.3.2 Вимоги до інтерфейсу програми

Інтерфейс програмного забезпечення для підтримки процесу розпізнавання об'єктів може бути організований у термінальному (консольному) режимі.

А.3.3 Вимоги до надійності

Програмне забезпечення для розпізнавання об'єктів повинно забезпечити надійне функціонування.

А.3.4 Умови експлуатації

Для експлуатації програмного забезпечення для розпізнавання об'єктів необхідна наявність персонального комп'ютера.

А.3.5 Вимоги до складу та параметрів технічний засобів

Для експлуатації програмного забезпечення для розпізнавання об'єктів необхідно таке апаратне та програмне забезпечення:

- оперативна пам'ять об'ємом чотири гігабайти або вище;
- не менше 4 Гб вільного простору на запам'ятовуючому пристрою апаратного засобу;
- процесор із тактовою частотою 2 ГГц або вище;
- графічний адаптер;
- встановлена операційна система.

A.3.6 Вимоги до маркування і пакування

Програма для підтримки процесу розпізнавання об'єктів може бути записана на будь-якому носії інформації.

На пакуванні повинна бути назва програми – «Програмне забезпечення для розпізнавання об'єктів».

A.4 Вхідні дані до роботи

Вхідними даними до програмного забезпечення для розпізнавання об'єктів є зображення або набір зображень, на яких необхідно виявити об'єкти, а також розпізнавальна модель, яка використовується для ідентифікації об'єктів.

Вихідними даними програмного забезпечення для розпізнавання об'єктів є виявлені на зображеннях об'єкти.

ДОДАТОК Б
Фрагмент тексту програми

```

lbls = ['litak', 'avto']
kilkimg = 2

IMGSSHLH = os.shlh.join('TF_OD', 'workspace', 'tringdm',
'imgs')

if not os.shlh.exists(IMGSSHLH):
    if os.name == 'posix':
        !mkdir -p {IMGSSHLH}
    if os.name == 'nt':
        !mkdir {IMGSSHLH}
for lbl in lbls:
    shlh = os.shlh.join(IMGSSHLH, lbl)
    if not os.shlh.exists(shlh):
        !mkdir {shlh}

for lbl in lbls:
    kap = gtf.VideoKapture(0)
    print('Zbir imgs for {}'.format(lbl))
    time.sleep(5)
    for imgnm in range(kilkimg):
        print('Zbir img {}'.format(imgnm))
        rtt, ramk = kap.read()
        imgnm =
os.shlh.join(IMGSSHLH, lbl, lbl+'.'+'{}.jpg'.format(str(uuid.uuid1()))))
        gtf.imwrite(imgnm, ramk)
        gtf.imshow('ramk', ramk)
        time.sleep(2)

```

```

        if gtf.waitKey(1) & 0xFF == ord('q'):
            break
    kap.release()
    gtf.destroyAllWindows()

    LBLIMG_SHLH = os.shlh.join('TF_OD', 'lblimg')

    if os.name == 'posix':
        !make qt5py3
    if os.name == 'nt':
        !cd {LBLIMG_SHLH} && prcc -o libs/rescs.py rescs.qrc

    !cd {LBLIMG_SHLH} && python lblImg.py

    TRAIN_SHLH      =      os.shlh.join('TF_OD',      'workspace',
    'tringdm', 'imgs', 'train')
    TEST_SHLH       =      os.shlh.join('TF_OD',      'workspace',
    'tringdm', 'imgs', 'test')
    ARCH_SHLH       =      os.shlh.join('TF_OD',      'workspace',
    'tringdm', 'imgs', 'arch.tar.gz')

    !tar -czf {ARCH_SHLH} {TRAIN_SHLH} {TEST_SHLH}

    CUSTOMMDLNAME = 'mysdmobnet'
    PRTRAINED_MDLNM = 'sdmobilentv2_fpnlit'
    PRTRAINED_MDL_URL =
    'http://download.tensorflow.org/MDLs/object_detectn/tf2/
    20200711/sdmobilentv2_fpnlit.tar.gz'
    TFRCDSCRPTNM = 'genrattfrcord.py'

```

```
LBLMAPNM = 'LBLmap.pbt.txt'
```

```
shlhs = {
    'WORKSPACE_SHLH':          os.shlh.join('Tensorflow',
    'workspace'),
    'SCRIPTS_SHLH':
os.shlh.join('Tensorflow', 'scripts'),
    'APIMDL_SHLH': os.shlh.join('Tensorflow', 'MDLs'),
    'ANNOTSHLH':          os.shlh.join('Tensorflow',
    'workspace', 'annotations'),
    'IMG_SHLH':          os.shlh.join('Tensorflow',
    'workspace', 'imgs'),
    'MDL_SHLH':          os.shlh.join('Tensorflow',
    'workspace', 'MDLs'),
    'PRTRAINED_MDL_SHLH':      os.shlh.join('Tensorflow',
    'workspace', 'pre-trained-MDLs'),
    'CHECKPNTSHLH':          os.shlh.join('Tensorflow',
    'workspace', 'MDLs', CUSTOMMDLNAME),
    'OUTPUT_SHLH':          os.shlh.join('Tensorflow',
    'workspace', 'MDLs', CUSTOMMDLNAME, 'export'),
    'TFJS_SHLH':os.shlh.join('Tensorflow',
    'workspace', 'MDLs', CUSTOMMDLNAME, 'tfjsexport'),
    'TFLITE_SHLH':os.shlh.join('Tensorflow',
    'workspace', 'MDLs', CUSTOMMDLNAME, 'tfliteexport'),
    'PROTOC_SHLH':os.shlh.join('Tensorflow', 'protoc')
}
```

```
files = {
    'PIPELNEKNFG':os.shlh.join('Tensorflow',
    'workspace', 'MDLs', CUSTOMMDLNAME, 'pipln.knfg'),
```

```

    'TFRCRDSCRIPT': os.shlh.join(shlhs['SCRIPTS_SHLH'],
TFRCRDSCRIPTNM),
    'LBLMAP': os.shlh.join(shlhs['ANNOTSHLH'], LBLMAPNM)
}

for shlh in shlhs.values():
    if not os.shlh.exists(shlh):
        if os.name == 'posix':
            !mkdir -p {shlh}
        if os.name == 'nt':
            !mkdir {shlh}

if os.name=='posix':
    !apt-get install protobuf-compiler
    !cd      Tensorflow/MDLs/research      &&      protoc
object_detectn/protos/*.proto  --python_out=.  &&  cp
object_detectn/packages/tf2/setup.py . && python -m pip
install .

import object_detectn

if os.name == 'posix':
    !wget {PRTTRAINED_MDL_URL}
    !mv                                     {PRTTRAINED_MDLNM+'.tar.gz'}
{shlhs['PRTTRAINED_MDL_SHLH']}
    !cd  {shlhs['PRTTRAINED_MDL_SHLH']}  &&  tar  -zxvf
{PRTTRAINED_MDLNM+'.tar.gz'}
if os.name == 'nt':
    wget.download(PRTTRAINED_MDL_URL)

```

```

!move                                {PRTTRAINED_MDLNM+'.tar.gz'}
{shlhs['PRTTRAINED_MDL_SHLH']}
!cd  {shlhs['PRTTRAINED_MDL_SHLH']}  &&  tar  -zxvf
{PRTTRAINED_MDLNM+'.tar.gz'}

```

```

lbls  =  [{ 'name':'litak',    'id':1},    { 'name':'avto',
'id':2}]

```

```

with open(files['LBLMAP'], 'w') as f:

```

```

    for lbl in lbls:
        f.write('itm { \n')
        f.write('\tnme:\'{}\'\n'.format(lbl['name']))
        f.write('\td:{}\n'.format(lbl['id']))
        f.write('}\n')

```

```

ARCH_FILES      =      os.shlh.join(shlhs['IMG_SHLH'],
'arch.tar.gz')

```

```

if os.shlh.exists(ARCH_FILES):

```

```

    !tar -zxvf {ARCH_FILES}

```

```

if not os.shlh.exists(files['TFRCRDSCRPT']):

```

```

    !git                clone                https://github.com/
{shlhs['SCRIPTS_SHLH']}

```

```

!python                {files['TFRCRDSCRPT']}                -x
{os.shlh.join(shlhs['IMG_SHLH'],        'train')}                -l
{files['LBLMAP']}      -o  {os.shlh.join(shlhs['ANNOTSHLH'],
'train.record')}

```

```

!python                {files['TFRCRDSCRPT']}                -x
{os.shlh.join(shlhs['IMG_SHLH'],        'test')}                -l

```

```

{files['LBLMAP']} -o {os.shlh.join(shlhs['ANNOTSHLH'],
'test.record')}

if os.name == 'posix':
    !cp {os.shlh.join(shlhs['PRTRAINED_MDL_SHLH'],
PRTRAINED_MDLNM, 'pipln.knfg')}
{os.shlh.join(shlhs['CHECKPNTSHLH'])}
if os.name == 'nt':
    !copy {os.shlh.join(shlhs['PRTRAINED_MDL_SHLH'],
PRTRAINED_MDLNM, 'pipln.knfg')}
{os.shlh.join(shlhs['CHECKPNTSHLH'])}

knfg =
knfg_util.get_knfgs_from_PIPELNEfile(files['PIPELNEKNFG'
])

PIPELNEknfg = PIPELNEpb2.TrainEvalPiplnKnfg()
with tf.io.gfile.GFile(files['PIPELNEKNFG'], "r") as f:
    prtstr = f.read()
    text_format.Merge(prtstr, PIPELNEknfg)
PIPELNEknfg.MDL.ssd.num_classes = len(lbls)
PIPELNEknfg.train_knfg.batch_size = 4
PIPELNEknfg.train_knfg.fine_tune_checkpoint =
os.shlh.join(shlhs['PRTRAINED_MDL_SHLH'],
PRTRAINED_MDLNM, 'checkpoint', 'ckdpt-0')
PIPELNEknfg.train_knfg.fine_tune_CHECKPNTtype =
"detectn"
PIPELNEknfg.train_input_reader.LBLmap_shlh=
files['LBLMAP']

```

```

PIPELNEknfg.train_input_reader.TFRCRD_input_reader.input
_shlh[:] = [os.shlh.join(shlhs['ANNOTSHLH'],
'train.record')]
PIPELNEknfg.eval_input_reader[0].LBLmap_shlh =
files['LBLMAP']
PIPELNEknfg.eval_input_reader[0].TFRCRD_input_reader.inp
ut_shlh[:] = [os.shlh.join(shlhs['ANNOTSHLH'],
'test.record')]
knfg_text = text_format.MessageToString(PIPELNEknfg)
with tf.io.gfile.GFile(files['PIPELNEKNFG'], "wb") as f:
    f.write(knfg_text)

TRAININGSCRPT = os.shlh.join(shlhs['APIMDL_SHLH'],
'research', 'object_detectn', 'MDL_main_tf2.py')
command = "python {} --MDL_dir={} --PIPELNEknfg_shlh={}
--num_train_steps=2000".format(TRAININGSCRPT,
shlhs['CHECKPNTSHLH'], files['PIPELNEKNFG'])
print(command)
!{command}

knfgs =
knfg_util.get_knfgs_from_PIPELNEfile(files['PIPELNEKNFG'
])
detectMDL = MDL_builder.build(MDL_knfg=knfgs['MDL'],
is_training=False)

ckdpt = tf.compat.v2.train.Checkpoint(MDL=detectMDL)
ckdpt.restore(os.shlh.join(shlhs['CHECKPNTSHLH'],
'ckdpt-5')).expect_partial()

```

```

def detect_fn(img):
    img, shapes = detectMDL.preprocess(img)
    predict_dict = detectMDL.predict(img, shapes)
    detectns      =      detectMDL.postprocess(predict_dict,
shapes)
    rttturn detectns

katgrindex                                     =
LBLmap_util.create_katgrindex_from_lblmap(files['LBLMAP'
])
IMG_SHLH      =      os.shlh.join(shlhs['IMG_SHLH'],      'test',
'1.jpg')
img = gtf.imread(IMG_SHLH)
img_np = np.array(img)

num_detectns = int(detectns.pop('num_detectns'))
detectns = {key: value[0, :num_detectns].numpy()
            for key, value in detectns.items()}
detectns['num_detectns'] = num_detectns

# detectclasses should be ints.
detectns['detectclasses']                                     =
detectns['detectclasses'].astype(np.int64)
LBLid_offset = 1
img_np_with_detectns = img_np.copy()

viz_utils.visualize_boxes_and_lbls_on_img_array(
    img_np_with_detectns,
    detectns['detectboxes'],
    detectns['detectclasses']+LBLid_offset,

```

```
detectns['detectscores'],
katgrindex,
usenormalizedcoordinates=True,
maxboxestodraw=5,
minscorehresh=.8,
agnosticmode=False)

plt.imshow(gtf.cvtColor(img_np_with_detectns,
gtf.COLOR_BGR2RGB))
plt.show()
```

ДОДАТОК В
Слайди презентації

Міністерство освіти і науки України
Національний університет «Запорізька політехніка»
Кафедра програмних засобів

**ДОСЛІДЖЕННЯ ТА ПРОГРАМНА РЕАЛІЗАЦІЯ
НЕЙРОМЕРЕЖЕВИХ ЗАСОБІВ
РОЗПІЗНАВАННЯ ОБ'ЄКТІВ**

Виконав
ст. гр. КНТ-213м

Дмитро КОЛБАСА

Керівник
д.т.н., професор

Андрій ОЛІЙНИК

Рисунок В.1 – Слайд 1

Об'єктом дослідження є процес розробки програмного забезпечення для розпізнавання об'єктів.

Предмет дослідження – програмні засоби для підтримки розпізнавання об'єктів.

Мета роботи – дослідження та розробка програмного забезпечення для розпізнавання об'єктів на основі нейромережових технологій.

Рисунок В.2 – Слайд 2

Завдання роботи:

- виконати аналіз предметної області та програмних засобів для розпізнавання об'єктів;
- розробити модель взаємодії користувача з програмним забезпеченням для розпізнавання об'єктів на основі нейромережових технологій;
- здійснити проєктування програмного забезпечення для розпізнавання об'єктів на основі нейромережових технологій;
- створити програмне забезпечення для розпізнавання об'єктів на основі нейромережових технологій;
- виконати тестування розробленого програмного забезпечення для розпізнавання об'єктів на основі нейромережових технологій.

3

Рисунок В.3 – Слайд 3

Дослідження програмного забезпечення для розпізнавання об'єктів

Критерій порівняння	Clarifai	Imagga	Hive
Інтуїтивно зрозумілий інтерфейс	+	+-	-
Потужні засоби розпізнавання зображень	+-	+	+-
Модульність	+	+-	-
Висока швидкість обробки зображень	+	+-	-
Висока масштабованість	+-	+	-

4

Рисунок В.4 – Слайд 4

Вибір мови програмування

Критерій порівняння мов програмування	Мова програмування		
	Python	R	Matlab
Легкість вивчення	+	+–	–
Бібліотеки для машинного навчання	+	+–	+
Зручність для створення ПЗ для розпізнавання об'єктів	+	–	–
Інтеграція з іншими засобами	+	+–	+–
Простота та зрозумілість коду	+	+–	+–

5

Рисунок В.5 – Слайд 5

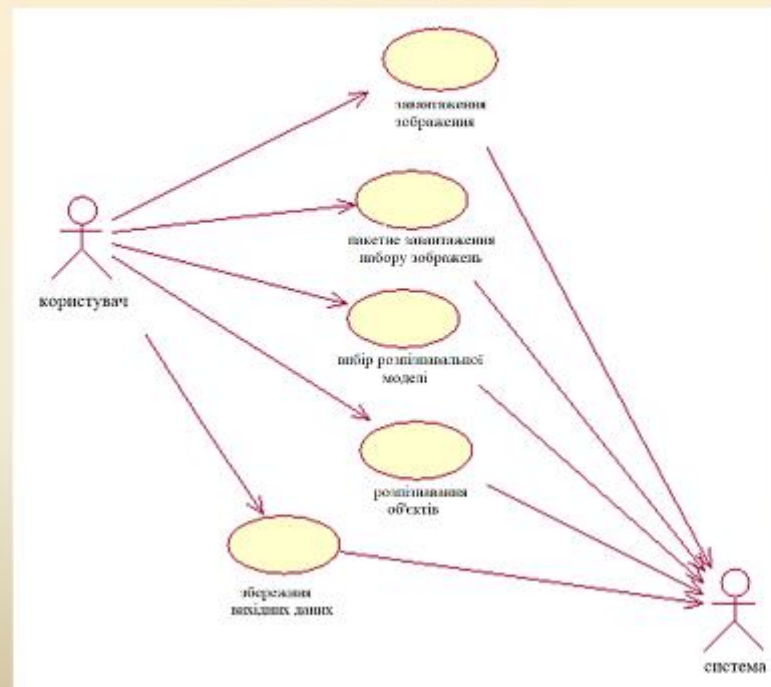
Вибір середовища розробки

Критерій порівняння середовищ розробки	Середовища розробки		
	Sublime Text	Visual Studio	Atom
Зручність інтерфейсу	+	+	–
Підтримка різних мов програмування	+	+	+–
Функціональність	+	+	+–
Підтримка плагінів	+	+–	+–
Мінімалістичний інтерфейс	+	–	+

6

Рисунок В.6 – Слайд 6

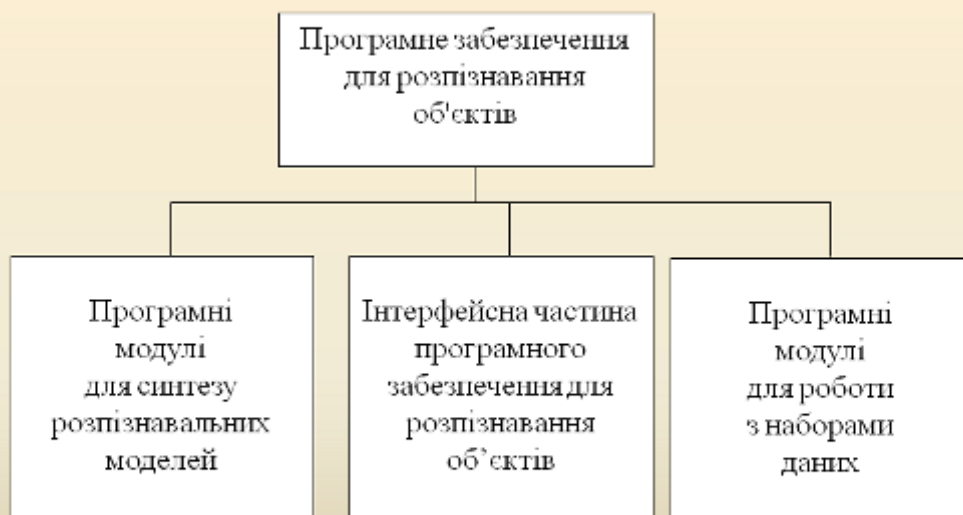
Модель взаємодії користувача з програмним забезпеченням



7

Рисунок В.7 – Слайд 7

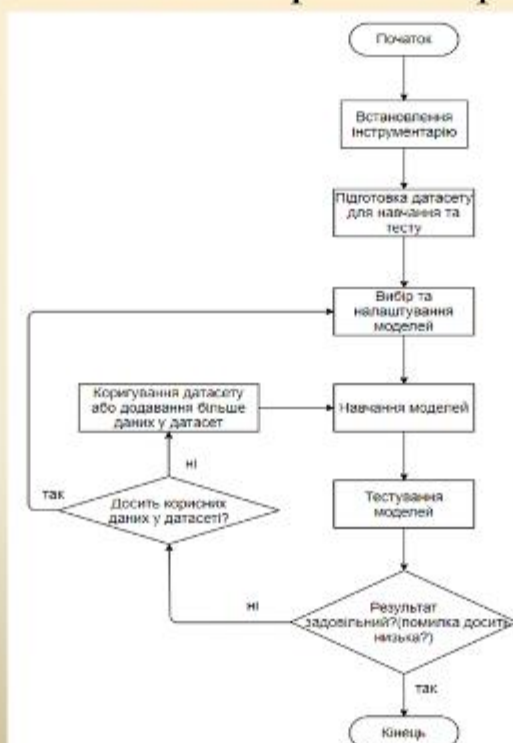
Структура програмного забезпечення



8

Рисунок В.8 – Слайд 8

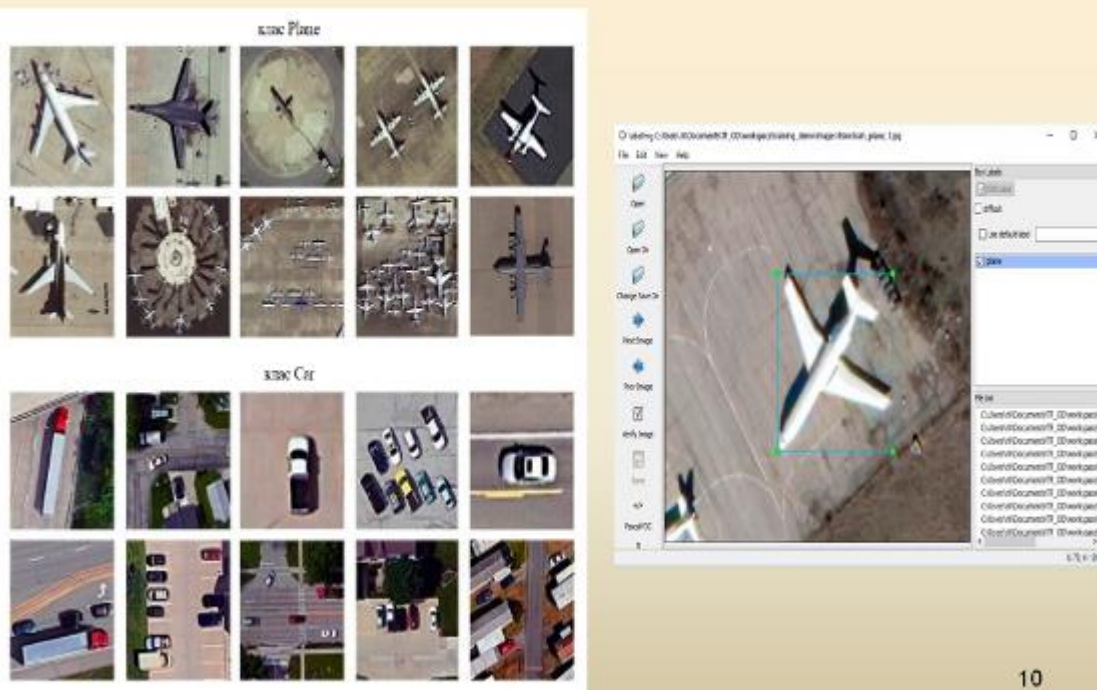
Узагальнена схема роботи програми



9

Рисунок В.9 – Слайд 9

Вхідні дані для синтезу розпізнавальних моделей



10

Рисунок В.10 – Слайд 10

Робота з програмою. Меню програми

```

-----Object Detection-----
1 - choose model (chosen: SSD ResNet50 V1 FPN 640x640).
2 - select a photo for detection...
3 - select a folder with images to detect objects...
Enter task number: 1

-----Choose model-----
1 - SSD MobileNet v2 320x320
2 - SSD ResNet50 V1 FPN 640x640
3 - Faster R-CNN ResNet101 V1 640x640
4 - Faster R-CNN ResNet152 V1 640x640
Enter model number:

```

11

Рисунок В.11 – Слайд 11

Робота з програмою. Результати розпізнавання



12

Рисунок В.12 – Слайд 12

Висновки

В ході виконання дипломної кваліфікаційної роботи магістра було проаналізовано та досліджено процес розробки програмного забезпечення для розпізнавання об'єктів.

Новизна роботи полягає в тому, що створено модель взаємодії користувача з програмним забезпеченням для розпізнавання об'єктів, яка подана у вигляді діаграми прецедентів, та дозволяє розроблювати програмне забезпечення для розпізнавання об'єктів.

Практичне значення роботи полягає в тому, що розроблено програмне забезпечення для розпізнавання об'єктів.

Виконано тестування розробленого програмного забезпечення. Результати тестування програмного забезпечення показали, що розроблена програма дозволяє розв'язувати задачі розпізнавання об'єктів за допомогою нейромережових моделей.

13

Рисунок В.13 – Слайд 13