МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

ІМЕНІ ТАРАСА ШЕВЧЕНКА

Факультет інформаційних технологій

Кафедра інтелектуальних технологій

ВИПУСКНА КВАЛІФІКАЦІЙНА РОБОТА

БАКАЛАВРА

НА ТЕМУ

Система розпізнавання фрагментів дерев з використанням згорткових нейронних мереж

Галузь знань **12 «Інформаційні технології»**

Спеціальність **122 «Комп'ютерні науки»**

Освітня програма **«Аналітика даних»**

Освітній рівень: бакалавр

Виконала: студентка 4 курсу, АнД-41

Воробченко Ю. С.

Керівник доцент Кіктєв М. О.

Випускна кваліфікаційна робота бакалавра допущена до захисту рішенням кафедри інтелектуальних технологій
Протокол № 13 від 05.06.2023 р.
зав. кафедри_____ доц. Іларіонов О.Є.

Київ – 2023

РЕФЕРАТ

Дипломна робота містить: 48 сторінок, 32 рисунків, 23 джерел.

Метою курсової роботи є створення системи для розпізнавання об’єктів та аналіз методів розпізнавання на прикладі задачі розпізнавання фрагментів дерев.

Об’єкт дослідження – набір візуальних даних з об’єктами, що належать до двох класів.

Предмет дослідження – точність, перспективи та методи розпізнавання квітів та бутонів дерев.

Методи дослідження – в роботі використовуються методи емпіричного дослідження (проводяться експерименти та порівнюються результати), методи машинного навчання для розпізнавання зображень

Для виконання експериментальних досліджень розроблено програму на мові програмування Python. Результати експериментальних досліджень оброблені і проаналізовані.

Основні результати роботи: було досліджено різні ефективність різних архітектур для розпізнавання об’єктів, виявлено найбільш перспективні, зроблено застосунок для використання натренованих моделей для розпізнавання.

КЛЮЧОВІ СЛОВА: розпізнавання, класифікація, нейронна мережа, глибинне навчання, згорткова мережа, дані.

ABSTRACT

The thesis contains: 48 pages, 32 figures, 23 sources.

The purpose of the course work is to create a system for object recognition and analyze recognition methods on the example of the task of recognizing tree fragments.

The object of research is a set of visual data with objects belonging to two classes.

The subject of the study is the accuracy, perspectives, and methods of recognizing flowers and tree buds.

Research methods - this work uses empirical research methods (experiments are conducted and the results are compared) and machine learning methods for image recognition.

A program in the Python programming language has been developed to perform experimental studies. The results of the experimental studies were processed and analyzed.

The main results of the work: various efficiencies of different architectures for object recognition were investigated, the most promising ones were identified, and an application for using trained models for recognition was made.

KEYWORDS: recognition, classification, neural network, deep learning, convolutional network, data

ЗМІСТ

ВСТУП 5
РОЗДІЛ 1. АНАЛІТИЧНИЙ ОГЛЯД. ПОСТАНОВКА ЗАВДАННЯ 6
1.1 Теоретичні відомості 6
1.2 Огляд літератури по темі 8
1.3 Огляд сучасного стану задачі розпізнавання образів і найбільш популярних рішень 10
1.4 Постановка задачі 16
1.5 Визначення вимог до додатку 17
1.6 Практична цінність розробки 18
РОЗДІЛ 2. ПРОЕКТНІ РІШЕННЯ 20
2.1 Огляд набору даних та підготовка даних 20
2.2 Засоби програмної реалізації 22
2.3 Розробка архітектури та макетів додатку 23
2.4 Розробка процесу навчання моделей 30
РОЗДІЛ 3. РЕАЛІЗАЦІЯ, ТЕСТОВИЙ ПРИКЛАД 32
3.1 Структура та специфікація модулів застосунку 32
3.2 Програмний застосунок 34
3.3 Відповідність функціональним вимогам 36
3.4 Результати експериментальних досліджень 37
3.5 Перевірка точності моделей в реальних умовах 40
ВИСНОВКИ 45
ДЖЕРЕЛА 46

ВСТУП

Одним із найперспективніших напрямів використання комп’ютерного зору і розпізнавання об’єктів є робототехніка. Використовуючи роботів, люди автоматизують все більше сфер свого життя. Але для ефективної їх роботи потрібні точні моделі для виявлення (локалізації) та класифікації об’єктів. В цій роботі розглядається ефективність існуючих архітектур мереж розпізнавання зображень.

Особливо важливою для людини сферою є аграрна сфера, тому автоматизація процесів, пов’язаних з нею, повинна бути пріоритетом для автоматизації виробничих процесів. Згідно даним міністерства сільського господарства США [1], на другому місці по витратам на дослідження за 2008-2018 роки є витрати на машинне навчання та аналіз даних.

Згідно міжнародному мета-аналізу 2019 року [2], тема глибинного аналізу швидко набирає популярність в останні роки:

Також, згідно тому ж аналізу, теми пов’язані з рослинами і їх класифікацією, посідають друге місце по актуальності.

РОЗДІЛ 1. АНАЛІТИЧНИЙ ОГЛЯД. ПОСТАНОВКА ЗАВДАННЯ

1.1 Теоретичні відомості

Задача розпізнавання образів — це задача віднесення вихідних даних до певного класу за допомогою виділення істотних ознак, що характеризують ці дані, із загальної маси несуттєвих даних. Складається з задачі локалізації та задачі класифікації.

Локалізація об’єкта - визначення розташування одного або кількох об’єктів на зображенні та їх відмічення навколо їх розміру.

Задача класифікації - проблема визначення, до якої категорії з набору належить спостереження. Класичними прикладами є відношення електронного листа до класу «спам» або «не спам», та встановлення діагнозу пацієнта по його симптомам.

Згорткова нейронна мережа (Convolutional Neural Network, CNN) - клас глибоких нейронних мереж, які застосовуються для розпізнавання візуальних образів. Вперше запропонована Яном Лекуном в 1989 році [3]. Основною ідеєю є послідовне чергування шарів згортки і шарів підвиборки (пулінгу). Мережа однонаправлена.

Регіональна згорткова нейронна мережа (Region-Based Convolutional Neural Network, R-CNN) - тип моделі нейронних мереж на базі згорткових нейронних мереж, які використовуються для розпізнавання об’єктів. Складаються з трьох модулів: модуль пропозиції локації, модуль виділення ознак, класифікатор.

Згортковий шар (the kernel) - шар нейронної мережі, в якому проводиться операція згортки зображення. Представляє собою фільтр, який проходить по матриці зображення і, аналізуючи її, виділяє матрицю ознак. Наприклад, перший шар в CNN зазвичай відповідає за захоплення низькорівневих ознак, як-то край, колір, градієнтна орієнтація і т. д. Може мати два типи результатів: один, в якому згорнута ознака має меншу розмірність у порівнянні з вхідною (Valid padding), і

другий, в якому результат має таку ж або більшу розмірність (Same padding). Останній результат досягається додаванням (паддінгом) нулів до країв вхідної матриці, як показано на малюнку.

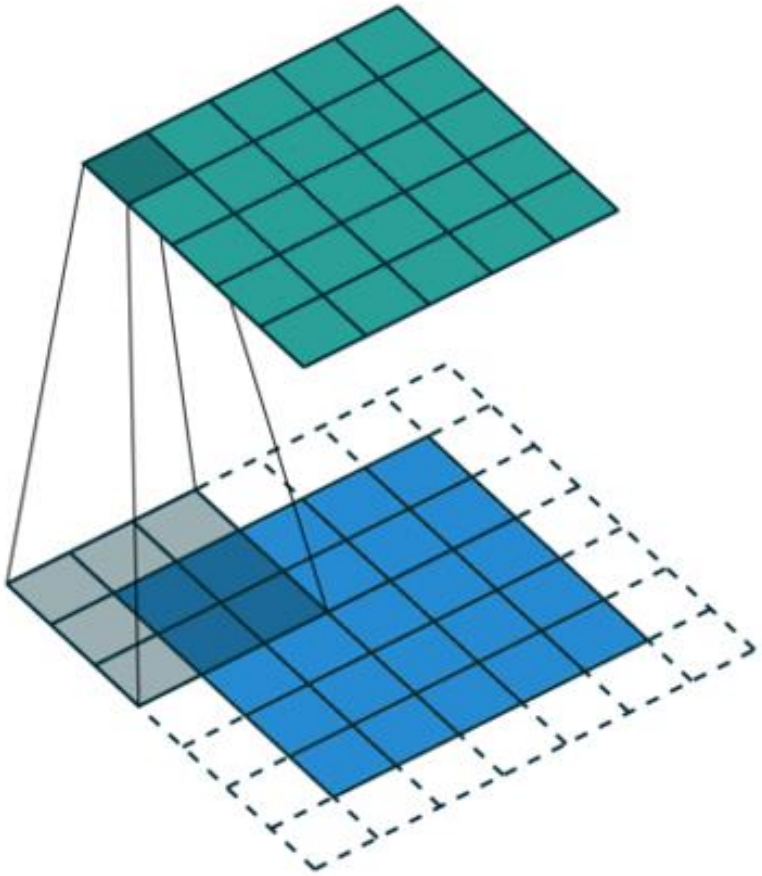

Рисунок 1.1 Візуалізація застосування кернелу до перетвореної матриці

Пулінговий шар (Pooling Layer) - як і згортковий шар, слугує для зменшення розміру обчислюємого елемента, а також для виділення домінуючих ознак. Використовуються два типи пулінгу: середній (Average Pooling) та максимальний (Max Pooling). Середній пулінг повертає середнє значення в матриці зображення, покритій кернелом, а максимальний пулінг повертає максимальне значення.

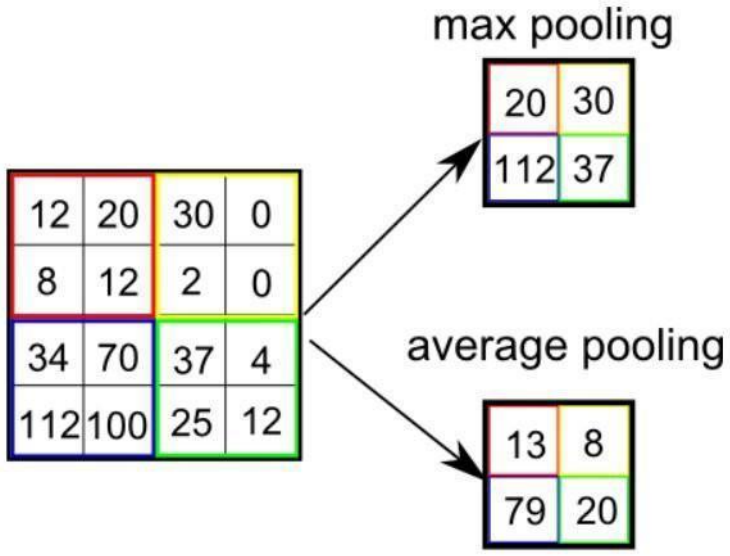

Рисунок 1.2 Ілюстрація роботи різних типів пулінгового шару

Повністю пов’язаний шар (Fully Connected Layer, або Dense Layer) - шар нейронної мережі, в якому всі нейрони попереднього шару пов’язані з кожним нейроном FC шару. [4]

Функція активації - функція, що порівнює вхідне значення з пороговим значенням. Якщо воно досягнуто, нейрон активовано; якщо значення менше за порогове, нейрон не активовано і вихід з нього не посилається на наступний шар.

Оптимізатори - алгоритми, що використовуються для незначної зміни параметрів, таких як ваги та швидкість навчання, для мінімізації втрат моделі.

1.2 Огляд літератури по темі

Проблема виявлення та класифікації розглядалась в багатьох наукових роботах, наприклад в роботі українських дослідників [5] було зроблено нейронну мережу для розпізнавання яблук на дереві та їх класифікації до одного з дев’яти класів - два для здорових і сім для хворих яблук. Для цього використовувалась архітектура MASK-RCNN.

В іншій роботі українських дослідників [6] було використано технології комп’ютерного зору для знаходження полуниці, визначення її координат та класифікація її зрілості. Навчання для більшої точності проводилось в умовах різного освітлення, а для більш точного розпізнавання використовувалась

техніка розмиття контурів об’єкта. Класифікація проводилась по характеристикам розміру і кольору.

Також задача розпізнавання об’єктів з метою оцінки кількості для застосування в аграрній сфері розглядалась в роботі китайських вчених на прикладі бавовни у 2020 році [7]. Вчені використовували вже застарілу модель YOLO v3, та досягли середньої квадратичної помилки від 0.50 до 0.60 на лінійний метр. Досліджувався вплив навчального набору даних на результати, з висновком що набір даних з найбільшою варіативністю показників дозволяє отримати найменшу похибку.

При зборі візуальних даних для навчання моделей або іншого практичного застосування з різних причин можуть з’являтись аномалії зображень. Для їх знаходження теж можуть використовуватись моделі для розпізнавання об’єктів, як в роботі українських дослідників [8], які використовують для цього щільну нейронну мережу та два варіанти моделі EfficientNet разом.

Дослідження, що використовує та аналізує використання рекурентних нейронних мереж (RNN) для ідентифікації яблук в кронах дерев [9], приходить до висновку, що найважливішими причинами помилок є неточна сегментація зображень та низька роздільна здатність камери. Також стверджується, що згорткова рекурентна нейронна мережа є найкращим вибором для розв’язання такої задачі.

В дослідженні проблеми розпізнавання зрілого врожаю при великій щільності засаджень поруч із бур’янами [10] зазначається, що маркування кожного елементу (кожного листку) для тестового набору даних надає можливість отримати більшу точність розпізнавання порівняно зі стандартними методами маркування. Згідно дослідженню, такий метод збільшує середню точність при впевненості 0.5 на 1.2% та практично вирішує проблему неправильних виявлень.

Робота зі схожою тематикою, зосереджена на розпізнаванні бур’яну для його контролю [11], зазначає що найкраще з поставленою задачею справляються моделі на базі згорткових нейронних мереж.

В дослідженні систем для автоматичної ідентифікації хвороб рослин [12] було побудовано легковажну згорткову мережу, що змогла досягти точності в 99.16% на наборі даних PlantVillage.

Для рішення задачі ідентифікації бур'яну та кунжуту було модифіковано модель YOLO v4 доданням механізму уваги [13]. Створену модель було порівняно з популярними моделями Fast R-CNN, SSD, YOLOv3, YOLOv4, та YOLOv4-tiny за критерієм точності, та отримано кращі результати ніж всі перераховані моделі з середньою точністю в 96.16%.

Точність розпізнавання на прикладі полуниці також розглядається при використанні тривимірних бінокулярних камер [14], для чого використовувались моделі YOLO v3 та Mask R-CNN. В даних умовах з використанням набору даних в 1000 зображень вдалось досягти точності в 93.4% та 94.5% відповідно.

Значних результатів в покращенні моделей на архітектурі CNN було досягнуто шляхом урахування не лише втрат при навчанні, а також і різкості (Sharpness-Aware Minimization, SAM) [15]. Завдяки цьому було досягнуто зменшення похибки в середньому на ~20% на найбільш популярних наборах даних.

Згідно аналізу факторів [16], що впливають на точність класифікації за допомогою нейронних мереж, найбільш впливовими є характеристики датасетів для тренування і перевірки, в той час як архітектура системи, в першу чергу кількість шарів, мала відносно малий вплив.

Наприклад, в роботі з Арізонського університету в США [17] було використано 12000 зображень, з яких лише 1000 є оригінальними а інші 11000 - аугментовані зображення.

1.3 Огляд сучасного стану задачі розпізнавання образів і найбільш популярних рішень

У 2015 році було створено архітектуру Fast R-CNN,на основі архітектури R-CNN [18]. Згідно звіту автора, створена модель працює в 9 разів швидше за оригінальну.

Архітектура моделі приймає фотографію та набір пропозицій регіонів як вхідні дані, які передаються через глибоку згорткову нейронну мережу. Попередньо підготовлена мережа CNN, така як VGG-16, використовується для виділення ознак. Кінець глибокої CNN — це спеціальний рівень, який називається «Рівень об'єднання регіонів інтересів» або «Об'єднання ROI», який витягує характеристики, специфічні для даного регіону-кандидата введення. Потім вихідні дані CNN інтерпретуються повністю зв'язаним шаром, після чого модель роздвоюється на два виходи: один для прогнозування класу через шар softmax, а інший із лінійним виходом для обмежувальної рамки. Потім цей процес повторюється кілька разів для кожної цікавої області на даному зображенні.

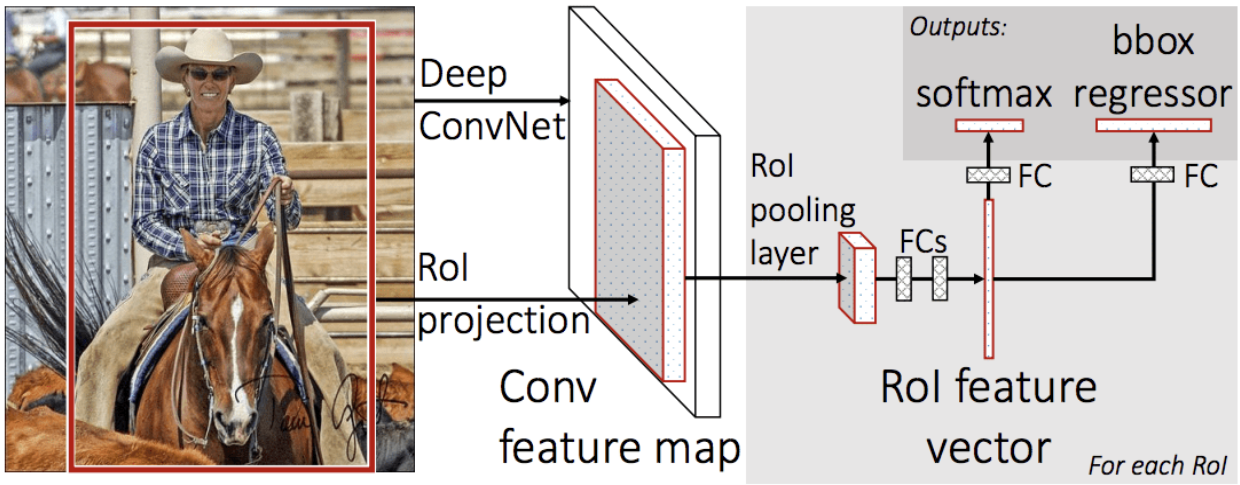

Рисунок 1.3 Схема роботи архітектури Fast R-CNN

Вже через рік мережа була покращена архітектурою Faster R-CNN [19], яка складається з двох модулів: мережа пропозиції локації та Fast R-CNN. Мережа пропозицій локацій працює як механізм привернення уваги до певних локацій для Fast R-CNN, інформуючи її куди потрібно «дивитись».

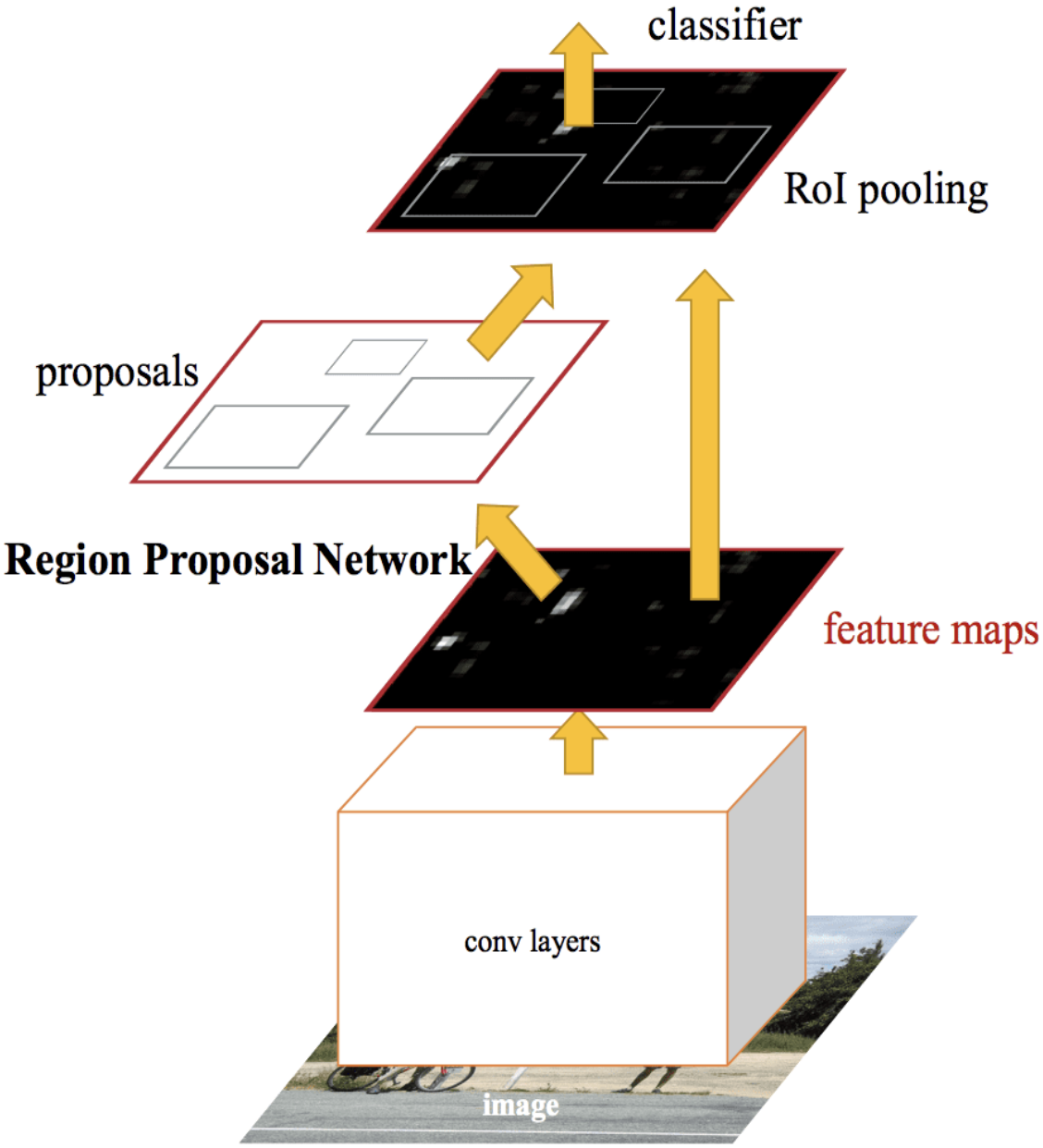

Рисунок 1.4 - схема архітектура Faster R-CNN

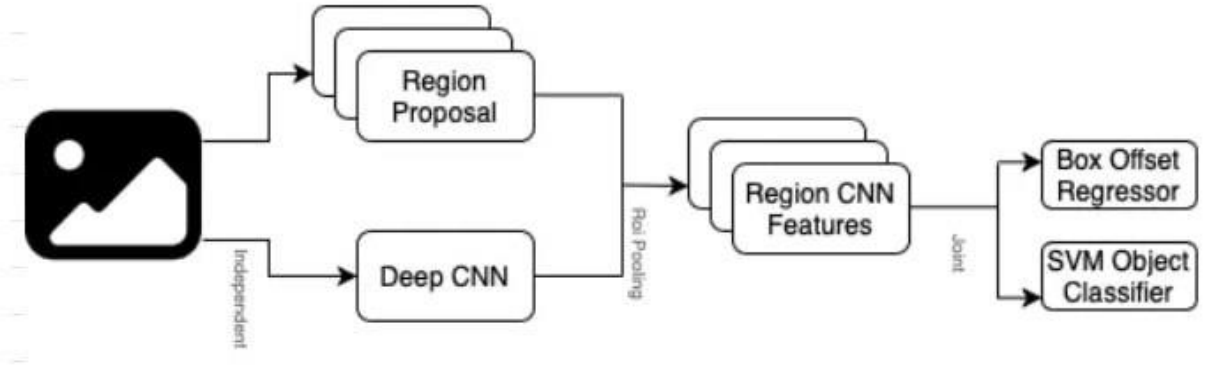

Figure 1: The architecture of Fast R-CNN.

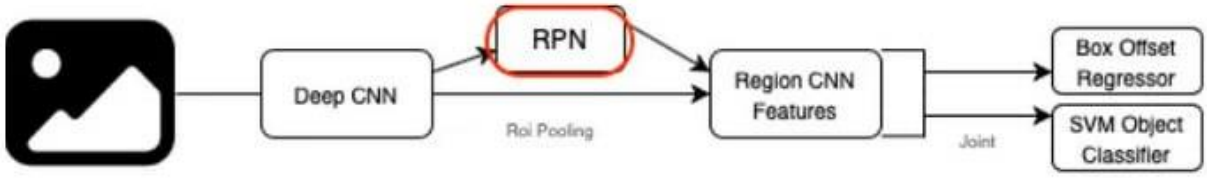

Figure 2: The architecture of Faster R-CNN.

Рисунок 1.5 - схема відмінностей Fast R-CNN та Faster R-CNN

Іншою групою моделей для розпізнавання є YOLO - You Only Look Once [20], тобто «ти дивишся лише раз». Менш точні ніж моделі R-CNN, моделі YOLO тим не менш значно швидші та за рахунок цього добре можуть розпізнавати об’єкти в реальному часі. Архітектура представляє собою єдину нейронну мережу, яка бере на вхід зображення і напряму передбачає рамки для виділення об’єктів.

Модель спершу розділяє вхідне зображення на клітини, де кожна клітинка відповідає за прогнозування обмежувальної рамки, якщо центр обмежувальної рамки об’єкту потрапляє в цю клітинку.

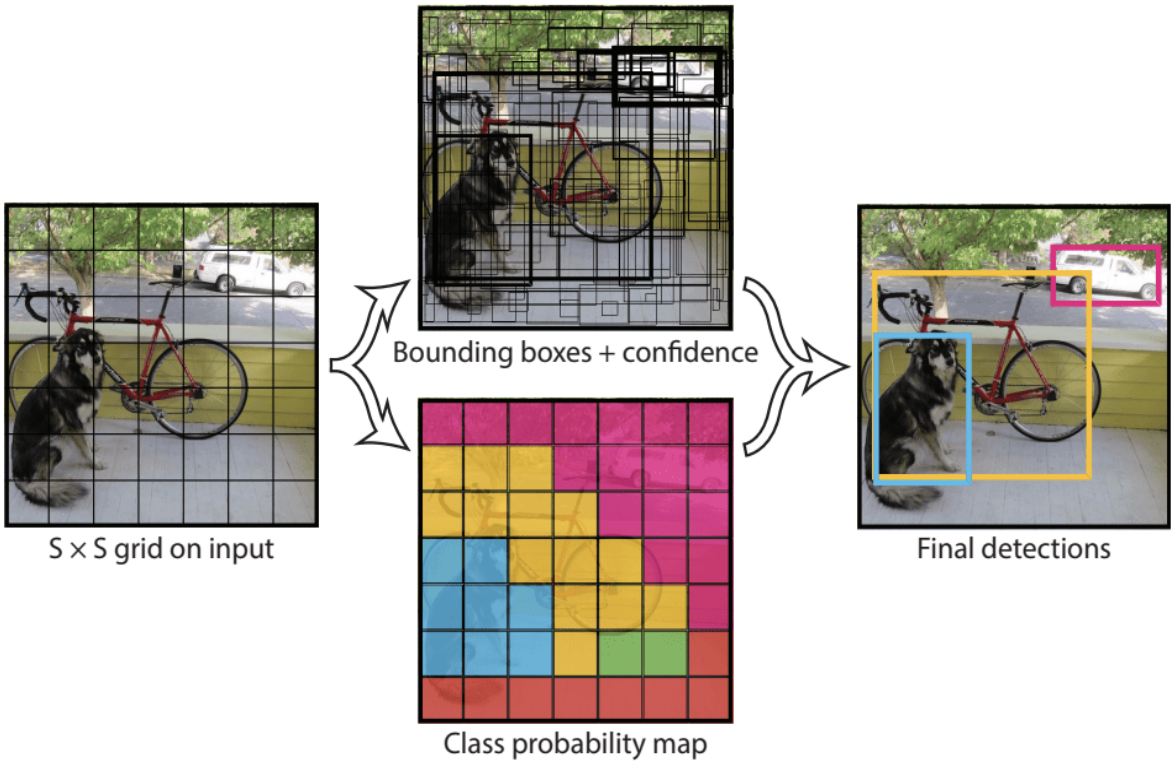

Рисунок 1.6 Схема роботи YOLO моделі

Одноразовий детектор (Single-Shot Detector, SSD) [21] - схожий по схемі роботи на YOLO, цей детектор працює у два кроки: видобуття карти ознак за допомогою мережі VGG16 і застосування згорткових фільтрів для виявлення об’єктів. Наприклад, згортковий шар Conv4_3 робить по чотири передбачення на кожну клітку зображення, даючи кожному передбаченню рейтинг віднесення до того чи іншого класу об’єктів (в тому числі нульового класу, тобто відсутності класу). Після цього вибирається найбільш високо оцінене передбачення для кожної клітинки.

Мережа для отримання карти ознак називається хребтом моделі (backbone model), а згорткові шари для розпізнавання - SSD голови (SSD head).

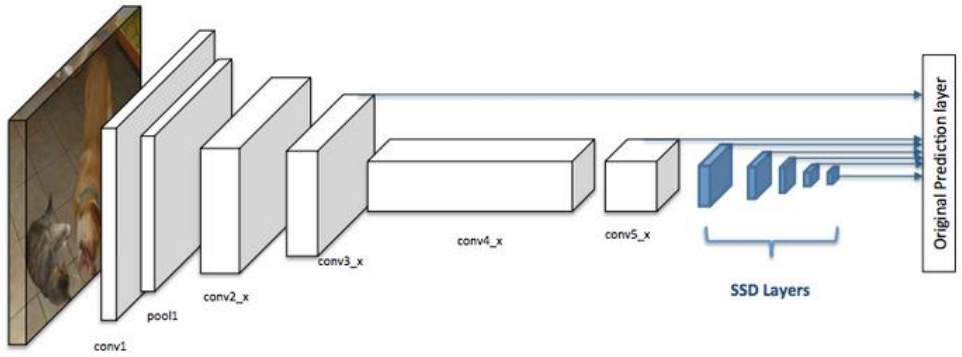

Рисунок 1.7 Схема моделі SSD

EfficientDet [22] - архітектура, яка утилізує хребет EfficienNet, особливістю якого є автоматичне масштабування мережі в залежності від вхідного зображення. Наприклад, для більшого зображення потрібна більша кількість слоїв мережі, щоб збільшити рецепторне поле, та більша кількість каналів, щоб захоплювати більш дрібні паттерни на зображенні.

Згідно дослідженню 2019 року, EfficientDet детектор досягає найкращих результатів на наборі даних COCO test-dev з 77 мільйонами параметрів та 410 мільярдами операцій з рухомою комою за секунду.

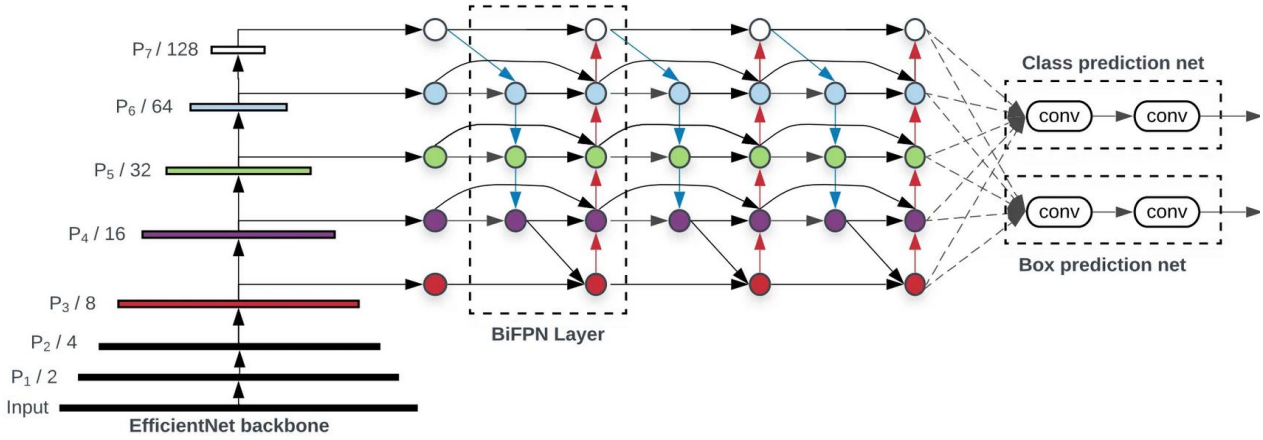

Рисунок 1.8 Схема архітектури EfficientDet

Автоматичне машинне навчання (Auto ML) [23] - використання автоматизованих інструментів і методів для автоматизації різних етапів конвеєра машинного навчання. Конвеєр ML включає такі завдання, як попередня обробка даних, розробка функцій, вибір моделі, налаштування гіперпараметрів та оцінка моделі. AutoML має на меті впорядкувати та спростити ці завдання шляхом

автоматизації якомога більшої кількості кроків, зменшуючи потребу в ручному втручанні.

Основна мета AutoML - демократизувати машинне навчання і зробити його більш доступним для людей і організацій, які не мають великого досвіду в галузі машинного навчання. Автоматизуючи повторювані та трудомісткі аспекти конвеєра машинного навчання, AutoML дозволяє неспеціалістам більш ефективно будувати та розгортати моделі машинного навчання. Хоча фокусом є неспеціалісти, автоматичне машинне навчання також дає змогу досвідченим аналітикам даних прискорити робочий процес і зосередитися на більш складних і творчих аспектах розробки ML-моделей.

Інструменти AutoML зазвичай надають зручний інтерфейс або фреймворк, який проводить користувачів через весь конвеєр ML.

AutoML можна застосовувати до різних задач ML, включаючи класифікацію, регресію, кластеризацію та аналіз часових рядів. Він дозволяє користувачам визначити свою проблему і дані, а система AutoML подбає про все інше, автоматично досліджуючи різні алгоритми ML, методи попередньої обробки і конфігурації гіперпараметрів, щоб знайти найкращу модель для даної задачі.

Різні компанії пропонують свої сервіси автоматичного машинного навчання, наприклад Google, Azura та Roboflow.

1.4 Постановка задачі

Цілю цієї роботи є дослідження методів розпізнавання об’єктів на зображеннях та перспективності застосування розпізнавання зображень в аграрній сфері на прикладі розпізнавання бруньок та квітів яблунь. Для демонстраційних та експериментальних цілей повинен бути створений додаток, що дозволяє з простим інтерфейсом використовувати натреновані та збережені моделі для розпізнавання зображень з виведенням результатів на екран.

Для визначення перспективності направлення потрібно порівняти між собою різні моделі розпізнавання об’єктів, видані їми результати та точність моделей.

Системні вимоги: для виконання роботи необхідна система з операційною системою Windows та не менш ніж 12Гб оперативної пам’яті для навчання моделей.

Вимоги до програмного забезпечення: встановлений Python версії 3.9 або більше, встановлені бібліотеки для Python tensorflow, tensorboard, pytorch, OpenCV, roboflow та бібліотеки від яких вони залежать. Повинні бути завантажені копії моделі YOLO v5 та набір моделей від tensorflow. Повинен бути наявний аккаунт в сервісі roboflow.

Програмний застосунок повинен бути виконаний за допомогою бібліотеки CustomTkinter.

Вхідною інформацією слугують статті про поточні методи для вирішення задачі розпізнавання зображень та створений набір даних, що являють собою фотографії яблуневих дерев.

1.5 Визначення вимог до додатку

Функціональні вимоги

- Локалізує та класифікує квіти на зображенні, обводячи їх
- Можливість завантажувати зображення з внутрішнього сховища або з камери
- Можливість вибирати різні моделі
- Підтримка операційної системи Windows
- Можливість додавати власні моделі

Нефункціональні вимоги

- Зрозумілий та інтуїтивний інтерфейс

- Швидкість використання - система повинна видавати результат не повільніше ніж за 5 секунд
- Стабільність

1.6 Практична цінність розробки

В цій роботі проблема розпізнавання розглядається на прикладі розпізнавання бутонів та розкритих квітів яблунь. Якщо застосувати мережу, натреновану для рішення цієї задачі, для використання роботом, він зможе дати наступні переваги:

1) Управління садом: власники яблуневих садів і фермери можуть отримати вигоду від такого робота в ефективному управлінні своїми садами. Точно визначаючи бруньки і розкриті квіти, робот може надати цінні дані про ріст і розвиток яблунь. Ця інформація може допомогти оптимізувати зрошення, боротьбу зі шкідниками та загальний стан саду. Наприклад, на основі отриманих даних про щільність зростання фермер може прийняти рішення про зменшення кількості дерев або відсікання гілок для покращення якості майбутнього врожаю та стану дерев.

2) Моніторинг запилення: бджоли та інші запилювачі відіграють вирішальну роль у процесі запилення яблунь. Робот, здатний розпізнавати розкриті квіти, може допомогти відстежувати присутність та активність запилювачів у саду. Ця інформація може бути використана для оцінки ефективності запилення та вжиття заходів для збільшення популяції запилювачів, якщо це необхідно.

3) Дослідження та розробки: дані, зібрані роботом про бутони та розкриті квіти, можуть бути цінними для дослідників і селекціонерів у галузі вирощування яблунь. Це може сприяти виведенню нових сортів з покращеними характеристиками, підвищеною стійкістю до хвороб чи шкідників або оптимізованою схемою цвітіння.

4) Прогнозування врожайності: аналізуючи стадію розвитку бруньок і розкритих квіток, робот може надати інформацію про потенційний урожай яблук. Ці дані можуть бути використані фермерами та менеджерами садів для планування графіків збору врожаю, оцінки обсягів виробництва та прийняття обґрунтованих рішень щодо розподілу ресурсів і ринкових стратегій.

Створений програмний застосунок дозволяє швидко протестувати та оцінити якість натренованих моделей, та може використовуватись як основа для створення програмного забезпечення для практичного застосування, наприклад при перенесенні програми в робота з отриманням інформації з його камери і виведенням її на носій керівника дій робота.

РОЗДІЛ 2. ПРОЕКТНІ РІШЕННЯ

2.1 Огляд набору даних та підготовка даних

Для тренування моделей використовується набір зі 100 фотографій дерев виду «яблуня Жигулівська», зроблених на камеру Sony Alpha ILCE-7M3 с об'єктивом Sony FE 24-240mm f/3.5-6.3 OSS (SEL24240). Фотографії створені не в приміщенні, при природному освітленні. Розмір зображень становить 7952 на 5304 пікселів.

На зображеннях було вручну розмічено розташування об’єктів за допомогою сервісу roboflow і підписано їх класи, після чого експортовано у форматі Pascal VOC.

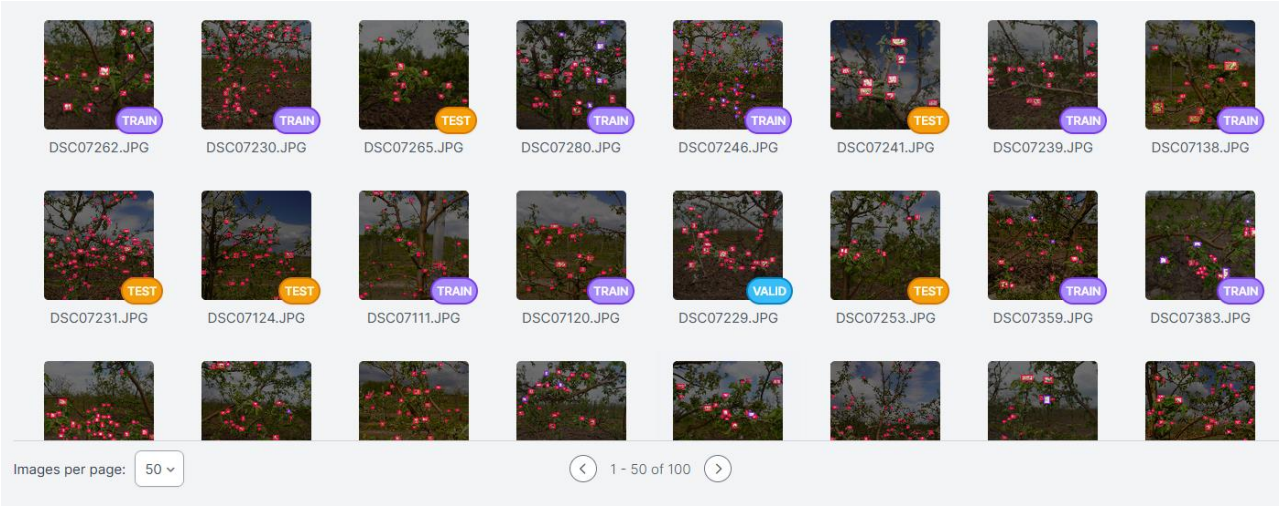

Рисунок 2.1 Набір навчальних даних після розмітки в сервісі roboflow

Рисунок 2.2 Приклад розміченого зображення

```
<object>
    <name>rosebud</name>
    <pose>Unspecified</pose>
    <truncated>0</truncated>
    <difficult>0</difficult>
    <occluded>0</occluded>
    <bndbox>
        <xmin>248</xmin>
        <xmax>272</xmax>
        <ymin>169</ymin>
        <ymax>196</ymax>
    </bndbox>
</object>
```

Рисунок 2.3 Приклад даних одного об’єкту

З огляду на характеристику системи, на якій проводилось тренування, розмір зображень перед застосуванням було зжато до 820 на 640 пікселів. Це пов’язано с тим, що при більших розмірах при тренуванні моделей на наявному обладнанні моделям не вистачало оперативної пам’яті комп’ютера.

На кожному зображенні було виділено від 17 до 160 об’єктів.

Набір даних було розбито на три частини: для тренування, валідації та тестування.

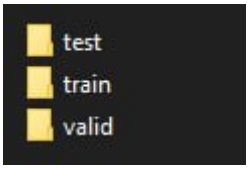

Рисунок 2.4 Структура навчальних даних

2.2 Засоби програмної реалізації

В якості програмного середовища розробки використовується легковаговий застосунок SublimeText.

Python - високорівнева мова програмування з динамічною строгою типізацією та автоматичним управлінням пам'яттю. Найпопулярніша мова програмування для задач аналізу даних та будування нейронних мереж.

Tensorflow - безкоштовна бібліотека для задач машинного навчання та штучного інтелекту, розроблена командою з Google та вперше випущена у 2015 році. Бібліотека має відкритий код і регулярно оновлюється. Бібліотека також надає доступ до файлів конфігурації набору моделей для різних задач, в тому числі для задачі розпізнавання об’єктів.

Keras - бібліотека з відкритим кодом, яка написана мовою Python і забезпечує взаємодію зі штучними нейронними мережами. Націлена на оперативну роботу з мережами глибинного навчання, при цьому спроектована так, щоб бути компактною, модульною та розширюваною. Бібліотека містить численні реалізації широко застосовуваних будівельних блоків нейронних мереж, таких як шари, цільові та передавальні функції, оптимізатори та безліч інструментів для спрощення роботи із зображеннями та текстом. Інтегрується з бібліотекою Tensorflow.

Roboflow - платформа та бібліотека для Python. Платформа надає можливості для попередньої обробки даних та навчання моделей комп’ютерного зору. Бібліотека надає можливість імпортувати та застосовувати натреновані на платформі моделі, або навчати свої локально.

PyTorch - фреймворк з відкритим кодом, що застосовується для машинного навчання. Був випущений командою Meta у 2016 році. Використовується в першу чергу для вирішення задач комп’ютерного зору та обробки природної мови. Має інтерфейс для мови Python. Надає дві високорівневі функціональності: тензорні обчислення з використанням графічного процесору та глибинні нейронні мережі на системі автоматичного диференціювання.

OpenCV (Open Source Computer Vision Library*)* - бібліотека з відкритим кодом, що містить функції та алгоритми для комп’ютерного зору та обробки зображень, написана на мові C++. Розроблена компанією Intel і випущена в 2006 році.

CustomTkinter - розширення вбудованої стандартної бібліотеки Tkinter для Python, що надає можливість будувати GUI, що виглядить більш новітньо, за допомогою тих же методів.

2.3 Розробка архітектури та макетів додатку

Додаток повинен мати просту архітектуру, що складається з візуального інтерфейсу та моделей для розпізнавання зображень. Було створено відповідну схему:

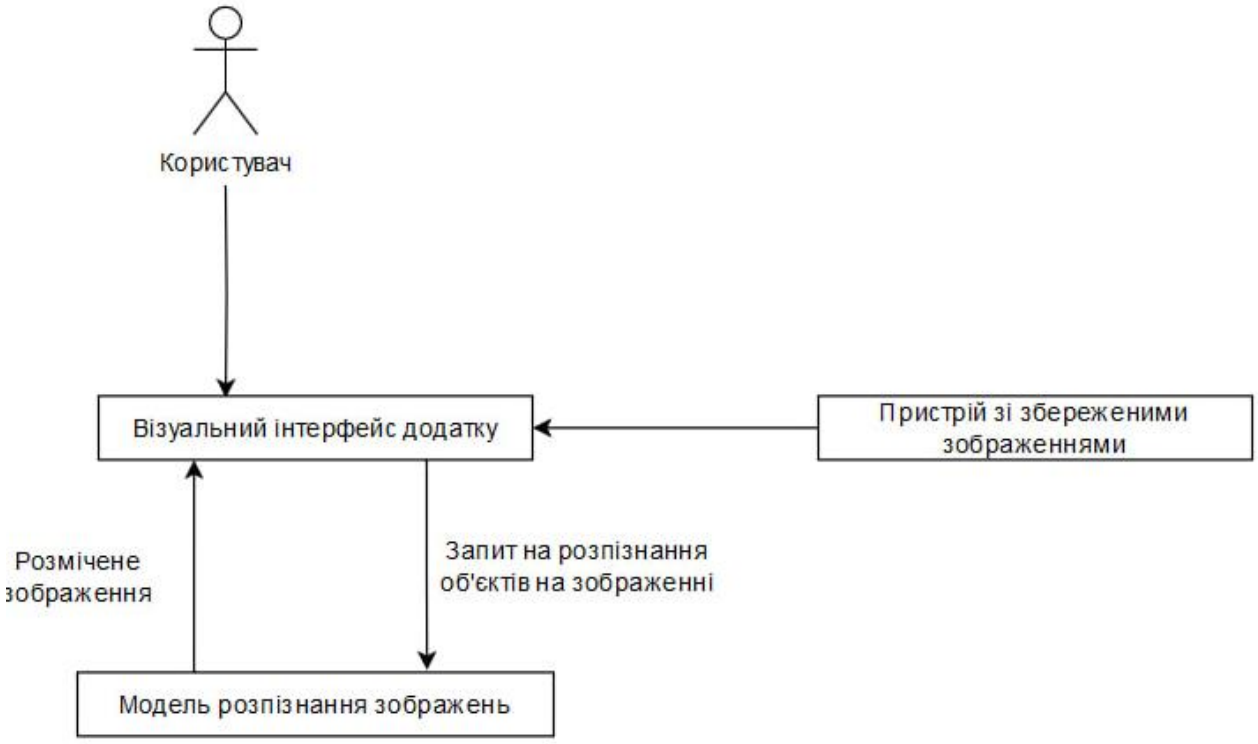

Рисунок 2.5 Архітектура програмного додатку

Користувач повинен мати можливість вибирати модель та зображення для проведення розпізнавання. Для моделювання процесу взаємодії користувача з застосунком було побудовано наступні моделі:

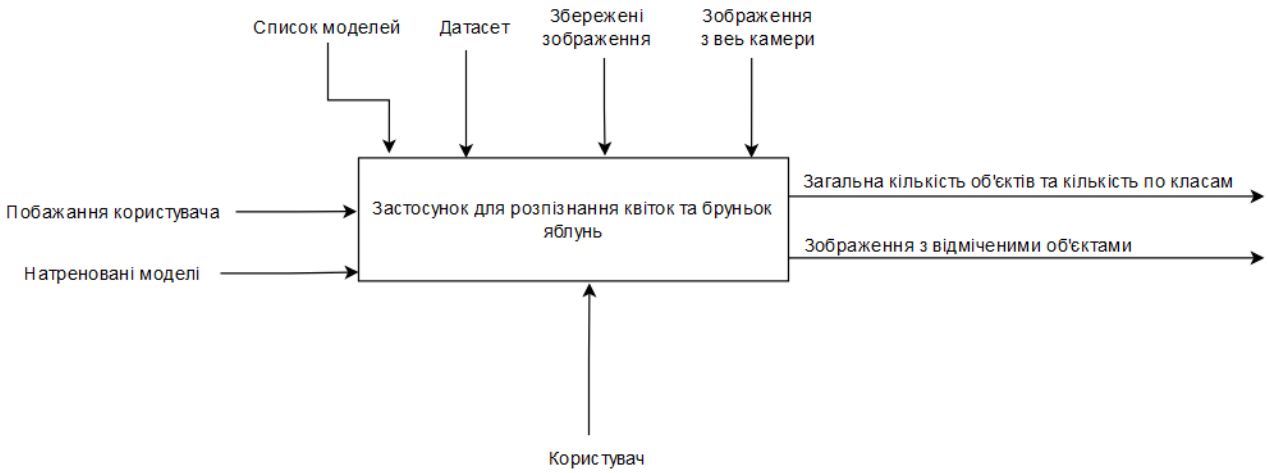

Рисунок 2.6 IDEF0 діаграма процесу використання додатку

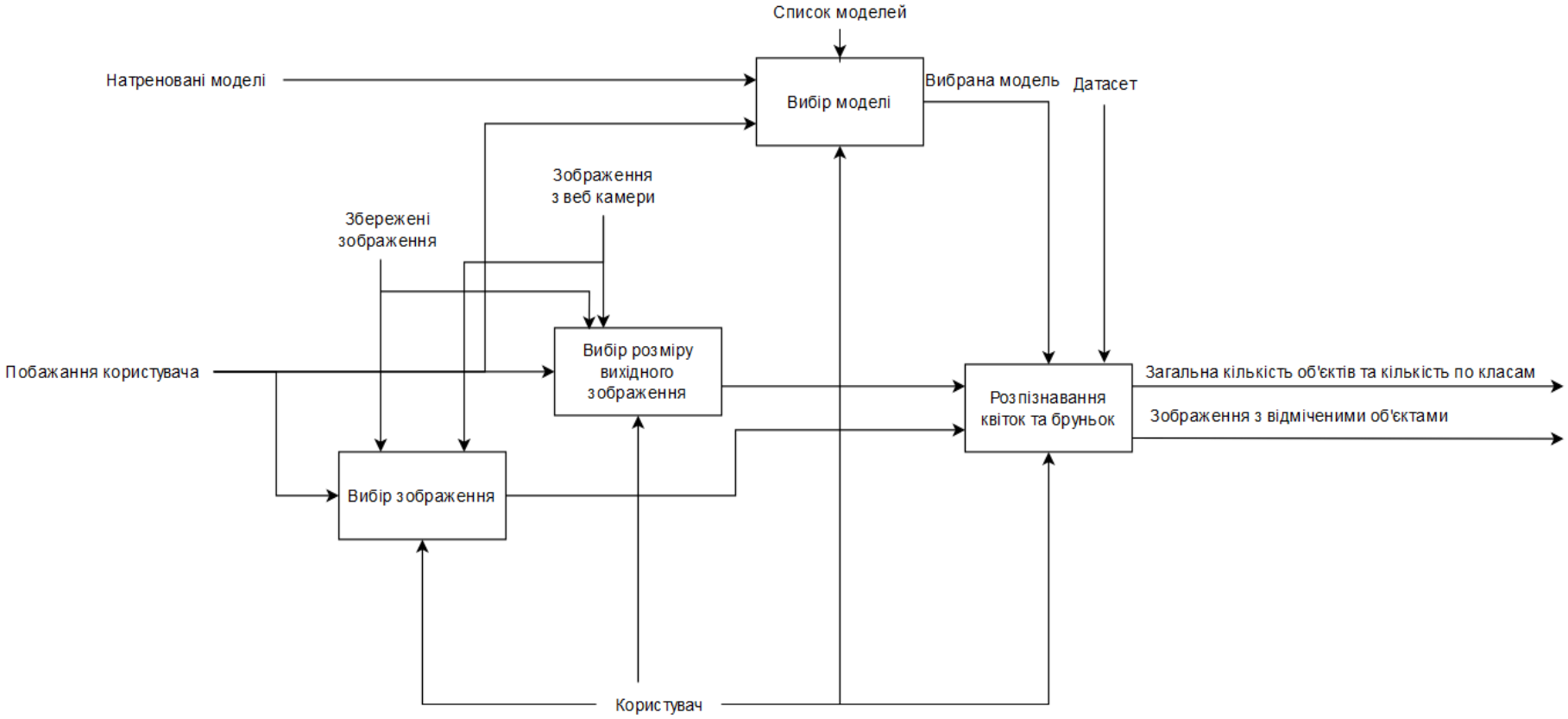

Рисунок 2.7 Декомпозиція діаграми процесу використання додатку

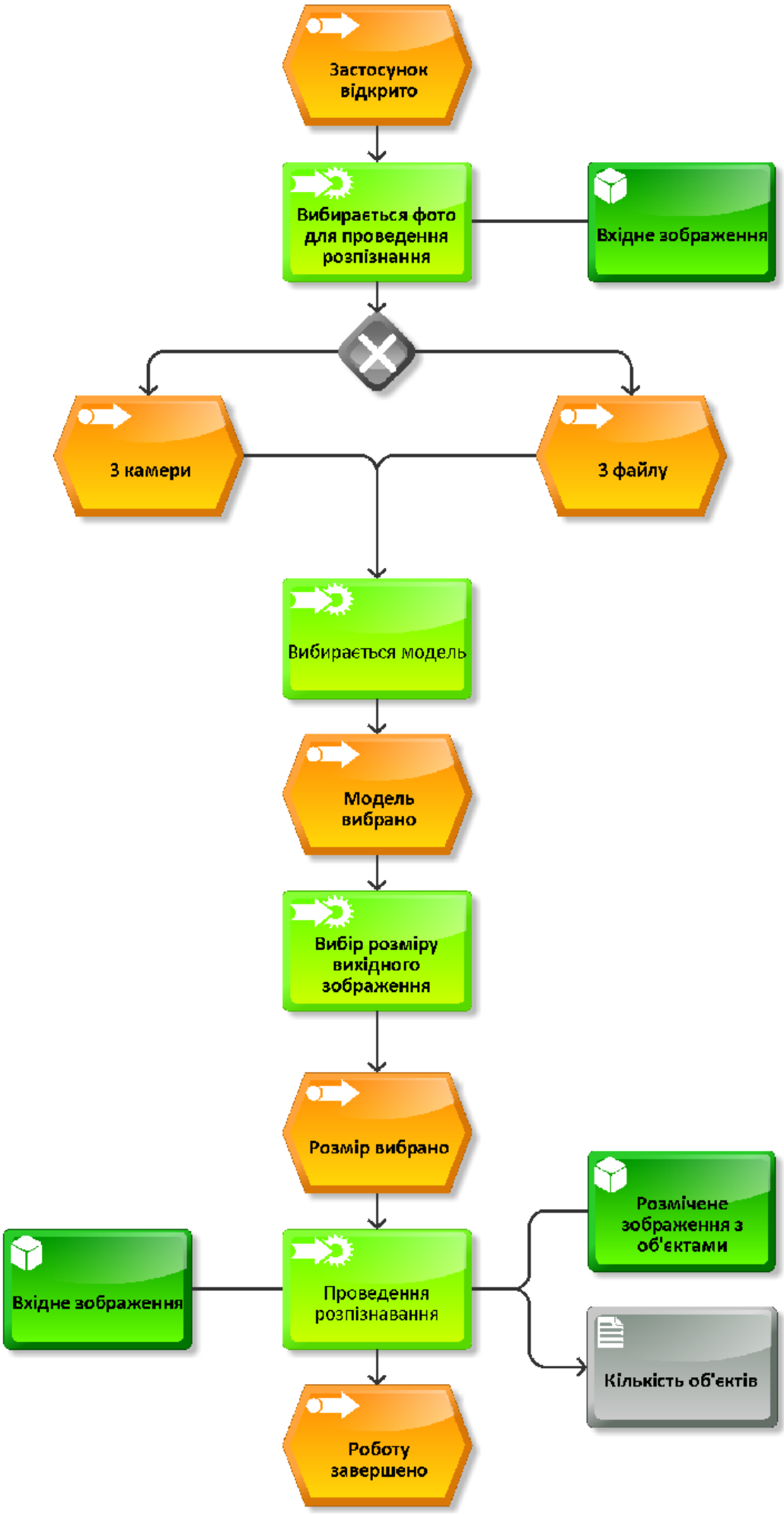
Застосунок відкрито
Вибирається фото для проведення розпізнання
Вхідне зображення
З камери
З файлу
Вибирається модель
Модель вибрано
Вибір розміру вихідного зображення
Розмір вибрано
Вхідне зображення
Проведення розпізнавання
Розмічене зображення з об'єктами
Кількість об'єктів
Роботу завершено

Рисунок 2.8 Схема взаємодії користувача з застосунком

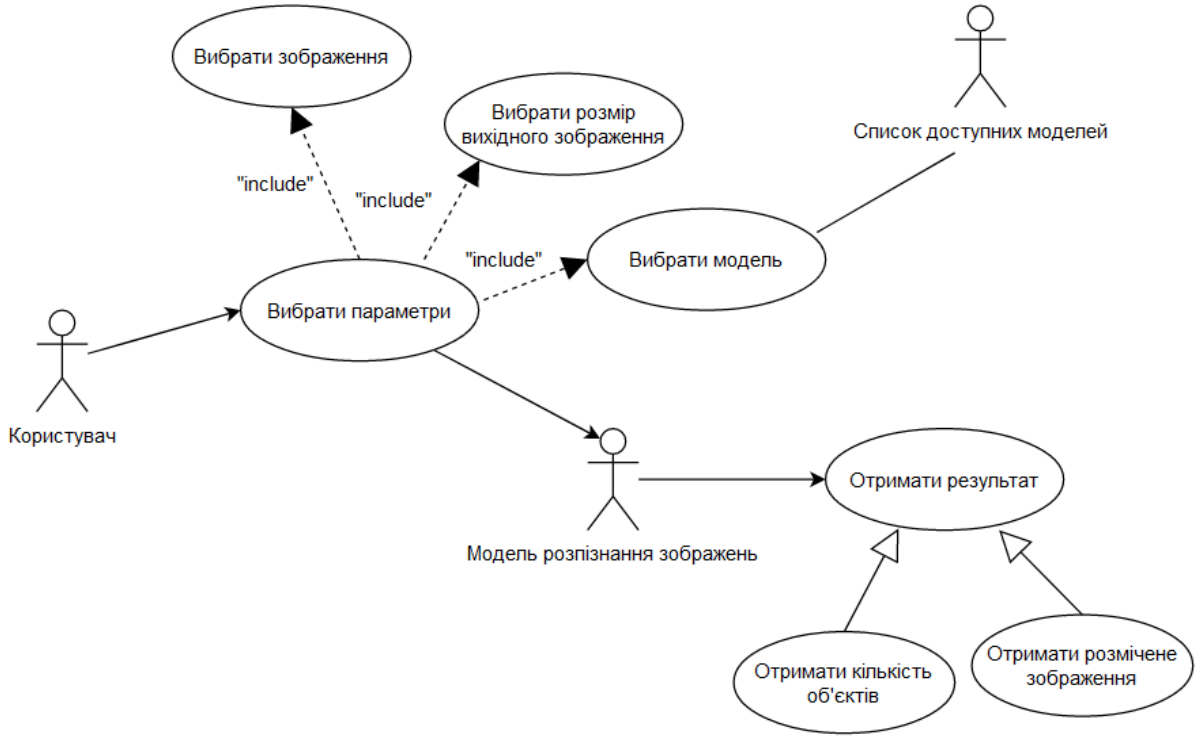

Рисунок 2.9 Діаграма взаємодії з застосунком

Для планування інтерфейсу програми було створено структурну схему додатку:

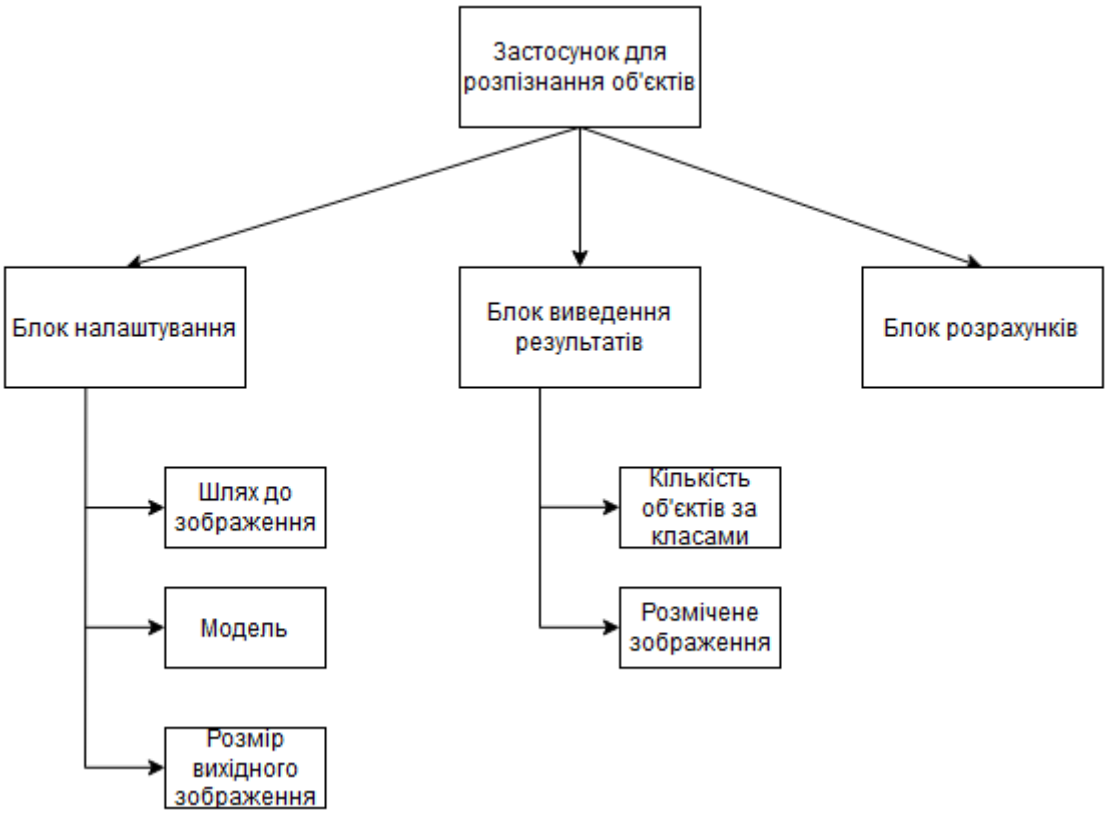

Рисунок 2.10 Структурна схема додатку

Для подальшого планування інтерфейсу було створено макет майбутнього додатку.

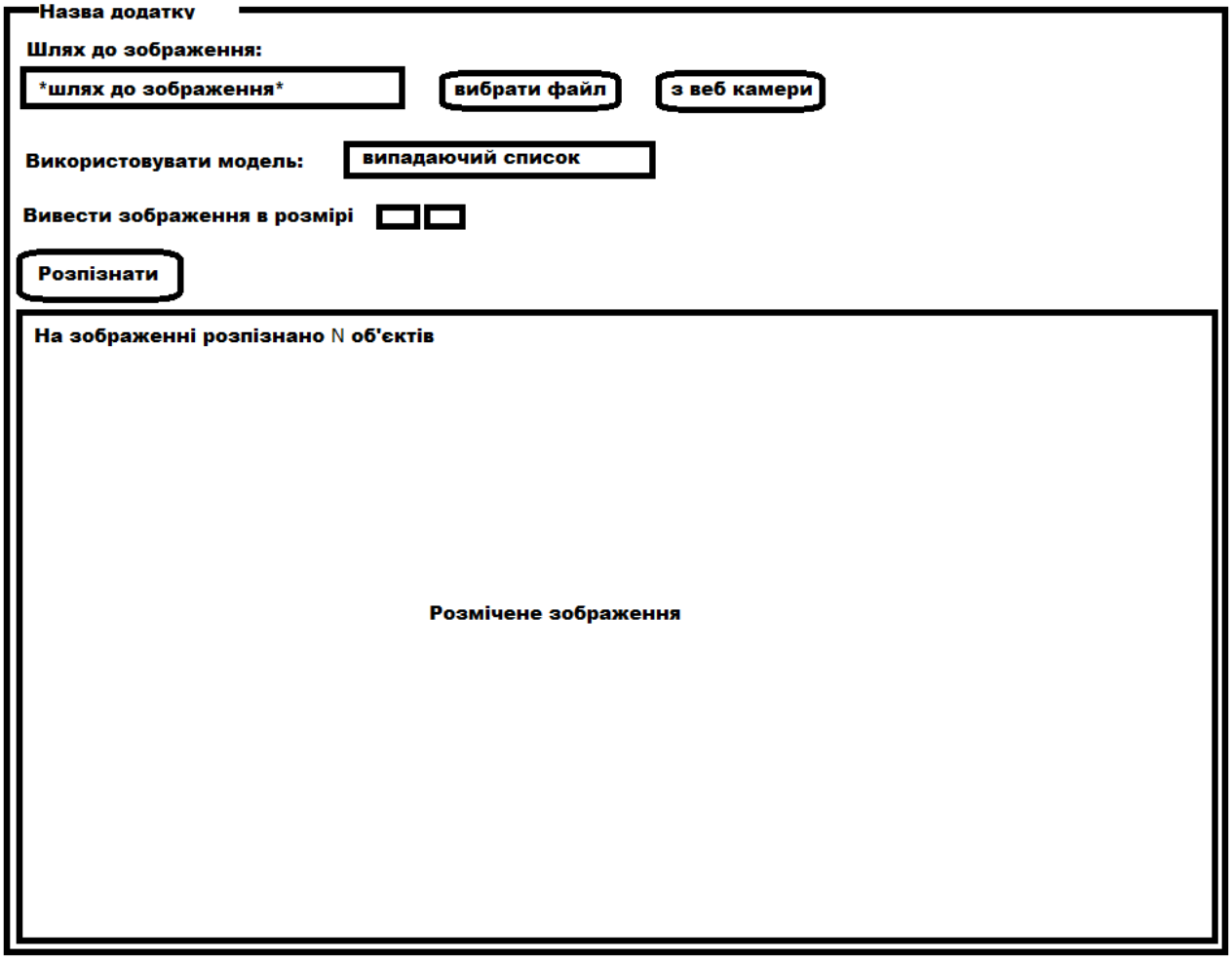

Рисунок 2.11 Макет застосунку.

Додаток повинен складатись з наступних полів:

1) Область налаштувань

- Поле вводу шляху до зображення
- Кнопка вибору файлу через провідник
- Кнопка отримання зображення з веб-камери
- Випадаючий список з завантаженими моделями
- Поле вводу вихідного зображення
- Кнопка для запуску розпізнавання

2) Область виведення результатів

- Текст з кількістю розпізнаних об’єктів
- Розмічене зображення

2.4 Розробка процесу навчання моделей

Основи для конфігурації моделей взяті з офіційних github сторінок моделей або з «саду моделей» бібліотеки tensorflow. Для навчання моделей конфігурації змінені, виставлена кількість класів 2 та розмір зображень 640x640. З огляду на кількість об’єктів при розмічені набору зображень та можливі їх збільшення при практичному застосуванні, для моделей було поставлено ліміт в 300 об’єктів на зображення.

При відсутності відеокарти з CUDA ядрами використовуються потужності процесора комп’ютера для навчання моделей.

Для покращення результатів проводиться аугментація зображень перед навчанням шляхом випадкового горизонтального повороту зображень (random horizontal flip).

Навчання моделей проводиться в наступні етапи:

1) Підготовка даних: перш ніж почати навчання, необхідно зібрати і підготувати дані для навчання. Це включає в себе створення розміченого набору даних, який складається із зображень і відповідних їм міток з класами об'єктів і координатами їхніх рамок. Для можливості використання для навчання такий набір даних було збережено в форматі Pascal VOC.

2) Визначення архітектури мережі: моделі для розпізнавання мають модульну архітектуру, що складається із серії блоків. Архітектура складається з основних блоків, таких як згорткові шари, пулінг, об'єднання та активаційні функції.

3) Ініціалізація ваг: перед початком навчання необхідно ініціалізувати ваги моделі. Для цього може використовуватись попередньо навчена модель, попередньо навчена на великому наборі даних.

4) Навчання: навчання моделі включає в себе оптимізацію функції втрат за допомогою алгоритму зворотного поширення помилки (backpropagation). Під час навчання модель аналізує зображення і передбачає класи та координати обмежувальних рамок об'єктів. Потім обчислюється функція втрат, яка порівнює

передбачені значення з істинними мітками. Градієнти функції втрат використовуються для оновлення ваг моделі з метою поліпшення передбачень.

5) Оптимізація параметрів: у процесі навчання моделі можуть бути визначені різні параметри моделі, такі як швидкість навчання, розмір пакета (batch size) і кількість епох (epochs). Оптимальні значення цих гіперпараметрів можуть бути знайдені за допомогою перебору та оцінювання результатів на валідаційному наборі даних.

6) Оцінка моделі: після завершення навчання модель оцінюють на тестовому наборі даних для вимірювання її продуктивності. В рамках цієї роботи оцінка проводиться за метрикою розпізнавання COCO (COCO detection metrics).

Рисунок 2.13 Життєвий цикл моделей машинного навчання

РОЗДІЛ 3. РЕАЛІЗАЦІЯ, ТЕСТОВИЙ ПРИКЛАД

3.1 Структура та специфікація модулів застосунку

Для відображення взаємодії внутрішніх компонентів системи було створено структурну схему (рисунок 3.1) та наведено специфікацію модулів (таблиця 3.1).

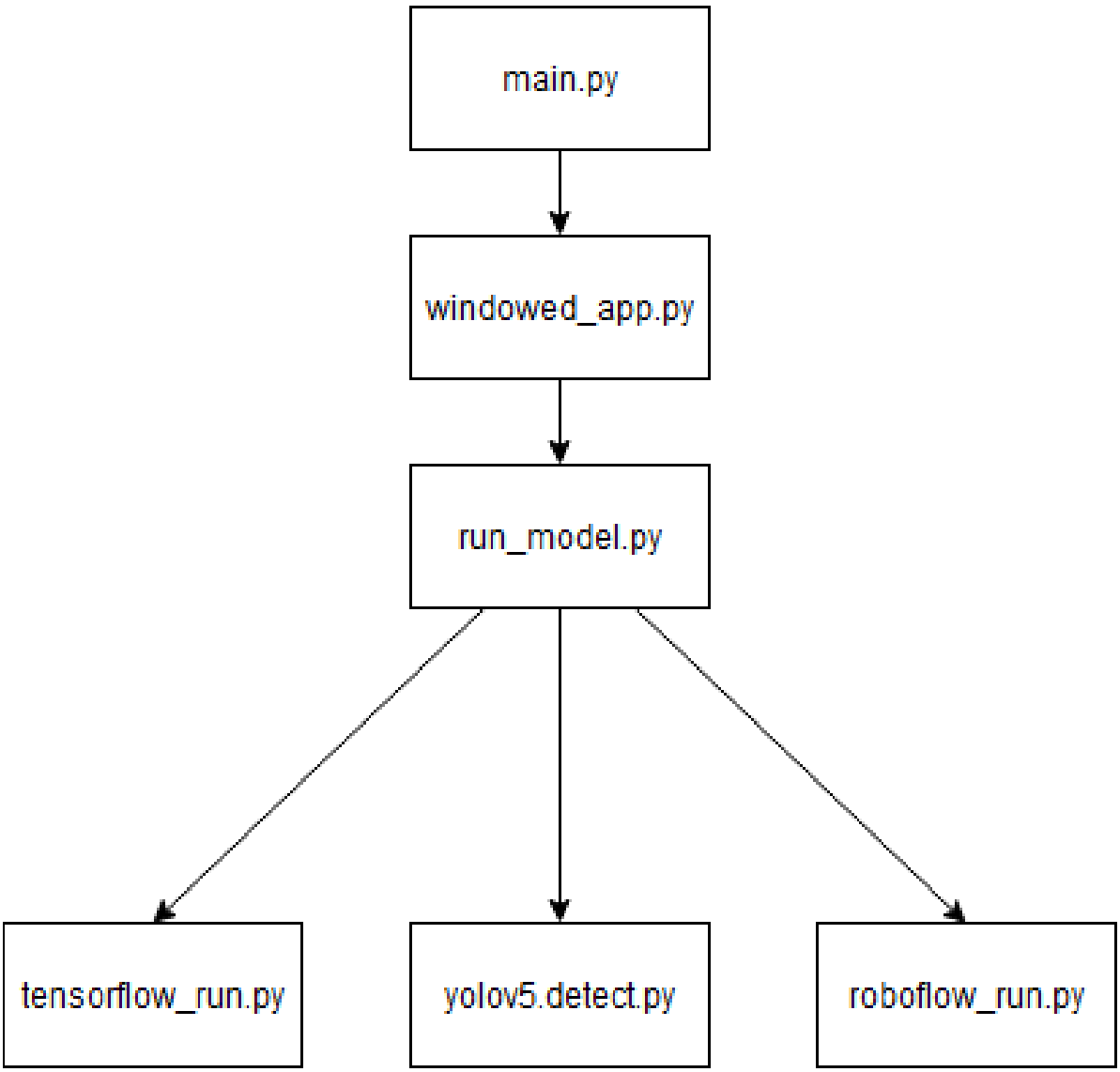

Рисунок 3.1 Структурна схема програмних модулів застосунку

Таблиця 3.1 Специфікація програмних модулів застосунку

main.py	Стартовий модуль, що запускає

	програму викликаючи модуль windowed_app
windowed_app.py	Модуль візуальної частини застосунку. Вміщує в себе будування інтерфейсу застосунку та завантаження моделей, функції отримання зображень з файлу та камери, та функцію виклику моделей розпізнавання для отримання та виведення результатів.
run_model.py	Модуль, що викликає відповідні функції для розпізнавання зображень відповідно до обраної моделі. Вхідні дані: ● модель ● назва моделі ● шлях до зображення ● шлях до назв класів
tensorflow_run.py	Проводить розпізнавання об’єктів та розмітку зображень за допомогою вибраної моделі збереженої за допомогою tensorflow.
yolo5.detect.py	Проводить розпізнавання об’єктів та розмітку зображень за допомогою моделі YOLO v5.
roboflow_run.py	Проводить розпізнавання об’єктів та

	розмітку зображень за допомогою моделі roboflow.

3.2 Програмний застосунок

Програмний інтерфейс складається з області налаштувань та області виведення результатів.

В області налаштувань у користувача є можливість вибрати файл, як вручну вписавши шлях до нього так і вибравши через провідник. Також надається можливість використати в якості зображення фотографію, створену через веб камеру.

Користувач може обрати одну з чотирьох завчасно натренованих моделей для розпізнавання зображень.

Надається можливість вибрати розмір зображення з розміченими об’єктами, яке буде виведене на екран.

В області виведення результатів після проведення операції розпізнавання виводиться загальна кількість розпізнаних об’єктів, а також використане зображення з розміченими положеннями всіх об’єктів на ньому.

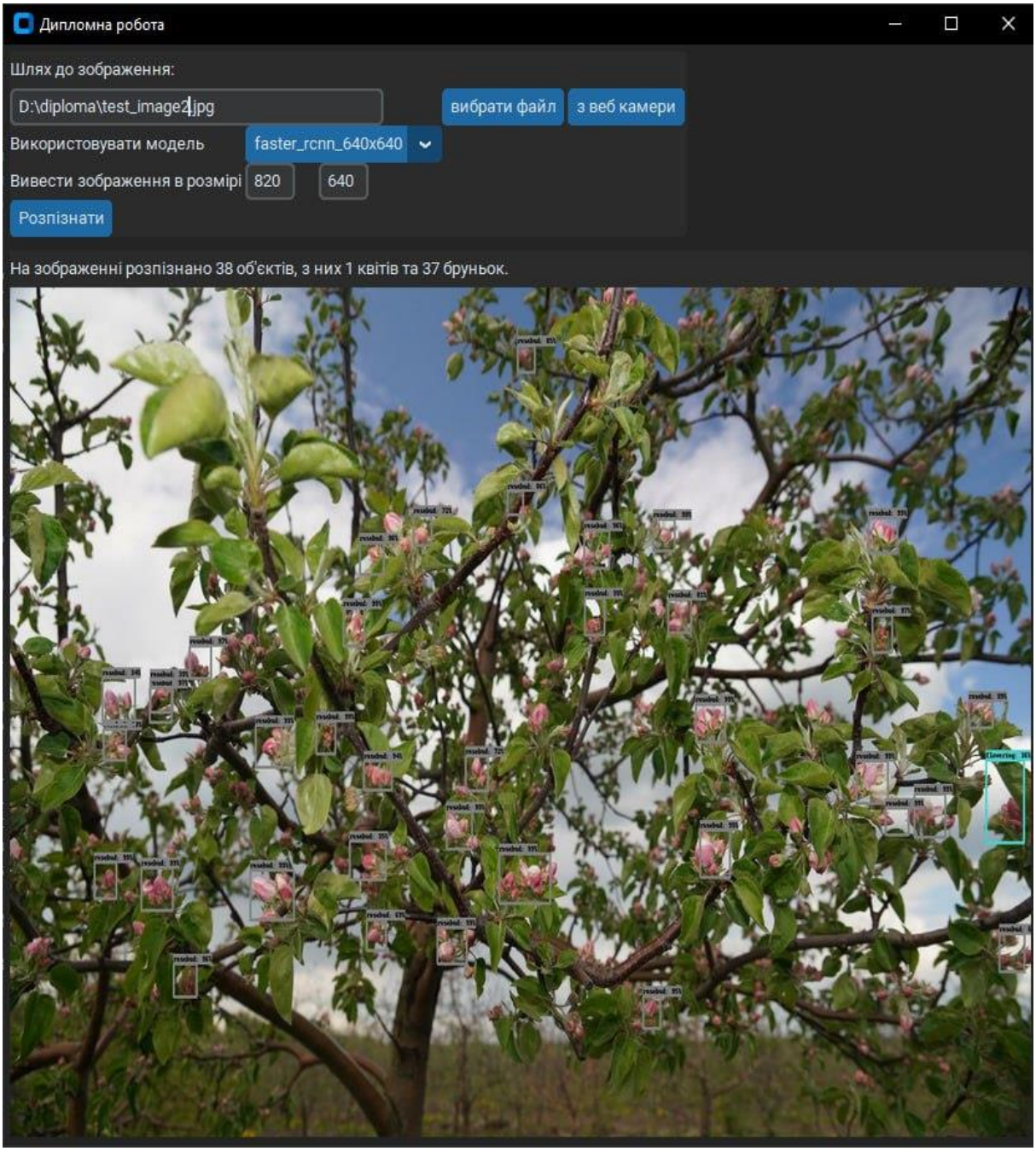

Рисунок 3.2 Інтерфейс програми

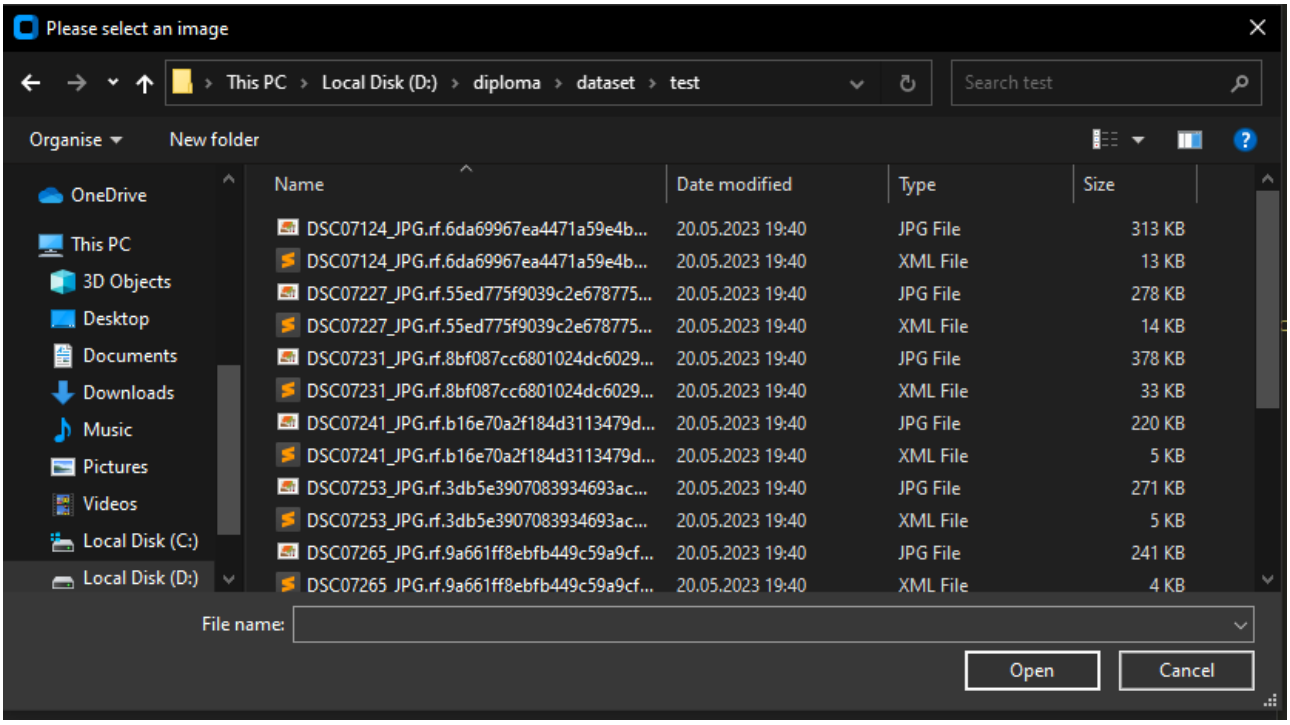

Рисунок 3.3 Інтерфейс провідника при виборі файлу

3.3 Відповідність функціональним вимогам

1) Локалізує та класифікує квіти на зображенні, обводячи їх

Програма за допомогою вибраної моделі виділяє області з розпізнаними об’єктами за допомогою прямокутників та підписує їх

2) Можливість завантажувати зображення з внутрішнього сховища або з камери

Програма надає можливість задавати шлях до зображення, збереженого локально, або створювати зображення за допомогою веб-камери.

3) Можливість вибирати різні моделі

На вибір користувачу даються 4 різні моделі для розпізнавання зображень.

4) Підтримка операційної системи Windows

Програма працює на платформі Windows 10 Home.

5) Можливість додавати моделі

Для додавання нової моделі у форматі збережених моделей tensorflow достатньо завантажити їх за відповідним шляхом та перезавантажити додаток, після чого користувач отримає можливість їх вибирати та використовувати.

3.4 Результати експериментальних досліджень

Було проведено валідацію моделей, для визначення точності при відсотку впевненості в 50%. Результати валідації наведені в таблиці 3.1. Валідація проводилась на зображеннях з навчального набору даних, що не були застосовані при навчанні моделі даних.

Таблиця 3.2 Порівняння точності моделей при валідації

Модель	Точність
SSD Resnet	0.330
Faster R-CNN	0.220
EfficientDet	0.269
Roboflow AutoML	0.4
YOLO v5	0.554

Проаналізовано темп навчання та зменшення похибки для моделей, а також витрачений на навчання час.

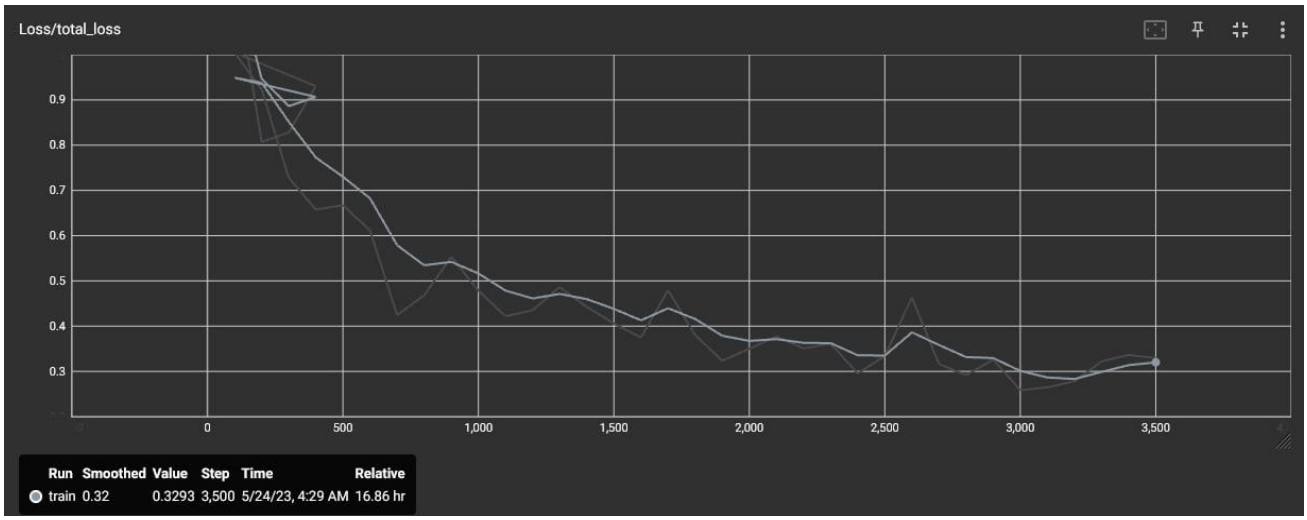

Рисунок 3.4 Динаміка зменшення похибки при навчанні моделі SSD Resnet

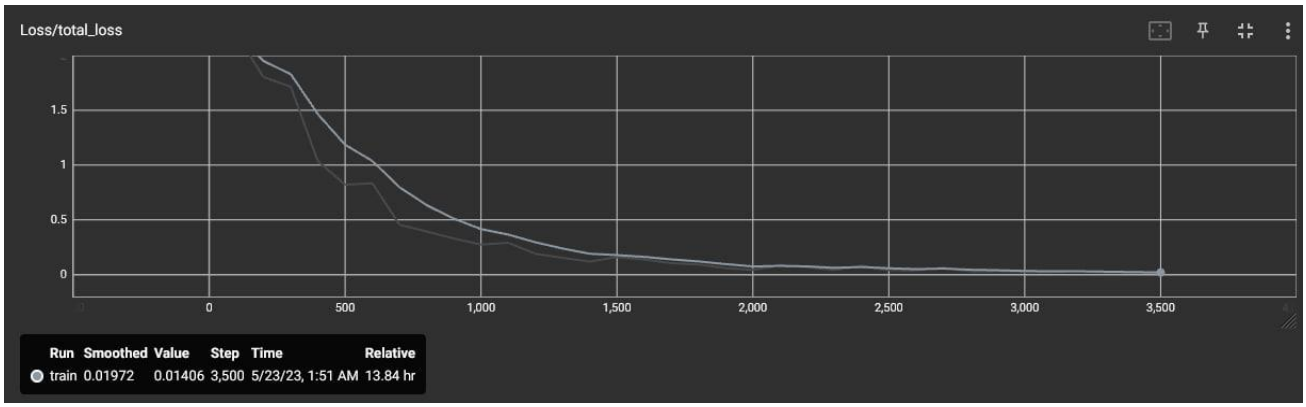

Рисунок 3.5 Динаміка зменшення похибки при навчанні моделі Faster R-CNN

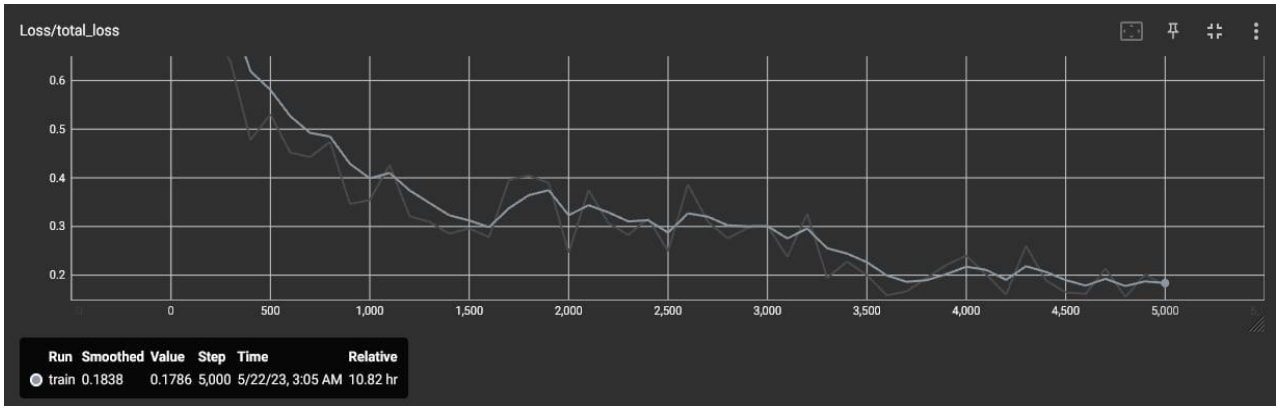

Рисунок 3.6 Динаміка зменшення похибки при навчанні моделі EfficientDet

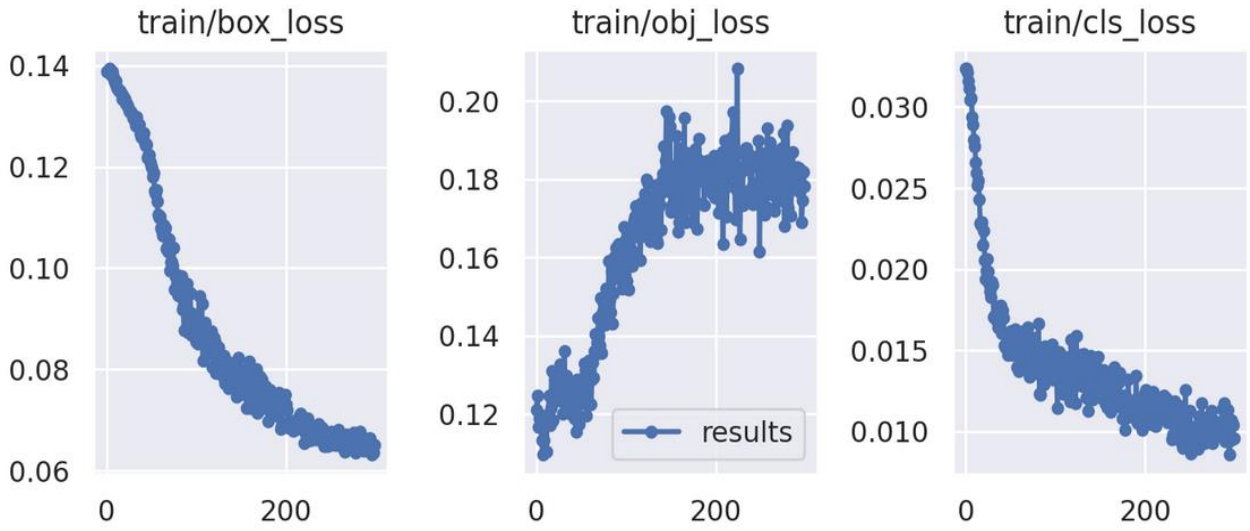

Рисунок 3.7 Динаміка зменшення похибки при навчанні моделі roboflow AutoML

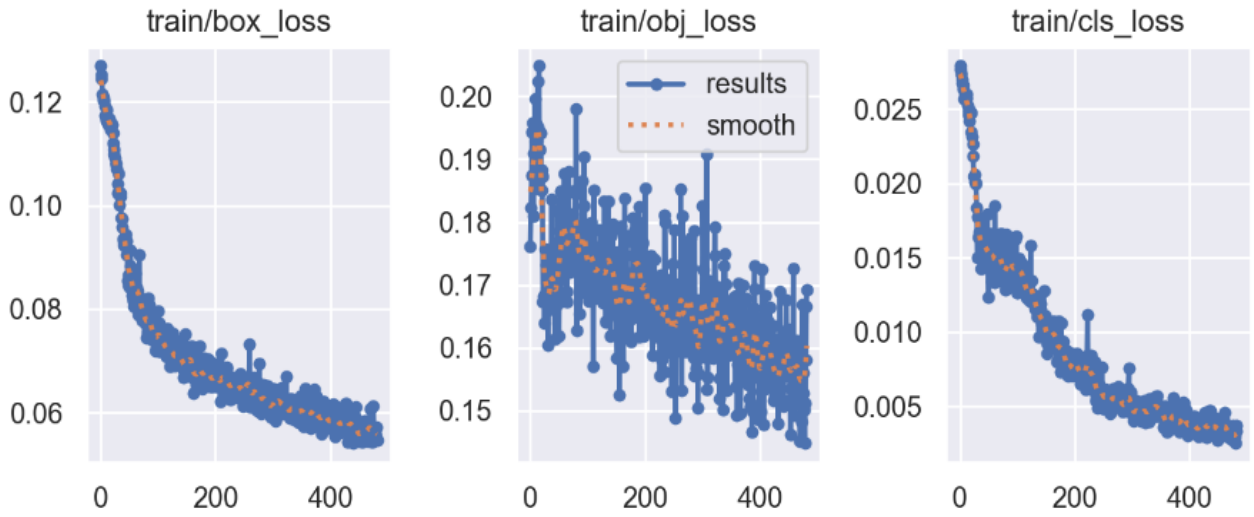

Рисунок 3.8 Динаміка зменшення похибки при навчанні моделі YOLO v5

Як видно з отриманих даних, лише модель Faster R-CNN досягла значень похибки в 0%, але показала при цьому найгірші результати при валідації. Це свідчить про перенавчання моделі на тренувальному наборі даних, і може бути покращено більш швидким зменшенням швидкості навчання моделі. Збільшення варіативності навчальної вибірки теж може покращити результат.

Навчання всіх моделей, крім roboflow AutoML, проводилось на процесорі AMD Ryzen 5 5600.

Навчання моделі roboflow AutoML проводилось в хмарному середовищі з доступом до більш потужних обчислювальних потужностей, що пояснює різницю в часі навчання. Загалом кореляції між часом навчання різних моделей та їх точністю немає.

Результати, наведені в таблиці 3.2, показують, що навчання моделі навіть на невеликих розмірах зображень може займати до ¾ доби, через що можна припустити збільшення часу навчання до декількох діб при використанні full HD або більших розмірів зображень.

Таблиця 3.3 Порівняння швидкості навчання моделей

Модель	Час на навчання, годин
SSD Resnet	16.86
Faster R-CNN	13.84

EfficientDet	10.82
Roboflow AutoML	0.43
YOLO v5	18.04

3.5 Перевірка точності моделей в реальних умовах

Для кращої перевірки перспективності моделей при використанні в реальних умовах було взято три зображення не з набору даних для навчання, та проведено розпізнавання на впевненості в 30%. Використані зображення створені при різному освітленні та мають різні кольорові гами, які відрізняються від таких у навчальному наборі даних, що робить експеримент більш наглядним. В цілях експерименту враховувались лише об’єкти, що знаходяться на передньому плані.

Рисунок 3.9 Зображення для перевірки

В ході експерименту було отримано наступні результати, відображені в таблиці 3.3:

Таблиця 3.4 Порівняння точності моделей на зображеннях не з тренувального набору даних

Модель	Точність
SSD Resnet	0.77
Faster R-CNN	0.80
EfficientDet	0.82

Roboflow AutoML	0.57
YOLO v5	0.88

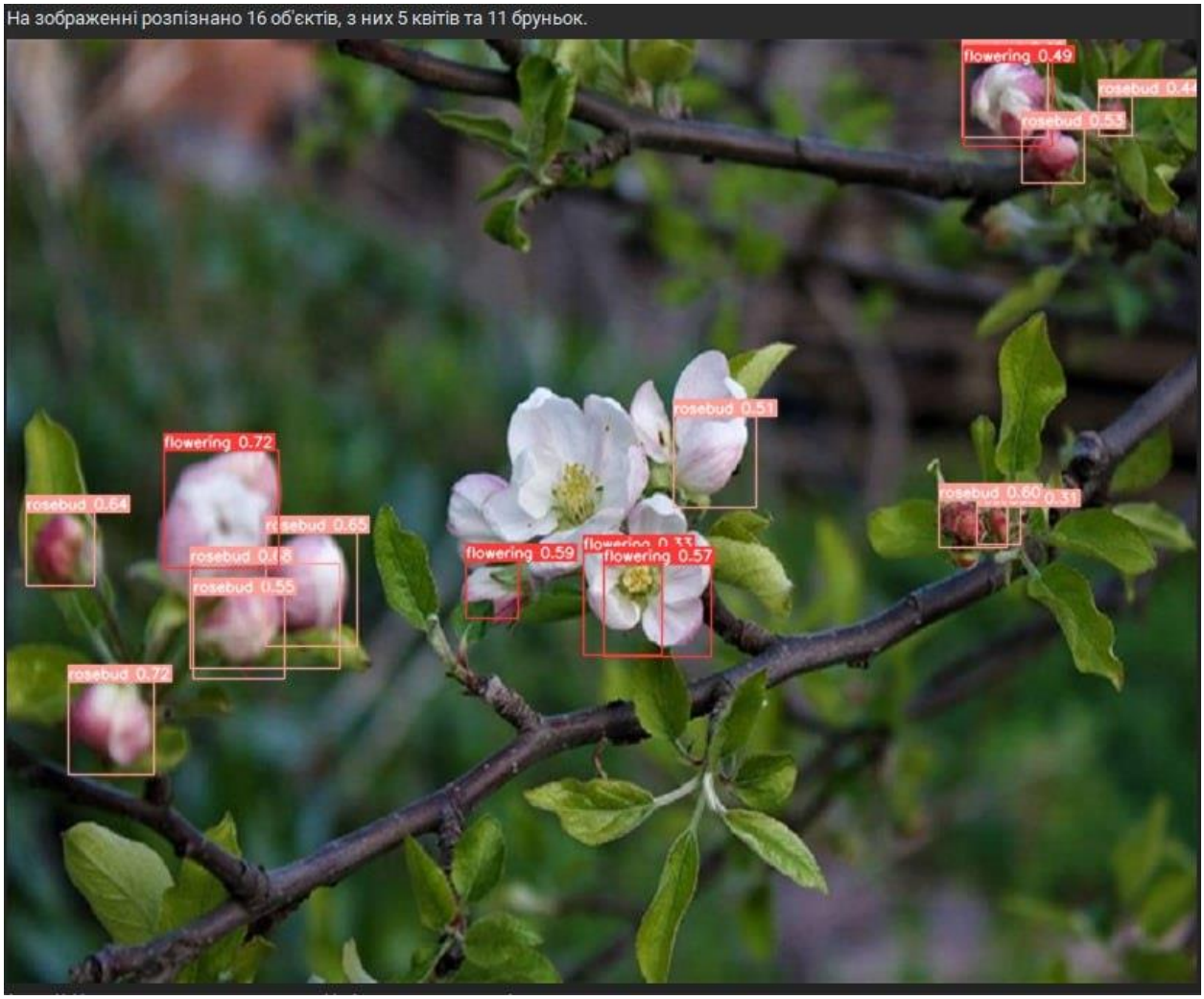

Рисунок 3.10 Приклад розпізнавання зображення з експериментального набору

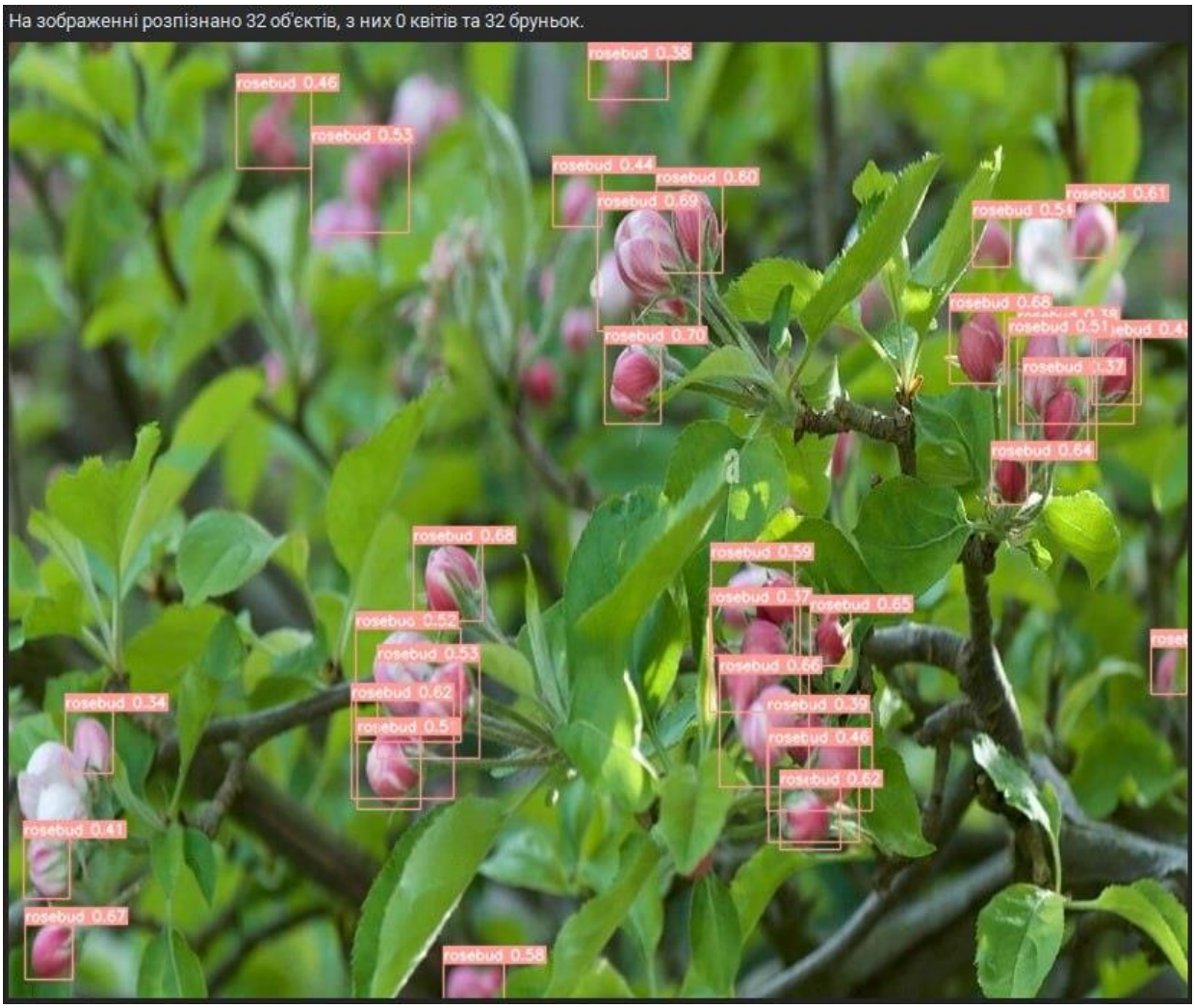

Рисунок 3.11 Приклад розпізнавання зображення з більшою кількістю об’єктів

Рисунок 3.12 Приклад розпізнавання зображення яблуні без квітів та бруньок

Згідно отриманим експериментальним даним, натреновані моделі показують досить високу точність при використанні в реальних умовах. Через природу навчальних та експериментальних даних, під час експерименту була частою ситуація коли один об’єкт часто був розпізнаний більше ніж один раз – це зумовлено різною відстанню між об’єктами та камерою на них.

В ході експерименту моделі також розпізнавали об’єкти, що були на фоні і не в фокусі зображення. Такі розпізнавання не враховувались для точності моделей, але можуть призвести до їх меншої цінності в якості збірників інформації. Запобігти цього можна було б додатковим шаром архітектури, що відсікав би частини зображення не в фокусі, залишаючи лише основне. Таким

чином, умовний робот міг би проходити від дерева до дерева, збираючи інформацію про кожне окремо, без втручання випадкових даних з других дерев.

ВИСНОВКИ

1. Майже всі моделі показують досить високу точність (>70%) для виконання практичних задач в аграрній промисловості та для збору інформації.

2. Найкращі результати показала модель YOLO v5, яка в той же час потребувала найбільше часу на навчання.

3. Автоматичне машинне навчання, хоча і є найбільш доступним для неспеціалістів засобом для використання розпізнавання об’єктів, не дає задовільні результати.

4. Проблема перенавчання може з’явитись навіть при використанні поступового зменшення швидкості навчання.

5. Більше різноманіття особливостей зображень в наборі даних дозволяє отримати кращі результати в умовах практичного використання.

6. Подальшого покращення можна досягти відсіканням об’єктів на фоні не в фокусі зображення.

ДЖЕРЕЛА

1. U.S. DEPARTMENT OF AGRICULTURE. (2023). *Sugar and sweeteners outlook: May 2023*. USDA ERS - Home. https://www.ers.usda.gov/
2. Ma, L., Liu, Y., Zhang, X., Ye, Y., Yin, G. & Johnson, B. (2019). Deep learning in remote sensing applications: A meta-analysis and review. ISPRS Journal of Photogrammetry and Remote Sensing. 152. 166-177. 10.1016/j.isprsjprs.2019.04.015.
3. Lecun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., & Jackel, L. D. (1989). Backpropagation applied to handwritten zip code recognition. *Neural Computation, 1*(4), 541-551.
4. Unzueta, D. (2022). *Convolutional layers vs fully connected layers*. Medium. https://towardsdatascience.com/convolutional-layers-vs-fully-connected-layers-364f05ab460b
5. Smirnov, I., Kutyrev, A., & Kiktev, N. (2021). *Neural network for identifying Apple Fruits on the crown of a Tree*. E3S Web of Conferences. https://www.e3s-conferences.org/articles/e3sconf/abs/2021/46/e3sconf_wfces2021_01021/e3sconf_wfces2021_01021.html
6. Khort, D., Kutyrev, A., Smirnov, I., Osypenko, V., & Kiktev, N. (2020). *Computer Vision System for recognizing the coordinates location and ripeness of strawberries*. Semantic Scholar. https://www.semanticscholar.org/paper/Computer-Vision-System-for-Recognizing-the-Location-Khort-Kutyrev/f0f8ab507ed1efcef8f9e8f1c1b67c86776ddf4b
7. Oh, S., Chang, A., Ashapure, A., Jung, J., Dube, N., Maeda, M., Gonzalez, D., & Landivar, J. (2020). *Plant counting of cotton from UAS imagery using deep learning-based object detection framework*. MDPI. https://www.mdpi.com/2072-4292/12/18/2981
8. Koval, B., & Khlevna, I. (2022). *Anomalies Analysis and Detection Using Computer Vision for Finding Defects in Plant Leaves Images*. CEUR Workshop Proceedings. https://ceur-ws.org/Vol-3347/Paper_6.pdf

9. Kutyrev, A., Kiktev, N., Kalivoshko, O., & Rakhmedov, R. (2022). *Recognition and Classification Apple Fruits Based on a Convolutional Neural Network Model*. CEUR Workshop Proceedings. https://ceur-ws.org/Vol-3347/Paper_8.pdf
10. Kong, S., Li, J., Zhai, Y., Gao, Z., Zhou, Y., & Xu, Y. (2023). Realtime detection of crop with dense planting using deep 2 learning at seedling stage. *Agronomy MDPI*.
11. Garibaldi-Márquez, F., Flores, G., Mercado-Ravell, D. A., Ramírez-Pedraza, A., & Valentín-Coronado, L. M. (2022). Weed classification from natural corn field-multi-plant images based on shallow and deep learning. *Sensors*, *22*(8), 3021. https://doi.org/10.3390/s22083021
12. Thakur, P. S., Sheorey, T., & Ojha, A. (2022). VGG-ICNN: A lightweight CNN model for Crop disease identification. *Multimedia Tools and Applications*, *82*(1), 497–520. https://doi.org/10.1007/s11042-022-13144-z
13. Chen, J., Wang, H., Zhang, H., Luo, T., Wei, D., Long, T., & Wang, Z. (2022). Weed detection in Sesame Fields using a YOLO model with an enhanced attention mechanism and feature fusion. *Computers and Electronics in Agriculture*, *202*, 107412. https://doi.org/10.1016/j.compag.2022.107412
14. Hu, H., Kaizu, Y., Zhang, H., Xu, Y., Imou, K., Li, M., Huang, J., & Dai, S. (2022). *Recognition and localization of strawberries from 3D binocular cameras for a strawberry picking robot using coupled YOLO/mask R-CNN*. International Journal of Agricultural and Biological Engineering. https://ijabe.org/index.php/ijabe/article/view/7306
15. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., & Houlsby, N. (2021). *An image is worth 16x16 words: Transformers for image recognition at scale*. arXiv.org. https://doi.org/10.48550/arXiv.2010.11929
16. Foody, G. M., & Arora, M. K. (1997). An evaluation of some factors affecting the accuracy of classification by an artificial neural network. *International*

Journal of Remote Sensing, *18*(4), 799–810. https://doi.org/10.1080/014311697218764

17. Li, W., & Hsu, C.-Y. (2018). Automated terrain feature identification from remote sensing imagery: A deep learning approach. *International Journal of Geographical Information Science*, *34*(4), 637–660. https://doi.org/10.1080/13658816.2018.1542697
18. Girshick, R. (2015). *Fast R-CNN*. arXiv.org. https://arxiv.org/abs/1504.08083
19. Ren, S., He, K., Girshick, R., & Sun, J. (2016). *Faster R-CNN: Towards real-time object detection with region proposal networks*. arXiv.org. https://arxiv.org/abs/1506.01497
20. Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). *You only look once: Unified, real-time object detection*. arXiv.org. https://arxiv.org/abs/1506.02640
21. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., & Berg, A. C. (2016). *SSD: Single shot multibox detector*. arXiv.org. https://arxiv.org/abs/1512.02325
22. Tan, M., Pang, R., & Le, Q. V. (2020). *EfficientDet: Scalable and efficient object detection*. arXiv.org. https://arxiv.org/abs/1911.09070
23. Liu, B. (2018). *A very brief and critical discussion on Automl*. arXiv.org. https://arxiv.org/abs/1811.03822