МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

ІМЕНІ ТАРАСА ШЕВЧЕНКА

Факультет інформаційних технологій

Кафедра інтелектуальних технологій

ВИПУСКНА КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА

НА ТЕМУ

«Програмний модуль розпізнавання об'єктів у промисловому фруктовому саду за допомогою згорткової нейромережі»

Галузь знань **12 «Інформаційні технології»**

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

Освітня програма **«Комп'ютерні науки»**

Освітній рівень: **бакалавр**

Виконав: студент 4 курсу, групи КН-41
спеціальності – 122 «Комп'ютерні науки»

Сопов Сергій Сергійович
(прізвище та ініціали)

Керівник к.т.н., доц. Кіктєв М. О.
(наук. ступінь, звання, прізвище та ініціали)

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

КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

ІМЕНІ ТАРАСА ШЕВЧЕНКА

Факультет інформаційних технологій

Кафедра інтелектуальних технологій

Спеціальність 122 «Комп'ютерні науки»

ЗАТВЕРДЖУЮ
Зав. кафедри інтелектуальних технологій

к.т.н., доц. Іларіонов О.Є.
(звання, прізвище та ініціали)

(підпис)

« ____ » _____ 2023 р.

З А В Д А Н Н Я
НА ВИПУСКНУ КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТОВІ

Сопову Сергію Сергійовичу
(прізвище, ім'я, по батькові)

1. Тема роботи: "Програмний модуль розпізнавання об'єктів у промисловому фруктовому саду за допомогою згорткової нейромережі "

затверджена наказом ректора від " 11 " листопада 2022 року № 4

2.Термін виконання проекту (роботи):): 31 травня 2023 року

3. Вихідні дані до роботи: близько 500 зображень з фруктами з відкритих баз (Fruits-360, Banana Ripeness Dataset)

4. Зміст пояснювальної записки (перелік питань, що підлягають розробці):

1) аналіз процесу розпізнавання плодів;

2) розробка архітектури системи розпізнавання плодів;

3) реалізація модуля по розпізнаванню об'єктів у промисловому фруктовому саду.

5. Перелік презентаційного матеріалу (з точним зазначенням обов'язкових презентацій):

мета, об'єкт та предмет (1 слайд), наглядний принцип роботи згорткові нейронні мережі (1 слайд), Порівняння існуючих додатків по роспізнаванню фруктів (1 слайд), Вимоги до програмного модуля (1 слайд), Дерево функцій системи (1 слайд), Контекстна діаграма модулю розпізнавання плодів (1 слайд), Концептуальна модель БД (1 слайд), Логічна і фізична структури моделі даних (1 слайд), Архітектура системи (1 слайд), Проєктування інтерфейсу програмного модуля (1 слайд), Стартове вікно (1 слайд), Відкриття додатком сховища і результат (1 слайд), Приклади роботи системи (2 слайди), Навчання НМ (3 слайди), висновки (1 слайд).

6. Консультанти з випускної кваліфікаційної роботи із зазначенням її розділів, що їх стосуються

Розділ	Консультант	Завдання видав	Завдання прийняв
1	Кіктєв М.О.	01.03.2022	01.03.2022
2	Кіктєв М.О.	05.04.2022	05.04.2022
3	Кіктєв М.О.	10.05.2022	10.05.2022

7. Дата видачі завдання 15 лютого 2023 року

Керівник випускної кваліфікаційної роботи / М.О. Кіктєв /
(підпис) (ініціали та прізвище)

Завдання прийняв до виконання / С.С. Сопов /
(підпис) (ініціали та прізвище)

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Назва етапів випускної кваліфікаційної роботи	Термін виконання етапів	Примітка
1	Обговорення постановки завдання та змісту пояснювальної записки	25.01.2023 -22.02.2023	
2	Вибір та формування теми	23.02.2023 - 26.02.2023	
3	Аналіз предметної області	27.02.2023 – 10.03.2023	
4	Вибір методів рішення задачі	11.03.2023 –20.04.2023	
5	Створення програмного модулю	21.04.2023 -10.05.2023	
6	Оформлення пояснювальної записки	11.05.2023 – 29.05.2023	

Керівник випускної кваліфікаційної роботи / М.О. Кіктєв /
(підпис) (ініціали та прізвище)

Студент / С.С. Сопов /
(підпис) (ініціали та прізвище)

Анотація

Сопов Сергій Сергійович виконав випускну кваліфікаційну роботу на тему “Програмний модуль розпізнавання об'єктів у промисловому фруктовому саду за допомогою згорткової нейромережі” за спеціальністю 122 – «Комп’ютерні науки».

Програмний модуль дозволяє на зображенні розпізнати плоди (банани, апельсини та яблука), визначити їх стан та кількість.

Модуль використовує алгоритми розпізнавання, що базуються на характеристиках форми, кольору та текстури плодів, а також алгоритми підрахунку.

Об'єкт дослідження: процес розпізнавання об'єктів у промисловому фруктовому саду для використання у робототехнічній системі.

Предмет дослідження: моделі, методи, інформаційні технології та згорткова нейронна мережа для розпізнавання фруктів у промисловому саду

Мета дослідження: підвищення ефективності розпізнавання фруктів та оцінювання їх якості у промислових фруктових садах.

Переваги даного програмного модуля включають надійну та швидку роботу, точне розпізнавання плодів, а також точний підрахунок їх кількості.

Ключові слова: програмний модуль, розпізнавання об'єктів, плоди, промисловий фруктовий сад, згорткова нейромережа, фрукти, стиглість, спорченість, автоматизація, якість продукції.

Summary

Sopov Serhii Sergiyovych completed his qualification work on the topic "Software module for object recognition in an industrial orchard using a convolutional neural network" in the specialty 122 – "Computer Science."

The software module allows for the recognition of fruits (bananas, oranges, and apples) in images and determines their condition and quantity. The module utilizes recognition algorithms based on shape characteristics, color, and texture of the fruits, as well as counting algorithms.

Research object: Object recognition process in an industrial fruit orchard for use in a robotics system.

Research subject: Models, methods, information technologies, and convolutional neural network for fruit recognition in an industrial orchard.

Research goal: To enhance the efficiency of fruit recognition and quality assessment in industrial fruit orchards.

The advantages of this software module include reliable and fast operation, accurate fruit recognition, and precise counting of their quantity.

Keywords: software module, object recognition, fruits, industrial fruit orchard, convolutional neural network, fruit, ripeness, spoilage, automation, product quality.

ЗМІСТ

Перелік умовних позначень і скорочень

HTML (HyperText Markup Language) – стандартизована мова розмітки веб-сторінок;

API (Application Programming Interface) – прикладний програмний інтерфейс;

XML (*Extensible Markup Language*) – розширена мова розмітки. Текстовий формат для обміну даними;

SQL (*Structured Query Language*) – мова структурованих запитів. Для взаємодії з базами даних;

UI (*User interface*) – графічний інтерфейс користувача;

MVC (*Model-View-Contoller*) – паттерн веб програмування. Складається з моделі сутності, візуального представлення та контроллеру виконання логіки;

SPA (*Single-page application*) – односторінковий додаток. Паттерн веб-програмування;

Вступ

Актуальність теми. У сучасному світі розвиток технологій вимагає постійного вдосконалення і модернізації об'єктів розпізнавання. В рамках цього контексту, дипломна робота присвячена розробці програмного модуля для розпізнавання об'єктів у промисловому фруктовому саду за допомогою згорткової нейромережі.

Головною метою даної роботи є розробка ефективної системи, яка забезпечує точне розпізнавання та класифікацію плодів, зокрема бананів, апельсинів та яблук, а також визначення їх стану залежно від стиглості та розрахунок їх кількості. Такий програмний модуль відіграє важливу роль у промисловому фруктовому саді, де автоматичне розпізнавання та збір врожаю за допомогою робототехніки стає все більш актуальним.

Проектні рішення, що лягли в основу розробки даного модуля, обґрунтовані на основі характеристик форми, кольору та текстури плодів. Для досягнення високої точності розпізнавання та підрахунку, використані надійні алгоритми, засновані на згорткових нейромережах. Такий підхід дозволяє забезпечити стабільну та ефективну роботу модуля навіть у випадку відмінностей у формі, кольорі та розмірі плодів.

Результати даної роботи можуть бути застосовані у різних галузях, включаючи сільське господарство, логістику та збір врожаю за допомогою робототехніки. Впровадження цього програмного модуля дозволить автоматизувати процеси розпізнавання, підрахунку та контролю якості фруктів, збільшити ефективність роботи промислових фруктових садів та знизити витрати на працю.

Актуальність обраної теми випливає з необхідності покращення модернізації промислових фруктових садів та використання передових технологій для збільшення врожайності та оптимізації процесів виробництва. Розробка модуля по розпізнаванню об'єктів, визначенню стану стиглості та кількості фруктів у промислових фруктових садах в рамках систем

автоматичного збору врожаю є актуальною з погляду підвищення продуктивності та ефективності вирощування фруктів. Традиційні методи збору вимагають великої кількості ручної праці та витрачання великого об’єму часу. Введення систем автоматичного збору врожаю, що використовують згорткові нейромережі для розпізнавання, надає фермерам і операторам саду важливу інформацію для планування збору врожаю, оптимізації процесу збору та прогнозування виробничих потреб.

Розробка даного модуля відкриває перспективи вдосконалення процесу збору врожаю, зниження витрат і підвищення якості продукції у промислових фруктових садах. Враховуючи спектр підтримуваних фруктів, можливість визначення стану стиглості та кількості фруктів, цей модуль стає потужним інструментом для оптимізації вирощування фруктів та збільшення виробничої потужності саду.

Розділ 1
АНАЛІЗ ПРОЦЕСУ РОЗПІЗНАВАННЯ ОБ'ЄКТІВ НА ЗОБРАЖЕННЯХ

1.1 Аналіз існуючих методів розпізнавання об'єктів на зображеннях

Розпізнавання об'єктів за допомогою нейронних мереж є однією з актуальних тем у галузі комп'ютерного зору і машинного навчання. Ця технологія дозволяє автоматично визначати та класифікувати різні види об'єктів на основі їх зовнішніх ознак, таких як форма, колір, текстура тощо.

Для розпізнавання об'єктів застосовуються різні типи нейронних мереж, зокрема згорткові нейронні мережі (CNN) та комбіновані мережі. Згорткові нейронні мережі добре справляються з виявленням візуальних ознак об'єктів на зображеннях, тоді як комбіновані нейронні мережі дозволяють моделювати залежності між різними частинами фрукта [1].

Основні етапи розпізнавання об'єктів за допомогою нейронних мереж включають підготовку навчального набору даних, тренування мережі на цих даних, та використання тренованої мережі для розпізнавання фруктів на нових зображеннях. Важливим кроком є надання моделі достатнього розмаїття тренувальних даних, які включають різні види фруктів з різними характеристиками.

Розпізнавання об'єктів нейронними мережами має різноманітні застосування, зокрема в сільському господарстві, виробництві харчових продуктів, сортуванні та автоматизації процесів збору врожаю. Відповідно, розробка точних та ефективних моделей розпізнавання об'єктів має великий потенціал для покращення продуктивності та оптимізації виробничих процесів у фруктових садах та промислових господарствах.

Розпізнавання об'єктів за допомогою нейронних мереж відкриває широкі можливості для автоматизації та оптимізації процесів у сфері вирощування та збору врожаю фруктових культур. Це дозволяє досягти ряду переваг і

покращити результати у порівнянні з традиційними методами розпізнавання та класифікації фруктів.

Однією з головних переваг є висока точність розпізнавання, що досягається завдяки потужним обчислювальним здібностям нейронних мереж та їх здатності до виявлення складних візуальних шаблонів та залежностей між ознаками об’єктів. Це дозволяє надійно ідентифікувати різні види фруктів та відрізняти їх від інших об'єктів [2].

Крім того, застосування нейронних мереж дозволяє визначати стан стиглості фруктів, що є важливим параметром в аграрному секторі. Нейронні мережі можуть аналізувати зображення фруктів та визначати їх рівень зрілості на основі кольору, текстури або інших характеристик. Це допомагає виробникам та садівникам визначати оптимальний час для збирання врожаю та уникнути втрат через неправильну стиглість.

Також розпізнавання об’єктів нейронними мережами може бути використане для визначення кількості фруктів на дереві або у певній області саду. Це дає змогу автоматично підраховувати врожайність та оцінювати рівень виробництва. Така інформація є цінною для планування виробничих процесів, розподілу ресурсів та прийняття рішень з управління садом.

Загалом, розпізнавання об’єктів нейронними мережами сприяє автоматизації та покращенню ефективності сільськогосподарських процесів, забезпечуючи точність, швидкість та надійність в розпізнаванні, визначенні стану та кількості фруктів у промислових фруктових садах.

Актуальність застосування розпізнавання об’єктів за допомогою нейронних мереж у сфері робототехніки зростає з кожним роком. Робототехніка в сільському господарстві має великий потенціал для автоматизації різних процесів, включаючи збір врожаю. Розпізнавання об’єктів з використанням нейронних мереж є ключовим елементом у розвитку цієї області.

Однією з основних переваг роботів у зборі врожаю є їхній потенціал для заміни ручної праці та забезпечення ефективного та швидкого збору фруктів.

Розпізнавання об’єктів нейронними мережами дозволяє роботам точно ідентифікувати фрукти та визначати їх стан і кількість. Це забезпечує автоматичне та ефективне збирання врожаю без необхідності великої людської праці.

Крім того, розпізнавання об’єктів нейронними мережами допомагає покращити якість та різноманітність можливих типів збору врожаю. Роботи можуть відрізняти фрукти за їхньою стиглістю та якістю, що дозволяє виробникам збирати лише оптимально зрілі фрукти, покращуючи якість та тривалість зберігання.

Також, розпізнавання фруктів нейронними мережами сприяє зменшенню втрат врожаю. Роботи можуть виявляти хвороби, пошкодження та інші недоліки на фруктах, що дозволяє вчасно вжити заходів для їхнього усунення та запобігання поширенню захворювань. Це забезпечує економію ресурсів та покращує продуктивність промислових фруктових садів.

Застосування розпізнавання фруктів нейронними мережами у робототехніці є актуальним напрямом розвитку, який сприяє автоматизації та покращенню ефективності сільськогосподарських процесів.

1.1.1 Згорткові нейронні мережі

Для візуального аналізу широко використовується CNN. CNN, як правило, розглядаються як нейронні мережі прямого поширення, які можуть швидко ідентифікувати, класифікувати та розпізнавати будь-які особливості на зображенні. Першу CNN, широко відому як LeNet, створив Ян Лекун у 1988 році, член дослідницької групи зі штучного інтелекту у Facebook [4].

У CNN вхідні дані мережі складаються зі значень пікселів зображення значень пікселів зображення, що мають різну вагу залежно від ознаки, яку потрібно виділити, як визначено в прихованому шарі. На вхідному зображенні CNN також складаються з повністю з'єднаних шарів для розпізнавання різних об'єктів, незважаючи на шар об'єднання та згортки (рис. 1.1).

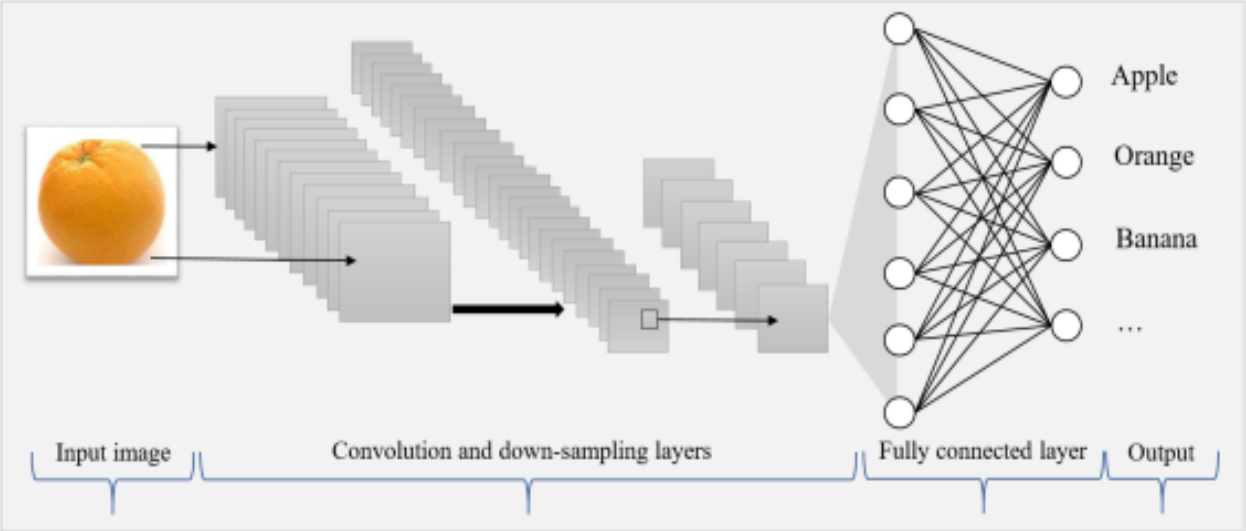


Рисунок 1.1 – Наглядний принцип роботи згорткові нейронні мережі

Операція згортки виконується в класифікаторі CNN над пікселями зображення. Вона складається з чотирьох найпоширеніших шарів. Перший – це шар згортки, завданням якого є згортка пікселів на зображенні з обраним ядром для вилучення або видалення різних особливостей. Другий – це шар ReLU, який визначає функцію активації, що може бути сигмоїдною або будь-якою нелінійною функцією.

Зображення проходить кілька разів між шарами згортки та ReLU, де всі від'ємні пікселі перетворюються на нуль і аналізуються тенденції та атрибути зображення. Третій шар відомий як шар Pooling (об'єднання), і основне призначення цього шару – трансформувати зображення до необхідного розміру без розмиття необхідного розміру, не розмиваючи його. Для цього шар об'єднання містить різні ядра для визначення гострих країв і виявлення різних контурів на зображенні. Потім зображення перетворюється в одномірну лінійну матрицю.

Останній шар – повністю з'єднаний, який використовується для ідентифікації зображень та класифікації зображень відповідно до досягнутої точності (довірчої ймовірності).

Переваги згорткового типу нейронних мереж:

– Розпізнавання в просторовому контексті. Згорткові нейронні мережі здатні виявляти локальні залежності у вхідних даних, враховуючи їх просторову структуру. Це дозволяє їм ефективно розпізнавати об'єкти та ознаки в зображеннях, аудіо- або відеоданих;

– Параметрична ефективність. Згорткові шари мають спільні ваги, що дозволяє значно зменшити кількість параметрів мережі. Це зроблено для збільшення ефективності навчання та зменшення обчислювальних витрат;

– Інваріантність до трансляції. Згорткові шари дозволяють отримувати інваріантність до трансляції об'єктів в зображеннях. Це означає, що нейромережа здатна розпізнавати об'єкти незалежно від їх положення на зображенні.

Недоліки згорткового типу нейронних мереж:

– обмежена розмірність вхідних даних. згорткові нейронні мережі оптимально працюють з вхідними даними фіксованого розміру. Розмірність вхідних зображень або даних потрібно налаштовувати під вимоги мережі, що може бути проблематичним у деяких сценаріях;

– втрата локальної інформації. згорткові шари мережі здатні виявляти локальні ознаки, але при цьому може втрачатися загальна контекстуальна інформація. Деякі взаємозалежності між різними частинами даних можуть бути недостатньо враховані.

– обчислювальні витрати. згорткові нейронні мережі можуть вимагати значних обчислювальних ресурсів для тренування та застосування, особливо при великих обсягах даних або складних моделях.

1.1.2 Сімейство нейромереж EfficientNet

EfficientNet використовує попередньо навчені згорткові нейронні мережі для виконання функцій, пов'язаних із зображеннями, як базової мережі. Ці базові мережі здатні навчатися на широкому спектрі даних, що дозволяє швидше створювати більш конкретні моделі з обмеженими навчальними даними можуть бути створені швидше. Такі мережі корисні для таких функцій,

як класифікація зображень і розпізнавання, що дає переваги у ситуацій, а також використання більш точних та ефективних моделей.

– Незважаючи на те, що звичайний процес масштабування довільно визначений, він все ж дає функціональні результати. EfficientNet спочатку проводить сітковий пошук базової мережі для визначення взаємозв'язків між різними масштабуючими розмірами мережі, враховуючи як розмір моделі, так і доступні обчислювальні ресурси.

– Обчислювальні ресурси. У більшості ситуацій результати початкового тестування показують вищу точності та швидкості [8].

Таблиця 1.1 Короткий опис архітектури EfficientNet-B0

Стадія	Оператор	Формат	Канали	Шари
1	Conv3x3	224x224	32	1
2	MBConv1, k3x3	112x112	16	1
3	MBConv6, k3x3	112x112	24	2
4	MBConv6, k3x3	56x56	40	2
5	MBConv6, k3x3	28x28	80	3
6	MBConv6, k3x3	14x14	112	3
7	MBConv6, k3x3	14x14	192	4
8	MBConv6, k3x3	7x7	320	1
9	Conv1x1 & Pooling & FC	7x7	1280	1

Переваги типу нейронних мереж EfficientNet:

– висока ефективність. EfficientNet має велику ефективність з точки зору обчислювальних витрат і розміру моделі. Вона досягається за рахунок оптимального масштабування моделі за глибиною, шириною та роздільною здатністю;

– краща точність. EfficientNet показує високу точність при класифікації зображень і вирішенні задач комп'ютерного зору. Вона досягається завдяки комплексному масштабуванню, що дозволяє зберегти якість моделі при зменшенні її розміру;

– застосовність на різних пристроях. EfficientNet підходить для використання на різних пристроях, включаючи мобільні телефони, планшети і вбудовані системи. Вона забезпечує баланс між точністю і обчислювальною складністю, що робить її ідеальним вибором для обмежених ресурсів.

Недоліки типу нейронних мереж EfficientNet:

– менша гнучкість. Оскільки EfficientNet оптимізована під конкретний масштаб моделі, вона може бути менш гнучкою в порівнянні з іншими типами мереж. Вона може не бути найкращим варіантом для задач, де потрібне детальне налаштування моделі;

– висока вимогливість до даних. EfficientNet може вимагати значну кількість даних для ефективного навчання. Якщо доступні обмежені об'єми даних, може бути важко досягти хорошої точності моделі;

– можливий втратний стиснення. Зменшення розміру моделі EfficientNet може призводити до певних втрат в точності. При підборі оптимального розміру моделі слід ретельно збалансувати між точністю і розміром.

1.1.3 Комбіновані методи

Комбіновані методи, які використовують комбінацію різних типів нейронних мереж, стають все більш популярними у сфері розпізнавання об’єктів. Ці методи поєднують в собі переваги різних типів мереж та дозволяють отримати більш точні й надійні результати [9].

Один з таких підходів – комбінація згорткових нейронних мереж (Convolutional Neural Networks, CNN) та рекурентних нейронних мереж (Recurrent Neural Networks, RNN). CNN використовуються для виявлення та витягування характеристик об'єктів на зображенні, таких як форма, текстура або кольори об’єктів. RNN, у свою чергу, допомагають врахувати контекстуальну інформацію та зв'язки між об'єктами на зображенні, що покращує розпізнавання та класифікацію об’єктів.

Ще один підхід – комбінація CNN та глибоких згорткових нейронних мереж (Deep Convolutional Neural Networks, DCNN). DCNN є більш потужними та складнішими моделями, які дозволяють автоматично витягувати високорівневі ознаки зображень. Поєднання CNN та DCNN дозволяє отримати більш глибокі та абстрактні представлення об’єктів [13], що поліпшує розпізнавання та дискримінацію між різними видами об’єктів.

Крім того, можна комбінувати нейронні мережі з іншими методами, такими як методи геометричної моделі або методи машинного навчання. Наприклад, можна використовувати комбінацію CNN з методами геометричної моделі для точного визначення положення та форми об’єктів у просторі. Це дозволяє отримати більш повну та точну інформацію про об’єкти.

Застосування сполучених методів дозволяє покращити якість та точність розпізнавання об’єктів, забезпечуючи більш повну та контекстуальну інформацію про об'єкти. Ці методи стають все більш популярними в галузі розпізнавання об’єктів і відкривають нові перспективи для автоматизації та оптимізації процесів у сільському господарстві .

Переваги сполучених методів, що поєднують різні типи нейронних мереж для розпізнавання об’єктів [13]:

– комбінація різних типів мереж дозволяє отримати більш точні результати розпізнавання об’єктів, завдяки використанню різних характеристик та контекстуальної інформації;

– краща здатність до узагальнення: Сполучені методи дозволяють отримати більш глибокі та абстрактні представлення об’єктів, що покращує здатність до узагальнення на нові та невидані раніше зразки;

– більш повна інформація: Комбінація різних методів нейронних мереж дозволяє отримати більш повну інформацію про об’єкти, включаючи їх форму, текстуру, кольори та контекстуальні зв'язки.

Недоліки сполучених методів:

– висока обчислювальна складність: Комбінація різних типів мереж може призводити до значного збільшення обчислювального завантаження та вимог до обладнання;

– потреба у великій кількості даних: Для ефективної роботи сполучених методів необхідна достатня кількість репрезентативних даних для тренування, що може бути складним завданням у випадку обмеженої доступності даних;

– складність налаштування та оптимізації: Використання сполучених методів вимагає дослідження та оптимізації різних параметрів та архітектур моделей, що може бути частота ресурсозатратним процесом.

Незважаючи на недоліки, сполучені методи знаходять широке застосування у сфері розпізнавання об’єктів.

1.2 Аналіз існуючих додатків по роботі із плодами

Існує перелік наступних програм по роботі із розпізнаванням фруктів:

– Fruit Recognition API by Clarifai. API використовує глибокі нейронні мережі для розпізнавання фруктів на зображеннях. Він надає високу точність розпізнавання та широкий набір підтримуваних фруктів. Програма також здатна виявляти інші ознаки, наприклад, колір і форму фрукту;

– Plantix. Програма розроблена спеціально для сільськогосподарських потреб і включає функцію розпізнавання фруктів. Вона надає користувачам можливість виявляти хвороби, шкідників та недоліки на фруктах. Також вона має розширений функціонал, який дозволяє аналізувати зображення листя, стебла та інших частин рослини;

– Fruit Recognition by ImageAI. Програма використовує згорткову нейромережу для розпізнавання фруктів на зображеннях. Вона може розпізнавати широкий спектр фруктів та використовує передобробку зображень для поліпшення точності розпізнавання;

– Fruit Detector and Classifier by DeepQuest AI. Програма також використовує згорткову нейромережу для розпізнавання фруктів. Вона може

виявляти фрукти на зображеннях та класифікувати їх за видами. Програма має високу швидкодію та добру точність розпізнавання.

Порівняння існуючих додатків по роспізнаванню фруктів наведено в табл. 1.2.

Таблиця 1.2 Порівняння існуючих додатків по роспізнаванню фруктів

Програма	Точність розпізнавання	Наявність додаткових функцій
Fruit Recognition API by Clarifai	Висока (>0,9)	Багато (>10)
Plantix	Висока (>0,9)	Достатньо (3-5)
Fruit Recognition by ImageAI	Висока (>0,9)	Багато (>10)
Fruit Detector and Classifier by DeepQuest AI	Достатня (0,7-0,9)	Достатньо (3-5)

1.3 Формування вимог до змісту програмного модуля і завдання на кваліфікаційну роботу

Результатом проекту є програмний модуль, який здатний автоматично розпізнавати та класифікувати об'єкти у промисловому фруктовому саду на основі зображень. Основні завдання включають виявлення фруктів, оцінку їх стану та кількості.

Використання систем комп'ютерного зору з нейронними мережами в сільськогосподарських агрегатах має великий потенціал для автоматизації обліку та контролю прийняття рішень.

Було проаналізовано низки наукових статей, які висвітлюють питання розпізнавання зображень плодів в промисловому садівництву.

В статті [10] було наведено приклад розробки системи для ідентифікації фруктів на кронах дерев, де в результаті полевого експерименту виявлено, що помилки розробленого програмно-апаратного комплексу у оцінці розміру фруктів переважно викликані неточною сегментацією зображень та низьким розширенням використовуваної камери.

В статті [11] було розгялнуто розробку інтелектуальної системи для оцінки просторового положення яблук на основі алгоритму YOLOv3 та

глибинної камери Real Sense D415 і в результаті доведено, що для роботизованого збирання яблук необхідно чітко зрозуміти, куди і наскільки перемістити захват.

Стаття [12] показала можливості виявлення аномалій у візуальних даних за допомогою засобів комп'ютерного зору. Також описано методи побудови композитної, налаштованої моделі комп'ютерного зору для аналізу зображень листя рослин та пошуку дефектів на них.

З урахуванням аналізу наявних наукових досліджень для досягнення поставленої мети було вирішено використовувати згорткову нейромережу. Спеціалізованість згорткових нейромереж для обробки зображень дозволяє їм ефективно працювати з великими наборами даних фотографій фруктів [13, 14]. Вони можуть виявляти різні ознаки і шаблони у зображеннях, що полегшує розпізнавання об'єктів.

Слід зауважити, що згорткові нейромережі є інваріантними до зміщення та масштабу об'єктів на зображенні, що означає, що вони можуть розпізнавати об'єкти незалежно від їх положення та розміру [15, 16, 17]. Також навчання згорткових нейромереж дозволяє досягти високої точності розпізнавання фруктів. Вони можуть виявляти складні шаблони та залежності між пікселями, що покращує якість розпізнавання. Обраний тип гнучкий і може бути налаштовано для конкретного завдання розпізнавання фруктів. Може бути розширено або модифіковано для виявлення стану і кількості фруктів, а також для виконання додаткових завдань, наприклад, виявлення хвороб або оцінки зрілості фруктів.

Таким чином, задача полягає у розробці програмного модулю з використанням згорткової нейромережі, який зможе точно та автоматично розпізнавати об'єкти в промислових фруктових садах, сприяючи покращенню управління та діагностиці стану рослин.

Системні вимоги:

– модуль повинен бути сумісний з операційною системою, такою як Windows;

– для кращої роботи модуля, бажано мати технічну базу на апаратурі запуску, таку як графічний чип GPU для обробки зображень або високопродуктивні процесори для швидкого виконання обчислень.

Функціональні вимоги:

– розпізнавання об'єктів: модуль повинен бути здатний розпізнавати фрукти на зображеннях, використовуючи згорткову нейромережу;

– визначення стану фруктів: модуль може мати функціонал для визначення стану фруктів, такий як зрілість або пошкодження;

– визначення кількості фруктів: модуль може мати можливість визначати кількість фруктів на зображенні;

Нефункціональні вимоги:

– точність: модуль повинен мати високу точність розпізнавання фруктів та виявлення їх стану [13];

– швидкодія: модуль повинен працювати швидко і ефективно, забезпечуючи оперативну обробку зображень та виконання розпізнавання;

– масштабованість: модуль повинен бути здатний працювати з великими наборами даних та обробляти зображення в режимі реального часу;

– надійність: модуль повинен бути стабільним і надійним, забезпечуючи точні результати незалежно від умов використання;

– зручність використання: модуль повинен мати зручний інтерфейс користувача, що дозволяє легко завантажувати зображення, отримувати результати та налаштовувати параметри моделі.

Вимогою окремого виду забезпечення є встановлення деяких бібліотек на апарат, що буде виконувати функцію розпізнавання через модуль.

Вхідною інформацією для модулю є зображення плодів. Кожне зображення містить один або декілька фруктів, які потрібно розпізнати. Зображення можуть мати різну роздільну здатність і формат, наприклад, JPG або PNG. [14] Для кращих результатів розпізнавання можуть бути застосовані попередні обробки, такі як зміна розміру, кадрування, нормалізація кольору або зменшення шуму.

Крім зображень, база даних повинна включати також додаткову інформацію про фрукти, як наприклад, мітки або класи, які вказують на вид фрукту або його стан (наприклад, зрілість або пошкодження). Ці мітки використовуються для тренування та оцінки моделі розпізнавання.

Важливо, щоб вхідна база даних була репрезентативною і містила різноманітні зображення фруктів з різних ракурсів, розмірів, освітлення та умов вирощування [15]. Це допомагає навчити нейромережу розпізнавати фрукти з високою точністю незалежно від умов застосування модуля.

Задачі та вимоги до кваліфікаційної роботи можна сформувати наступним чином:

1. Дослідити та проаналізувати існуючі методии та алгоритмім розпізнавання об'єктів на зображеннях, зокрема з застосуванням згорткових нейромереж.

2. Розробити програмний модуль для розпізнавання фруктів, ступеня їх стиглості або спорченості за допомогою згорткової нейромережі.

3. Вибрати та навчити відповідну архітектуру згорткової нейромережі для розпізнавання об’єктів у промисловому фруктовому саду.

4. Зібрати та підготовати датасет зображень фруктів для тренування та тестування моделі.

5. Провести навчання згорткової нейромережі на підготовленому датасеті.

6. Провести валідацію та тестування розробленого програмного модуля з використанням реальних зображень фруктів.

7. Провести оцінку точності та продуктивності розробленого модуля на основі метрик розпізнавання об'єктів.

8. Провести аналіз результатів та порівняти їх з існуючими методами розпізнавання об'єктів.

1.4 Висновки до розділу 1

Обрано тему розроблення модулю розпізнавання фруктів, що являтиме собою підхід до вирішення завдань визначення стиглості та кількості фруктів у промислових фруктових садах. Аналіз загальної задачі розпізнавання фруктів підтверджує актуальність даної розробки, оскільки існуючі програми надають обмежені можливості і не враховують всіх необхідних факторів, таких як виявлення стану фруктів та інші додаткові функції.

При розробці модуля обрано оптимальні сучасні методи згорткових нейронних мереж, що дозволяють отримати високу точність розпізнавання та узагальнення на різні види фруктів.

Також сформовано функціональні та нефункціональні вимоги до програмного модуля та поставлено завдання, які необхідно виконати в межах кваліфікаційної роботи.

РОЗДІЛ 2
РОЗРОБКА АРХІТЕКТУРИ СИСТЕМИ РОЗПІЗНАВАННЯ ПЛОДІВ

2.1 Аналіз функцій системи

Перед розглядом аналізу функцій системи розпізнавання типу плодів, необхідно чітко визначити, які функції повинна виконувати така система [16, 17, 18]. На основі функціонального аналізу можна скласти дерево функцій (рис.2.1) та карти процесів, щоб краще зрозуміти, як буде працювати програма з класифікації плодів.

Рисунок 2.1 – Дерево функцій програмного модуля розпізнавання об’єктів

Основні функції системи розпізнавання типу плодів включають:

– Зчитування та обробка фотографій плодів: Система повинна мати здатність отримувати фотографії плодів, а потім проводити попередню обробку для видалення шуму та підготовки зображення для подальшого аналізу [19].

– виділення ознак: Система повинна визначати ключові ознаки або характеристики на фотографії, які можуть відрізняти різні типи плодів. Це можуть бути колір, форма, текстура, розмір та інші атрибути [20].

– класифікація плодів: На основі виділених ознак система повинна здати змогу класифікувати плоди на визначені типи [21]. Для цього можуть використовуватися методи машинного навчання, нейронні мережі або інші алгоритми класифікації.

– оцінка надійності класифікації: Після класифікації система повинна забезпечити оцінку надійності розпізнавання. Це може включати побудову довірчого інтервалу для визначення точності класифікації та інші метрики ефективності.

Розглянемо DFD (Data Flow Diagram – графічний метод моделювання потоку даних та процесів у системі) для візуалізації та аналізу потоків даних між різними компонентами системи (рис.2.2).

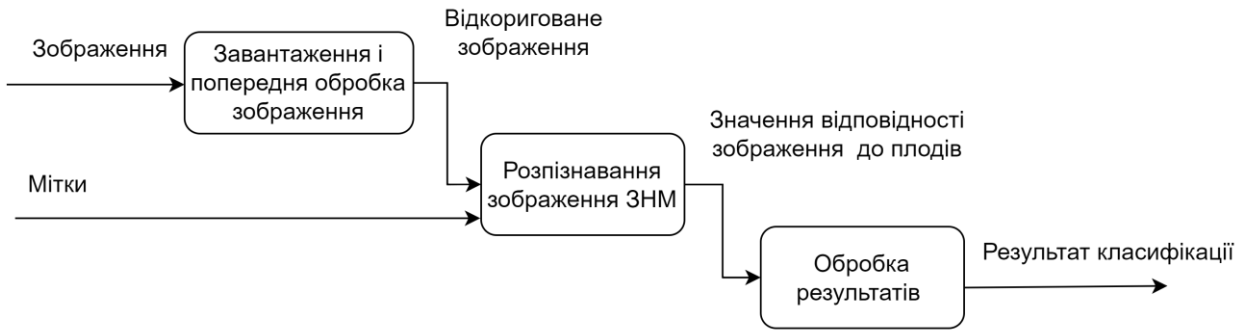


Рисунок 2.2 – Контекстна діаграма IDEF0 програмного модуля розпізнавання плодів

Представимо детальний опис DFD для програмного модуля розпізнавання фруктів:

– вхідні дані:

1. Зображення –вхідне зображення, яке потрібно розпізнати на наявність фруктів.

2. Мітки – інформація про правильні класифікації фруктів на зображенні. Використовується для навчання та оцінки точності системи.

– процеси:

1. Завантаження і попередня обробка зображення – забезпечується завантаження зображення, його попередня обробка для підготовки до розпізнавання. Може включати зміну розміру зображення, нормалізацію, фільтрацію шуму, тощо.

2. Розпізнавання зображення згортковою НМ – використовуэться згорткова нейронна мережа для розпізнавання фруктів на зображенні. Вона аналізує зображення та визначає класи фруктів, їх місцезнаходження та інші характеристики.

3.Обробка результатів – проводиться обробка результатыв розпізнавання, включаючи відповідність зображення до фруктів та класифікацію фруктів. Він може включати порігову фільтрацію, видалення неправильних результатів, обчислення метрик якості, тощо.

– вихідні дані кожного процесу:

1.Відкориговане зображення – зображення після попередньої обробки, готове для розпізнавання фруктів. Передається в процес розпізнавання зображення Згортковою НМ.

2.Значення відповідності зображення до плодів – значення відповідності, що зображення містить визначені фрукти (генерується процесом розпізнавання).

3. Результат класифікації – охоплює класифікацію фруктів на зображенні. Цей вихід містить інформацію про класи фруктів, їхню кількість та додаткові характеристики, такі як ступінь стиглості та якість фруктів. Ця інформація може бути використана для подальшої обробки або відображена для користувача.

Архітектура інформаційної системи (ІС) формується на основі необхідних функцій, обраної технологічної платформи та доступних ресурсів (рис. 2.3).

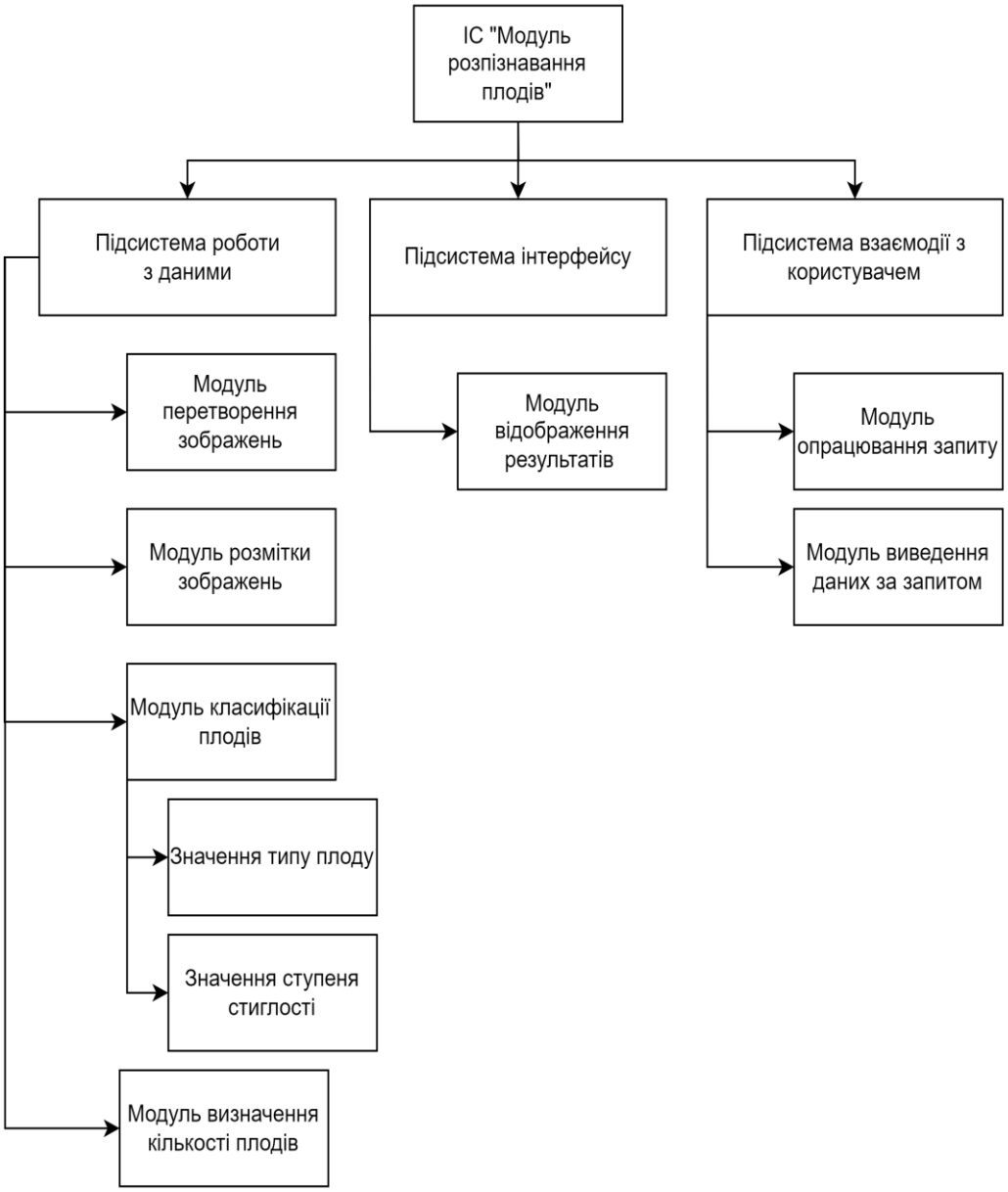


Рисунок 2.3 – Архітектура програмного модуля

Розглянемо розроблену архітектуру:

1. Фронтенд компонент: включає інтерфейс користувача та функціональність для обробки даних, які система буде отримувати від користувачів.

2. Бекенд компонент: додаток, який виконує функцію серверної частини системи і забезпечує роботу моделі для розпізнавання плодів.

3. Модуль збору даних: здійснює збір даних від користувачів та передачу їх на бекенд компонент для подальшого аналізу [23].

4. Модуль розпізнавання плодів: використовує модель YOLO для аналізу даних та класифікації плодів.

5. Модуль візуалізації результатів: забезпечує відображення результатів на інтерфейсі користувача.

Підсистема інтерфейсу містить модуль відображення результатів, який відповідає за візуалізацію результатів розпізнавання плодів, зокрема відображення зображень з позначеними класами та стиглістю плодів (табл. 2.1).

Таблиця 2.1 Перелік необхыдних модулів для розпізнавання об’єктів

Підсистема роботи з даними	Підсистема інтерфейсу	Підсистема взаємодії з користувачем
– Модуль перетворення зображень – модифікує зображення до стандартного вигляду. – Модуль розмітки зображень – задає маркування до кожного класу. – Модуль класифікації плодів – використовує модель YOLO та повертає значення типу плоду та ступінь стиглості плоду. – Модуль визначення кількості плодів – аналізує зображення і видає кількість плодів на ньому.	– Модуль відображення результатів	– Модуль опрацювання запиту. – Модуль виведення даних за запитом (користувачу необхідно побачити результат його запиту)

Підсистема роботи з даними складається з кількох модулів, які отримано за рахунок модифікації моделі YOLO, в якій всі модулі вже вбудовано і тільки вимагається їх переналаштування (але за рахунок відкритого коду системи можна вносити модифікації):

– модуль перетворення зображень, відповідає за модифікацію зображень до стандартного формату, наприклад, зміна їх розмірів або нормалізацію яскравості;

– модуль розмітки зображень, служить для задання маркування або анотацій до різних класів об'єктів або плодів на зображеннях;

– модуль класифікації плодів для визначення типу плоду та ступеня його стиглості;

– модуль визначення кількості плодів, який аналізує зображення і надає інформацію про кількість плодів, які присутні на ньому [11].

У підсистемі взаємодії з користувачем є модуль опрацювання запиту, який приймає та обробляє запити користувача, і модуль виведення даних за запитом, який передає користувачеві інформацію про розпізнані плоди, їх класифікацію, стиглість та кількість на зображенні.

2.2 Інформаційне забезпечення системи

Для розробки даного додатку потрібно створити базу даних, яка відображатиме всі сутності та їх взаємозв'язки.

На основі аналізу предметної області розпізнавання плодів на зображеннях [1, 11, 24, 25, 26, 27] можна сформувати наступне представлення структури, взаємозв'язків та характеристик, які можуть бути включені в концептуальну модель:

1. Сутність "Плід":

– Ідентифікатор плоду (fruit_id): унікальний ідентифікатор плоду.

– Назва плоду (fruit_name): назва або опис плоду (наприклад, яблуко, банан, апельсин).

– Характеристики плоду (fruit_features): додаткова інформація про плід, які можуть включати колір, форму, текстуру тощо.

2. Сутність "Зображення плоду":

– Ідентифікатор зображення (image_id): унікальний ідентифікатор зображення плоду.

– Ідентифікатор плоду (fruit_id): зовнішній ключ, пов'язаний з ідентифікатором плоду в таблиці "Плід".

– Шлях до зображення (image_path): шлях до файлу зображення плоду.

– Дата зображення (image_date): дата, коли було зроблено зображення.

3. Сутність "Модель розпізнавання":

– Ідентифікатор моделі (model_id): унікальний ідентифікатор моделі розпізнавання.

– Назва моделі (model_name): назва або опис моделі розпізнавання.

– Шлях до файлу моделі (model_path): шлях до файлу, який містить параметри моделі.

4. Сутність "Результати розпізнавання":

– Ідентифікатор результату (result_id): унікальний ідентифікатор результату розпізнавання.

– Ідентифікатор зображення (image_id): зовнішній ключ, пов'язаний з ідентифікатором зображення в таблиці "Зображення плоду".

– Ідентифікатор моделі (model_id): зовнішній ключ, пов'язаний з ідентифікатором моделі в таблиці "Модель розпізнавання".

– Мітка розпізнавання (classification_label): класифікаційна мітка, яка визначає вид плоду.

– Ступінь впевненості розпізнавання (classification_confidence): ймовірність, що плід був розпізнаний правильно.

На основі даного опису можна представити концептуальну модель бази даних програмного модуля (рис. 2.4).

Логічна структура моделі даних визначає зв'язки і взаємозв'язки між таблицями та їх структуру без прив'язки до конкретної бази даних або мови запитів. Основна мета логічної моделі даних - описати взаємозв'язки між сутностями та їх атрибутами.

В нашому випадку, логічна структура моделі даних має такий вигляд (рис. 2.5):

– сутність "Фруктові зображення" з атрибутами: ідентифікатор зображення, шлях до зображення, дата зображення тощо;

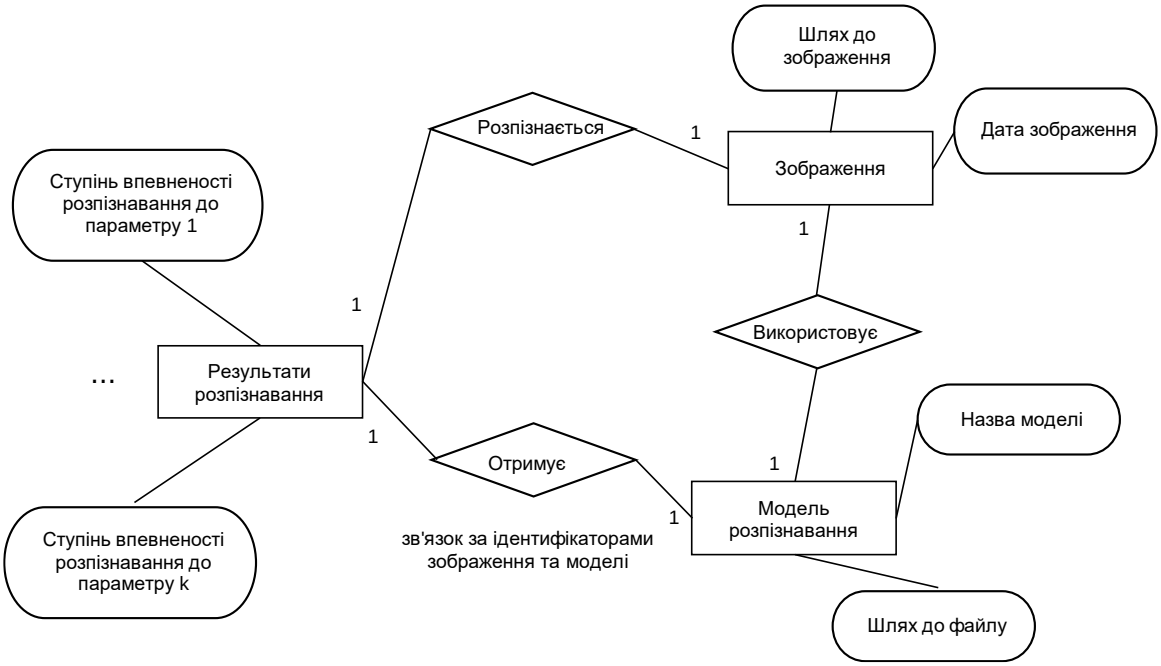


Рисунок 2.4 – Концептуальна модель БД

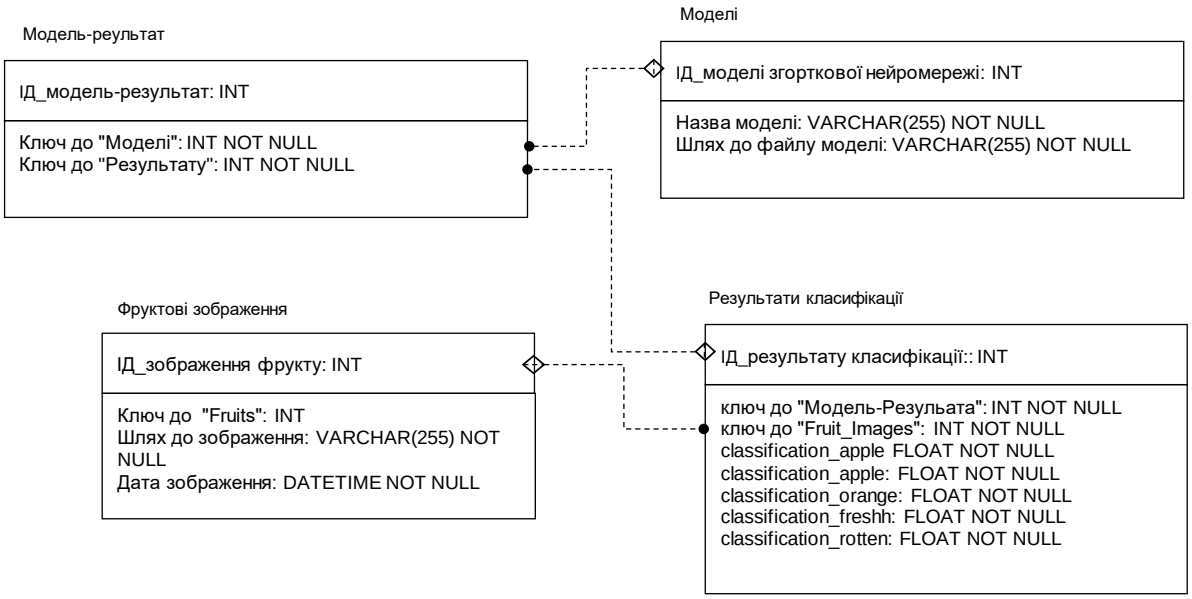


Рисунок 2.5 – Логічна структура моделі даних

– сутність "Моделі" з атрибутами: ідентифікатор моделі, назва моделі, шлях до файлу моделі тощо.

– сутність "Результати класифікації" з атрибутами: ідентифікатор результату, ідентифікатор зображення, ідентифікатор модель-результат, поля ступенів відношення до об’єктів з бази (для бази на вузькопрофільному

розпізнаванні для промислових фруктовиз садів нема сенсу створювати окрему сутність Плоди або Фрукти, тому що це тільки ускладнить структуру БД).

Фізична модель бази даних визначає спосіб організації та збереження даних на фізичному носії, такому як жорсткий диск або хмарна інфраструктура. Основним елементом фізичної моделі є таблиці, колонки та взаємозв'язки між ними (рис. 2.6).

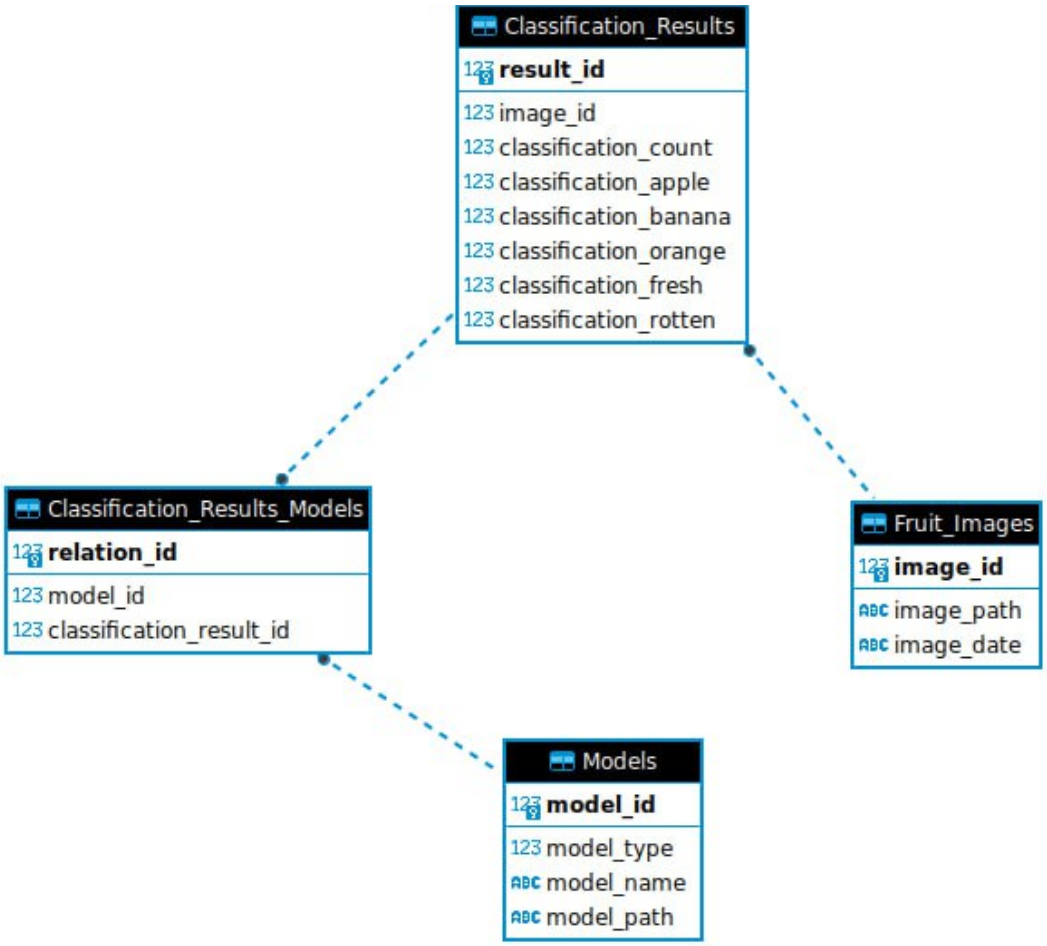


Рисунок 2.6 – Структура фізичної моделі даних

Результати навчання не зберігаються безпосередньо в базі даних, оскільки це великі обсяги даних і зберігання їх у базі може бути неефективним. Замість цього, результати навчання (навчені ваги моделей, метрики, графіки тощо) зберігаються у вигляді файлів на файловій системі.

Базу даних розроблено під SQLite, але можна обрати будь-яку СУБД, тому що вона не несе критичні навантаження під час навчання системи чи розпізнавання зображень.

2.3 Проєктування інтерфейсу програмного модуля

Розробка інтерфейсу програмного модуля включає наступні кроки:

1. Визначення вимог: перед початком розробки необхідно чітко визначити вимоги до інтерфейсу. Це включає визначення функціональних вимог (що повинен робити інтерфейс) і нефункціональних вимог (наприклад, естетика, доступність, продуктивність).

2. Проектування інтерфейсу: на основі вимог необхідно розробити дизайн інтерфейсу. Це включає визначення структури, компонентів, елементів керування, розміщення елементів і вигляду інтерфейсу. Важливо враховувати зручність використання, простоту навігації і зворотну зв'язок з користувачем.

3. Розробка прототипу: на цьому етапі розробляють прототип інтерфейсу, щоб перевірити його функціональність та взаємодію з користувачем. Прототип може бути інтерактивним, що дозволяє тестувати різні сценарії взаємодії та вносити коригування за потреби.

4. Реалізація інтерфейсу: після схвалення прототипу необхідно реалізувати інтерфейс. Це може включати розробку фронтенду (HTML, CSS, JavaScript) і зв'язок інтерфейсу з бекендом (наприклад, за допомогою API).

5. Тестування: після реалізації інтерфейсу важливо провести тестування для перевірки його працездатності, взаємодії з користувачем і відповідності вимогам. Тестування може включати функціональне тестування, тестування взаємодії, тестування відповідності дизайну тощо.

6. Впровадження і підтримка: після успішного завершення тестування інтерфейс готовий для впровадження. Підтримка включає в себе реагування на відгуки користувачів, вирішення проблем та вдосконалення інтерфейсу з часом.

За результатом аналізу функціональних вимог було побудовано схему відображення необхідних компонентів на вікні (рис. 2.7).

Детальний опис структури вікна:

– назва вікна: розміщена по центру вгорі вікна, містить заголовок, який визначає контекст або назву модуля;

Рисунок 2.7 – Розміщення інформаційних компонентів на вікні програмного модуля

– ліва колонка: результат розпізнавання до плодів, яка відображає шкали відповідності знайдених об'єктів на малюнку. Використовується для оцінки, наскільки точно знайдені об'єкти (яблука, банани, апельсини) відповідають реальним об'єктам;

– права колонка: вихідний малюнок з розміткою, який відображає маркування, де кожен знайдений об'єкт позначений та має підпис, під малюнком вказується кількість знайдених об'єктів;

– внизу вінкна шкали стиглості, що відображають шкали відповідності стиглості і гнилості знайдених фруктів на малюнку (передбачається середня оцінка за всіма об’єктами).

Розробка прототипу з шкалами має наступний вигляд (рис. 2.8).

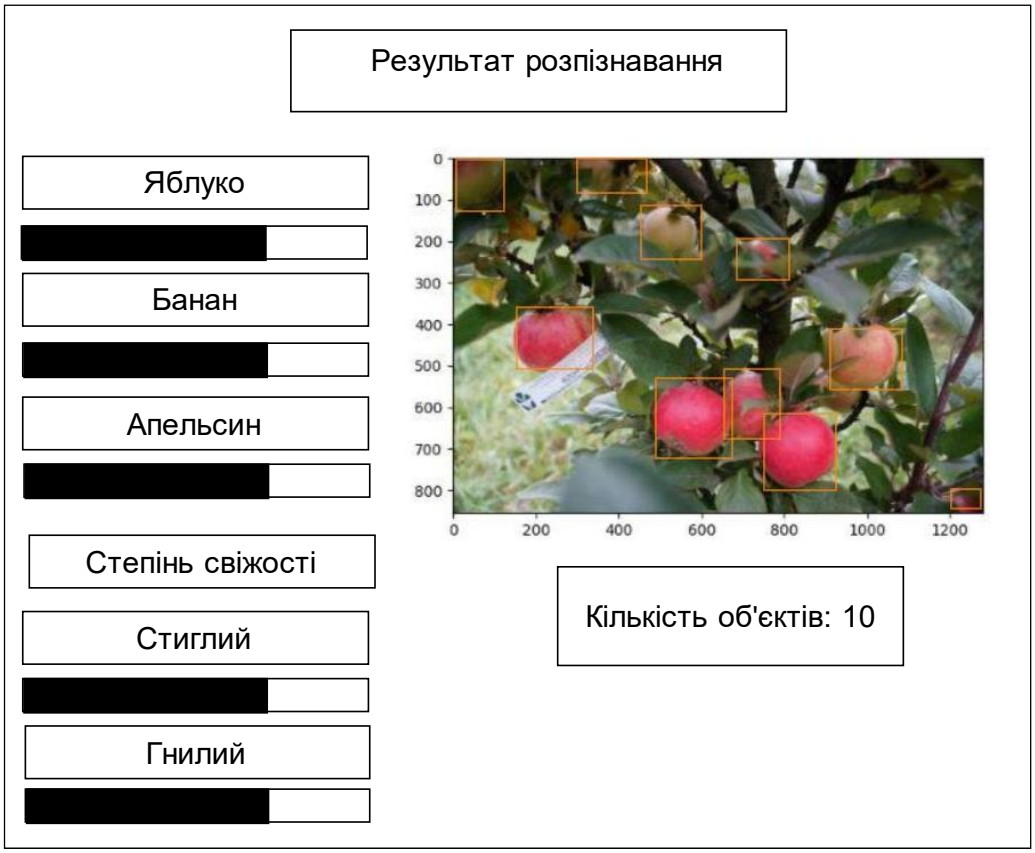


Рисунок 2.8 – Прототип інтерфейсу головного вікна програмного модуля

2.4 Висновки до розділу 2

У розділі було проведено розробку архітектури системи розпізнавання плодів. Під час функціонального аналізу були визначені головна мета функціонування системи і побудована карта процесів роботи застосунку. Для детальнішого опису функціональності була створена діаграма процесу ЯК-БУДЕ, де виділені фрагменти, які раніше не розглядались.

Архітектура інформаційної системи розпізнавання плодів має просту структуру, засновану на базі даних, що включає таблиці з інформацією про плоди. За допомогою концептуальної та логічної моделей, розроблених у цьому розділі, можна визначити структуру бази даних. Для системи необхідні таблиці, що містять дані про плоди, такі як назви, опис, властивості тощо.

Також система повинна мати таблицю для зберігання результатів розпізнавання плодів, де будуть міститися дані про плоди, які були аналізовані,

та відповідні результати. Ця таблиця дозволить зберігати і використовувати інформацію про розпізнані плоди для подальшого аналізу та відображення результатів.

Описані таблиці можуть бути використані для реалізації різних функцій системи, таких як пошук плодів за їх характеристиками, фільтрація за певними параметрами, створення звітів про розпізнавання плодів тощо. Це дозволить користувачам системи здійснювати ефективний пошук та аналіз розпізнаних плодів з урахуванням їх властивостей.

Також було проведено проєктування інтерфейсу програмного модуля, яке визначило розташування основних інформаційних компонентів на вікні результату.

РОЗДІЛ 3
РЕАЛІЗАЦІЯ МОДУЛЯ ПО РОЗПІЗНАВАННЮ ОБ'ЄКТІВ У ПРОМИСЛОВОМУ ФРУКТОВОМУ САДУ

3.1 Обрані методи реалізації

Загальний вигляд було розроблено як веб-додаток, що має свої причини:

– веб-додаток забезпечує зручний інтуїтивний інтерфейс, який може бути легко використовуваний навіть некваліфікованими користувачами. Користувачі зможуть взаємодіяти з модулем за допомогою браузера з будь-якого пристрою, що робить його більш доступним;

– завдяки веб-додатку користувачі зможуть отримати доступ до модулю з будь-якого місця з підключенням до Інтернету. Це особливо корисно для використання в системах, де необхідно віддалено контролювати та візуалізувати розпізнавання фруктів;

– веб-додатки мають перевагу в тому, що їх можна легко масштабувати для обробки більшої кількості користувачів або оптимізувати для роботи з великим обсягом даних. Вони можуть бути розгорнуті на серверах або в хмарних сервісах, що дозволяє ефективно використовувати ресурси;

– веб-додатки добре інтегруються з іншими компонентами технологічного стеку, такими як бази даних, зовнішні сервіси або інші модулі. Це дозволяє легко розширювати функціональність модулю та забезпечувати його взаємодію з іншими системами;

– завдяки веб-додатку можна зручно візуалізувати результати розпізнавання фруктів, їх кількості та стиглості у вигляді графіків, діаграм або зображень. Це дозволяє користувачам легко аналізувати дані та отримувати зрозумілу інформацію (рис. 3.1).

Першим і головним способом реалізації є мова програмування. Обрання мови програмування Python для розробки веб-додатку з модулем розпізнавання фруктів є виправданим і має кілька суттєвих переваг.

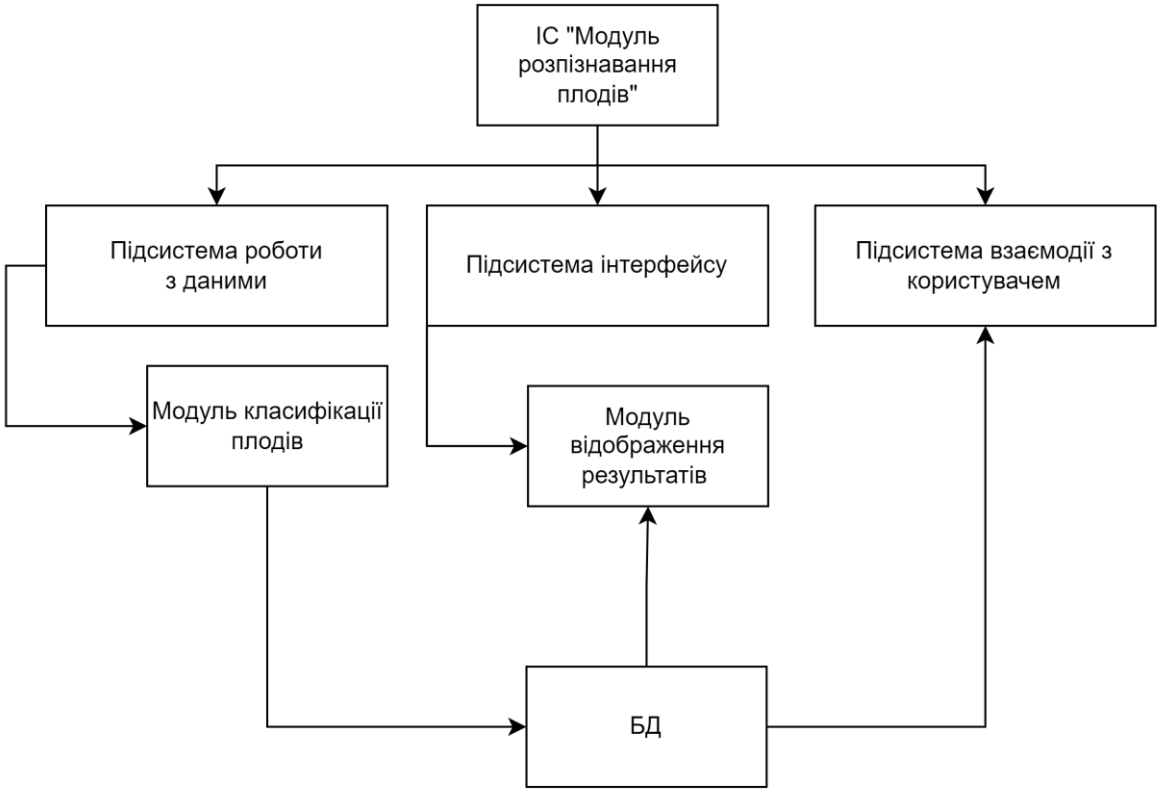


Рис. 3.1.Схема взаємодії підсистем з базою даних для формування веб-сторінки з результатами

Python пропонує потужні фреймворки для веб-розробки, такі як Django і Flask. Ці фреймворки надають гнучкість та готові рішення для швидкої розробки веб-додатків. Django забезпечує повноцінну підтримку для веб-додатків, включаючи маршрутизацію URL, керування формами, взаємодію з базами даних та автентифікацію користувачів. Flask є легковаговим фреймворком, який дозволяє розробникам самостійно обрати необхідні компоненти для свого проекту. Використання цих фреймворків значно спрощує процес розробки веб-додатку та забезпечує зручну архітектуру для модуля розпізнавання фруктів.

Також, ця мова є вже випробованою у сфері штучного інтелекту. Python є однією з найпопулярніших мов програмування для розробки рішень у галузі машинного навчання та глибокого навчання. Бібліотека TensorFlow, яка надає широкий спектр інструментів для створення та тренування нейронних мереж, має добре розвинену підтримку Python. Використання TensorFlow разом з

Python дозволяє легко виконувати класифікацію фруктів за допомогою згорткових нейромереж, зокрема YOLOv8, яка є потужним алгоритмом розпізнавання об'єктів.

Python має велику та активну спільноту розробників, що призводить до наявності багато корисних пакетів та модулів. В ході написання роботи, були використані бібліотеки притаманні лише цій мові. Популярність при користування в розробці передбачає можливість легко знаходження підказок для потрібного рішення та приклади коду для різних аспектів.

Мова відома своєю простотою та зрозумілістю синтаксису. Це робить код легко читабельним та зрозумілим для команди розробників, що може сприяти швидкому розумінню та підтримці коду. Це є важливим критерієм, адже любе програмне забезпечення повинно мати змогу бути вдосконаленим по потребі. Швидке розуміння коду допомагає за короткі терміни провести необхідні маніпуляції із готовим кодом, завдяки можливості швидко згадати або зрозуміти код.

Python є гнучкою мовою програмування, що дозволяє запускати додатки на різних платформах, що забезпечить для модуля розпізнавання фруктів може працювати на різних операційних системах і пристроях без необхідності значних змін.

Обравши Python для розробки веб-додатку з модулем розпізнавання фруктів, буде використано потужну та зручну мову програмування, що поєднує в собі веб-розробку, машинне навчання та взаємодію з TensorFlow, що дозволить швидко розробити функціональний веб-додаток для розпізнавання фруктів та збору необхідної інформації за допомогою робототехніки.

Для реалізації у вигляді веб-застосунка було обрано мову розмітки HTML. Обрання мови HTML для розробки веб-додатку, який взаємодіє з модулем розпізнавання фруктів у браузері, має свої переваги і логічні пояснення.

HTML (HyperText Markup Language) є стандартною мовою розмітки для створення веб-сторінок та веб-інтерфейсів. Використання HTML дозволяє

легко створювати структуру сторінок, включати текст, зображення та інші мультимедійні елементи, а також взаємодіяти з користувачем за допомогою форм та кнопок.

Мова є універсальною, яка підтримується всіма сучасними браузерами. Розроблений веб-додаток, написаний на HTML, буде працювати на будь-якому браузері без необхідності в додатковій конфігурації або зміні коду.

Інтеграція з іншими мовами та технологіями у даної мови розмітки знаходиться на високому рівні. HTML легко поєднується з іншими мовами програмування та технологіями.

HTML добре підтримується на мобільних пристроях та адаптивних веб-сайтах. Це означає, що ваш веб-додаток буде доступним не лише на комп'ютерах, а й на смартфонах та планшетах, що є важливим фактором в епоху мобільних технологій.

HTML – стандартний та універсальний інструмент для створення інтерфейсу, взаємодії з користувачем та інтеграції з іншими технологіями модуля веб-додатку. Мова на сьогоднішній день є однією із найпрогресивніших, тому розробити функціональний та доступний веб-додаток для розпізнавання фруктів у браузері буде легше.

Також було використано TensorFlow – відкриту бібліотеку для машинного навчання та глибокого навчання, розробленою компанією Google. Вона надає широкі можливості для створення та навчання штучних нейронних мереж і використання їх для розв'язання різноманітних завдань у сфері обробки зображень.

TensorFlow має велику кількість інструментів та функцій для обробки зображень, таких як завантаження, попереднє оброблення, побудова та навчання моделей глибокого навчання. Вона надає готові функції для роботи з зображеннями, такі як обрізка, зміна розміру, нормалізація, аугментація та інші.

Бібліотека використовує граф обчислень, що дозволяє ефективно виконувати операції з обробки зображень на графічних процесорах (GPU) та

інших обчислювальних пристроях. Це дозволяє прискорити обробку зображень та навчання моделей.

TensorFlow має розширену підтримку глибокого навчання, зокрема набір готових моделей та алгоритмів для розпізнавання об'єктів, класифікації зображень, сегментації та інших завдань обробки зображень. Це дозволяє швидко побудувати та навчити модель для конкретного завдання без необхідності розробки алгоритмів з нуля. Використані інші бібліотеки:

– os – бібліотека надає функціональність для взаємодії з операційною системою, таку як створення та видалення файлів, робота зі шляхами до файлів тощо;

– sys – бібліотека містить функції та змінні, пов'язані зі системою Python. В даному випадку, ймовірно, вона використовується для доступу до командного рядка та аргументів командної строки.

– numpy – це основна бібліотека для наукових обчислень у Python. Вона надає потужні структури даних, такі як масиви та матриці, і високошвидкісні операції над ними. Використовується для обробки та маніпулювання зображеннями;

– cv2 – бібліотека OpenCV (Open Source Computer Vision) надає функції для обробки зображень та комп'ютерного зору. Використовується для читання та обробки зображень фруктів у модулі розпізнавання;

– Flask – це легкий фреймворк для розробки веб-додатків у Python. Використовується для створення веб-сервера та маршрутизації запитів до відповідних функцій-обробників;

– keras – використовується для завантаження навченої моделі для класифікації фруктів на основі зображень. Keras є високорівневим інтерфейсом для побудови та навчання нейромереж у Python;

– PIL – бібліотека Python Imaging Library використовується для роботи з зображеннями, зокрема для завантаження, збереження та маніпуляції зображеннями;

– io – бібліотека надає функціональність для роботи з потоками вводу-виводу. В даному випадку використовується для створення буферу пам'яті для збереження графічного зображення;

– matplotlib – бібліотека використовується для візуалізації даних та створення графіків. В даному випадку використовується для побудови графіку оригінального зображення фрукта;

– base64 – бібліотека надає функції для кодування та декодування даних у форматі base64. Використовується для кодування зображення у форматі base64 для передачі через веб-додаток.

Бібліотека ultralytics є набором інструментів для комп'ютерного зору та обробки зображень. Клас YOLO в цій бібліотеці відноситься до реалізації алгоритму об'єктного розпізнавання YOLO (You Only Look Once). YOLO є потужним алгоритмом, який дозволяє виявляти та класифікувати об'єкти на зображеннях та відео в реальному часі. За допомогою класу YOLO з бібліотеки ultralytics можна створювати об'єкти, які можуть виконувати обробку зображень, виявлення об'єктів та інші завдання, пов'язані з розпізнаванням об'єктів на зображеннях та відео.

Використана модель: YOLOv8 (You Only Look Once v8). Вона є однією з останніх версій популярної моделі для об'єктного розпізнавання і локалізації, розробленою командою Ultralytics. Ця модель є поєднанням мережі-екстрактора ознак і детектора об'єктів.

YOLOv8 відомий своєю високою швидкістю роботи, оскільки вона пропонує розпізнавання об'єктів в режимі реального часу. Модель використовує штучні нейронні мережі з побудовою на основі групування анкорних скриньок та використанням механізму попереднього пропуску для швидкої обробки зображень.

Модель досягає високої точності розпізнавання об'єктів завдяки використанню різних механізмів, таких як структурна оцінка, мультимасштабний підхід та покращена архітектура мережі. Це дозволяє моделі розпізнавати широкий спектр об'єктів з високою точністю.

Також може працювати з різними роздільними здатностями зображень, що дозволяє моделі адаптуватися до різних вимог щодо обробки зображень. Вона також підтримує виявлення об'єктів у реальному часі навіть на великих відеопотоках. YOLOv8 здатна розпізнавати об'єкти з різних категорій, таких як люди, автомобілі, тварини тощо. Вона може локалізувати об'єкти, визначати їх клас та навіть прогнозувати орієнтацію об'єктів.

3.2 Опис програмної реалізації та взаємодії між модулями

Веб-додаток має структуру, що складається із бази даних, моделей підрахунку, визначення типу та стиглості, частини веб-інтерфейсу та головної частини, яка об’єднує в собі частину, що бачить користувач, взаємодію з ним та вивід результатів моделей. При запуску модулю відкривається браузер, який встановлений за замовчуванням і відкриваються файли у папках (рис. 3.2). В ньому користувач бачить початкову сторінку додатку (рис. 3.3).

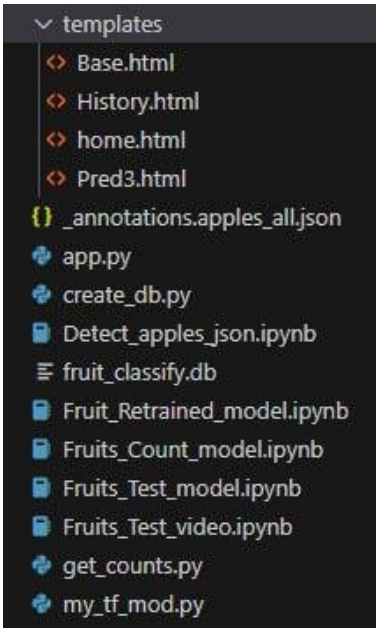


Рисунок 3.2 – Папки проєкту

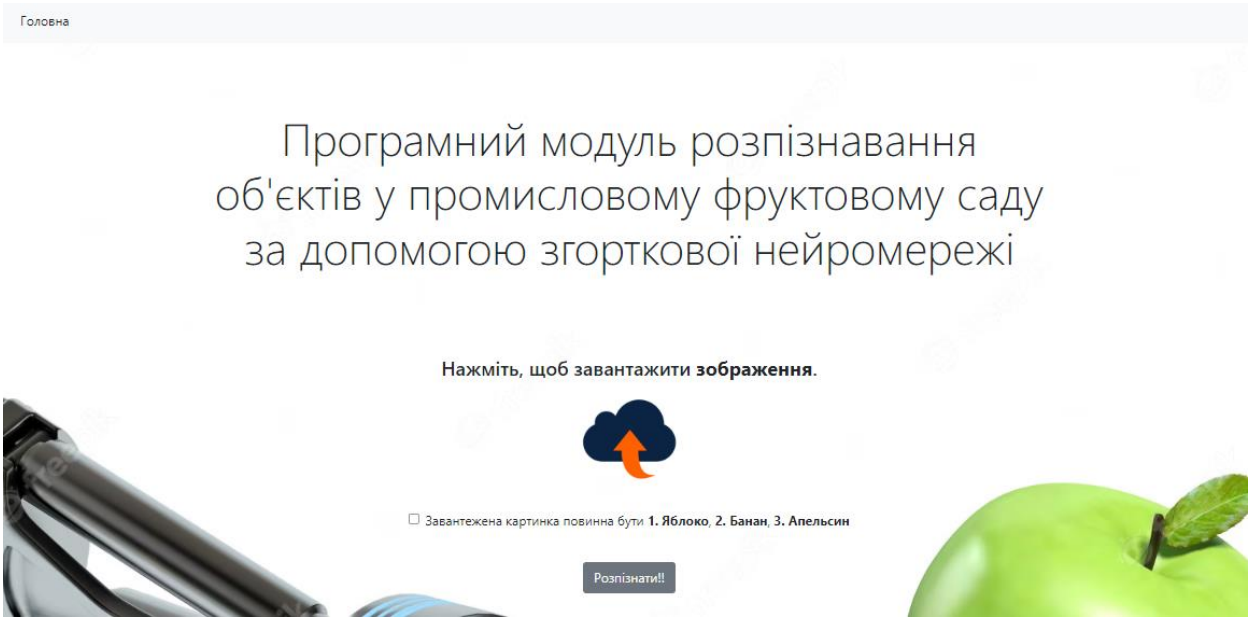


Рисунок 3.3 – Початкова сторінка модулю

Взаємодію між сторінками програмного модуля представлено на схемі (рис. 3.4).

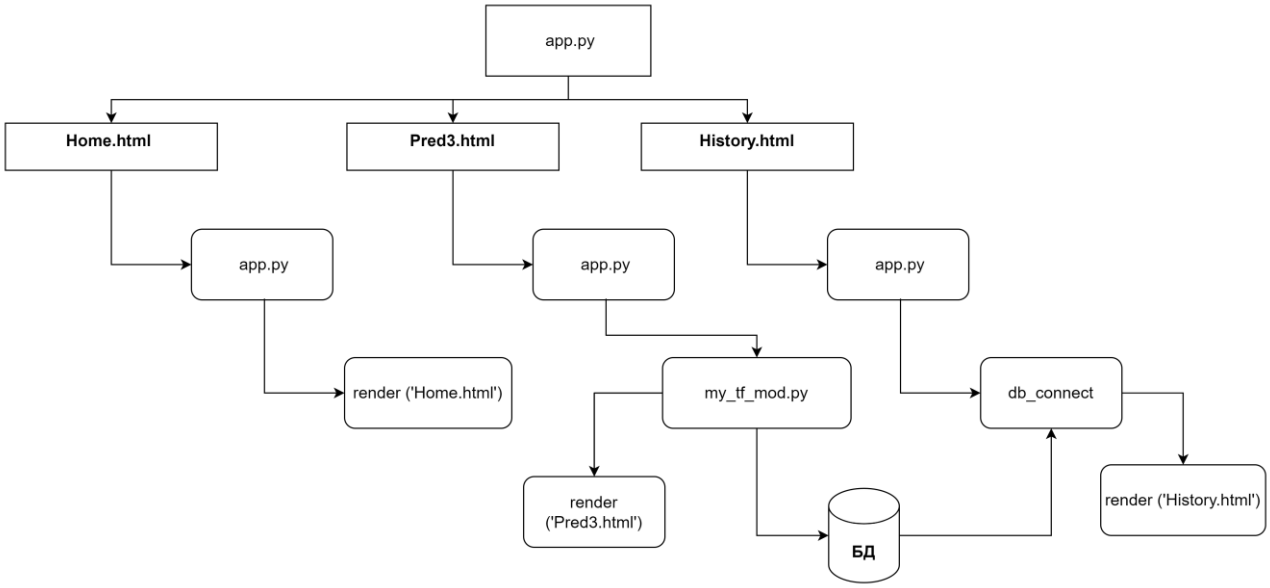


Рисунок 3.4 – Схема формування сторінок

Рядком @app.route('/Prediction', methods=['GET', 'POST']) активізується метод отримання даних та після зчитування імені файлу відбувається його запис: image.save(image.filename).

Після завантаження запускається модуль розмітки зображення, який визначає всі об’єкти на зображенні: my_tf_mod.classify_fruit(img). Які потім заносяться до БД.

При натисканні на зображення хмари зі стрілкою, користувач відкриває вікно для перегляду сховищ пристрою, на якому запущено програму (рис 3.5).

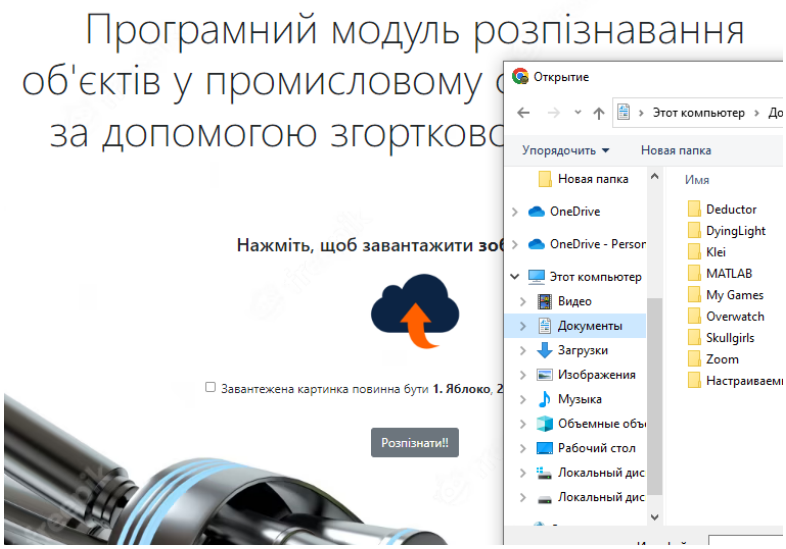


Рисунок 3.5 – Відкриття додатком сховища

Далі користувач повинен обрати рисунок, який потрібно проаналізувати. Після чого нажати на кнопку, що підтверджує те, що завантажений рисунок належить до одного з трьох класів (рис. 3.6).

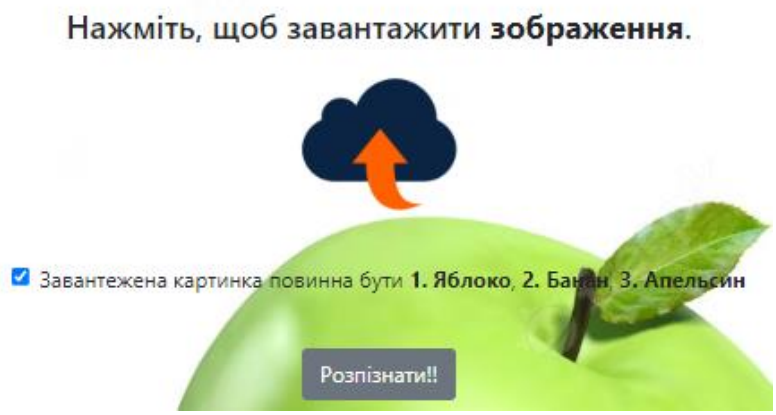


Рисунок 3.6 – Вигляд сторінки після завантаження на неї рисунка та погодження з умовою

Також якщо користувач не погоджується із умовами, то додаток не дасть продовжити роботу, видавши повідомлення мовою операційної системи (рис. 3.7). Із часом в декілька секунд, повідмлення пропадає.

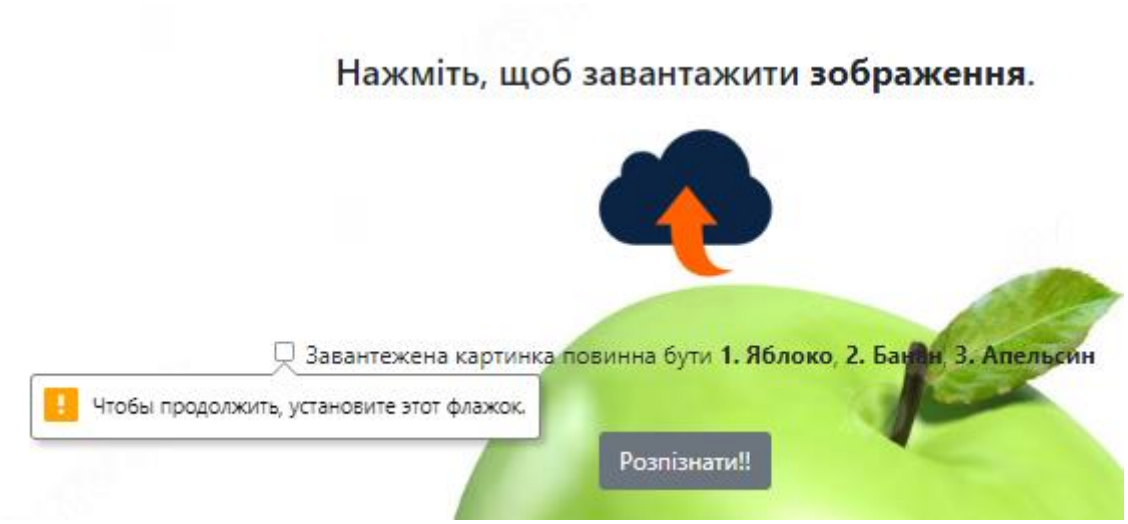


Рисунок 3.7 – Вигляд сторінки в перші секунди в разі непогодження з умовами і спробі розпочати роботу модулю

Потім вкладка браузера завантажується знову і користувач бачить результат роботи модулю (рис. 3.8).

Рисунок 3.8 – Результат роботи програмного модуля

При явному виіднесенні плоду до однієї з груп буде відображено 100% відповідності (рис. 3.9).

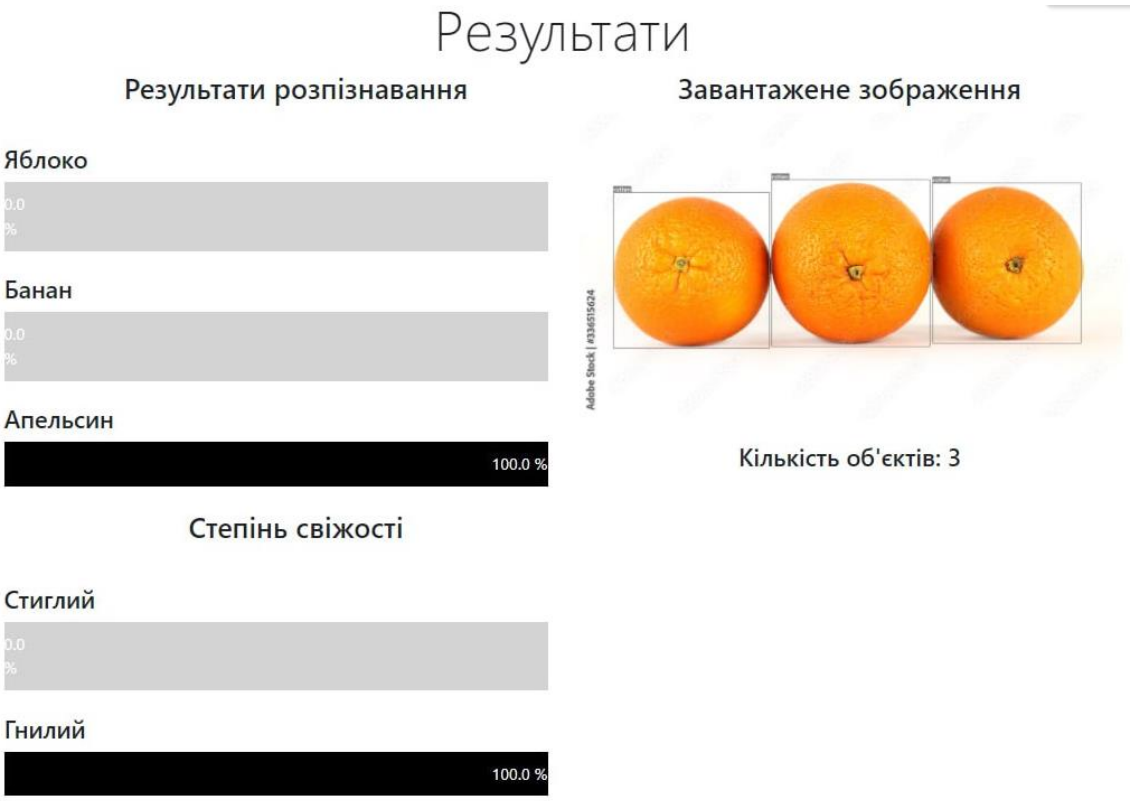


Рисунок 3.9 – Результат роботи програмного модуля з 100% відповідністю до визначеного плоду

Даний модуль може визначити ступінь свіжості плодів (рис. 3.10).

Рисунок 3.10 – Результат роботи програмного модуля з визначенням свіжості.

Використання програмного модуля було опробовано на зображеннях з великою кількістю об’єктів (рис. 3.11), на яких визначалась кількість, ознака

стгиглості (fresh) та ознака гнилості (rotten). Якщо кількість визначених об’єктів більше 1, тоді на вікні відображаються середні значення ознак стаглості і гнилості.

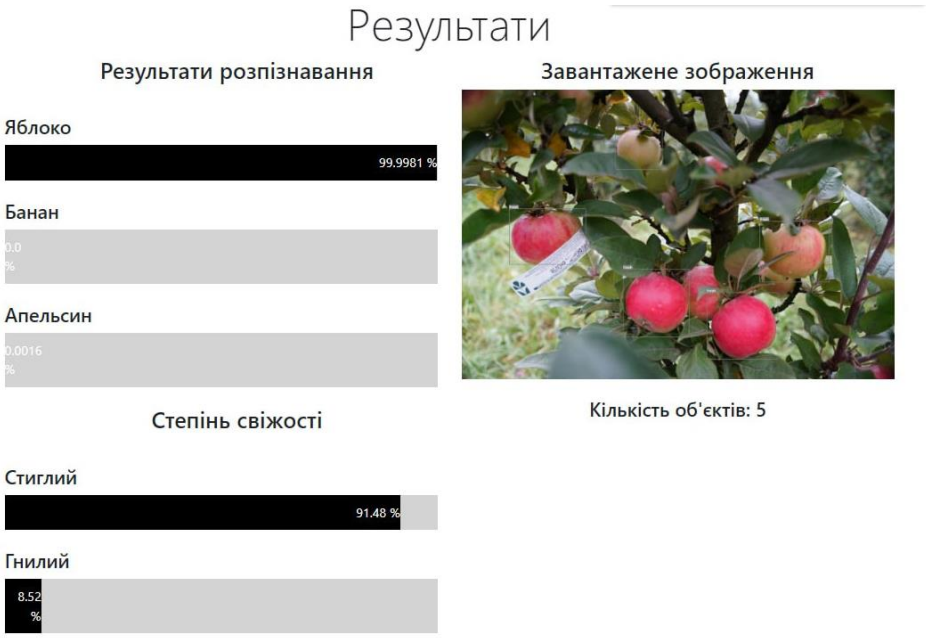


Рисунок 3.11 – Приклад роботи програмного модуля з підрахунком кількості

Для збереження результатів аналізу реалізовано режим відображення історії розпізнавань. Схема генерації вікна на рис. 3.12, а вигляд вікна на рис. 3.13 (дані результатів беруться з БД).

3.3 Навчання нейронної мережі

Ця система розпізнавання фруктів на основі згорткової нейронної мережі має можливість навчання, що дозволяє досягти високої точності та гнучкості у розпізнаванні та класифікації фруктів.

Особливості роботи системи навчання включають наступні правила роботи з моделлю YOLO:

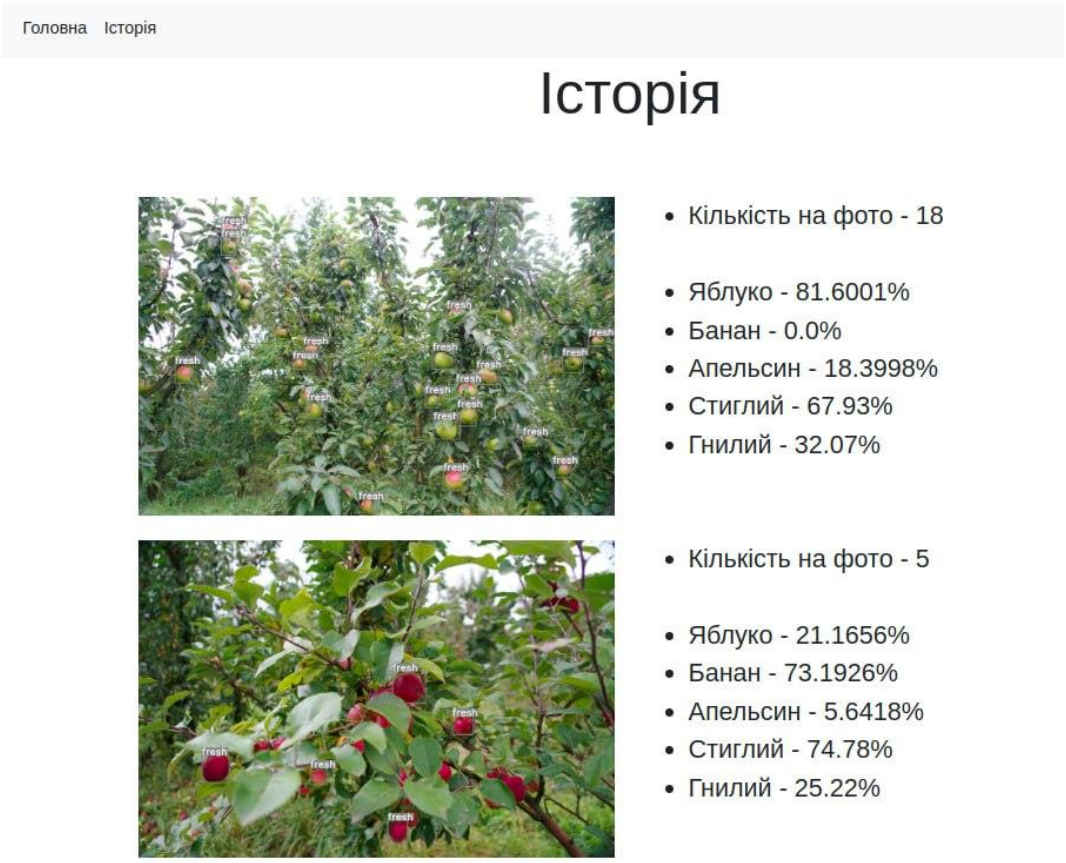


Рисунок 3.12 – Вікно «Історія розпізнавань»

1. Набір даних: для навчання системи розпізнавання фруктів необхідний великий набір зображень фруктів різних видів, що включають різні пози, розміри та особливості. Цей набір даних може бути створений вручну або за допомогою автоматичного збору зображень фруктів.

2. Підготовка даних: Для ефективного навчання моделі необхідно попередньо обробити дані. Це може включати зменшення розмірів зображень, нормалізацію значень пікселів, вирівнювання розмірів зображень тощо.

Для обробки зображень для базового наповнення нейронної мережі було використано безкоштовний ресурс Roboflow (universe.roboflow.com/sopov/fruits-classifier), який дозволяє створити власий проєкт для розмітки (рис. 3.13), завантажити в нього фотографії (рис. 3.14).

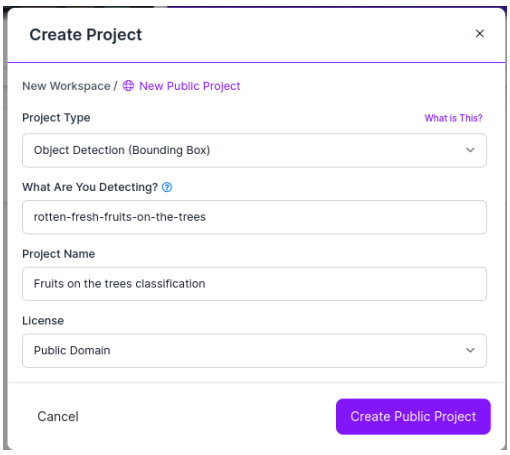


Рисунок 3.13 – Створення проєкту на ресурсі Roboflow

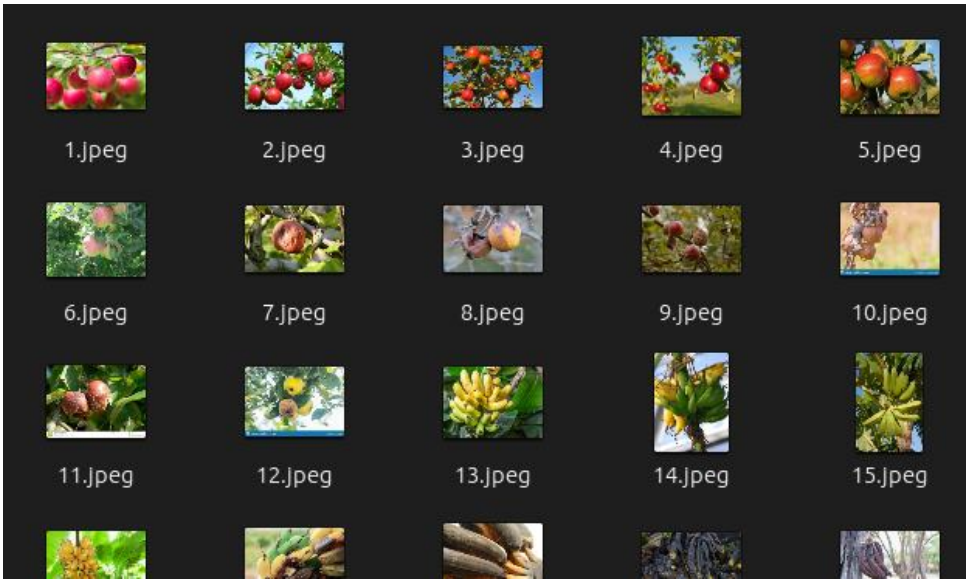


Рисунок 3.14 – Завантаження власних зображень в проєкт на ресурсі Roboflow

Після цього необхідно провести вручну розмітку 4-5 зображень, які дозволять навчити нейронну мережу класам об’єктів (рис. 3.15), обрати співвідношення тестових і валідаційних даних (рис. 3.16), а результат зберегти в форматі YOLO для подальшого використання в НМ (рис. 3.17). Дані для попередньої розмітки завантажувались блоками по 50 фотографій, тому базового навчання розмітці на 9-10% вибірки було достатньо (дану кількість було отримано експериментальним шляхом).

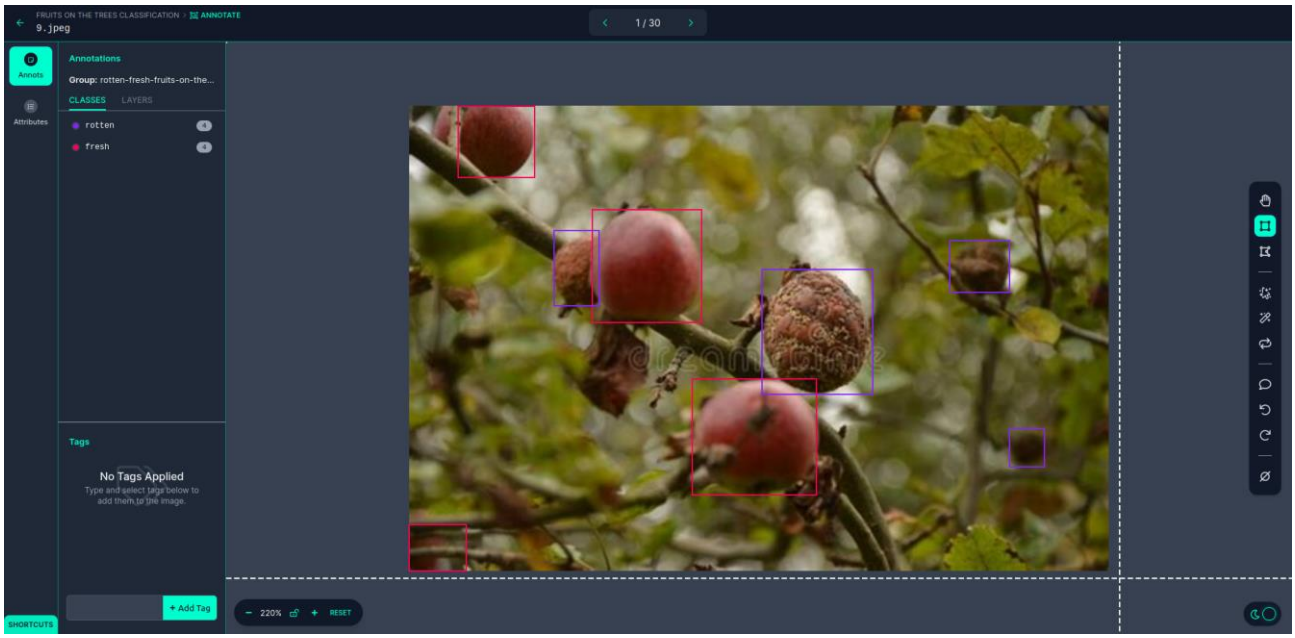


Рисунок 3.15 – Проведення ручної розмітки зображень

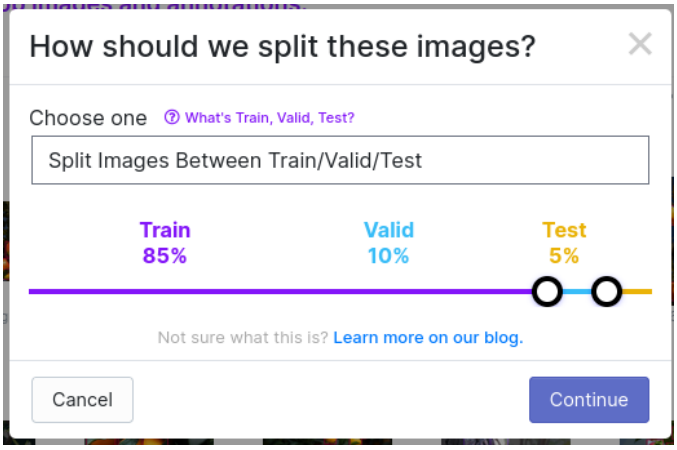


Рисунок 3.16 – Розподіл зображень між тренувальною, валідаційною та тестовими наборами

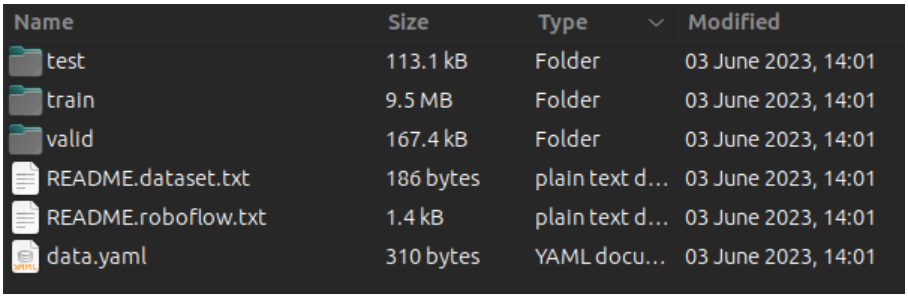


Рисунок 3.17 – Датасет зображень з розміткою: тренувальний, валідаційний та тестовий набори

Для визначення ступеня стиглості фруктів можна використовувати різні датасети, які містять зображення фруктів у різних станах зрілості. Ось кілька відомих датасетів, які можуть бути використані для цієї задачі:

1. Fruits-360: Цей датасет містить більше ніж 90 000 зображень фруктів з 131 різноманітного виду. Кожен фрукт представлений у декількох станах зрілості, від незрілого до дозрілого. Датасет Fruits-360 є широко використовуваним в галузі комп'ютерного зору та машинного навчання.

2. Banana Ripeness Dataset: Цей датасет містить зображення бананів у різних стадіях зрілості. Він включає незрілі, дозрілі та перезрілі банани. Цей датасет спеціалізується на визначенні ступеня зрілості бананів і може бути корисним для вивчення впливу зрілості на класифікацію.

Ці датасети можна використовувати для тренування моделі розпізнавання ступеня стиглості фруктів. Вони надають різноманітність зображень фруктів у різних станах зрілості, що дозволяє моделі навчитися впізнавати та класифікувати їх. Однак, варто також збирати власні дані, оскільки це дозволяє збільшити різноманітність зображень та адаптувати модель до конкретних умов і типів фруктів, що використовуються в промислових фруктових садах.

Всього для навчання нейронної мережі на визначення класу об’єкту (типу плодів) було відібрано 203 зображення яблук, 121 зображеення апельсинів та 109 зображень бананів. Після обробки зображень і їх трансформації (обертання, прибирання шумів, обрізка) в базі утворилось більше 900 зображень. На цих даних проведене навчання на визначення параметрів об’єктів (стиглість (fresh), спорченість (rotten)).

3. Архітектура згорткової нейронної мережі: в системі використовується згорткова нейронна мережа, яка має спеціалізовані шари для виявлення різних ознак фруктів, таких як форма, текстура та колір. На основі рекомендацій в документації [11] було налаштовано архітектуру, яка найбільше підходить для потреб проєкту (рис. 3.18).

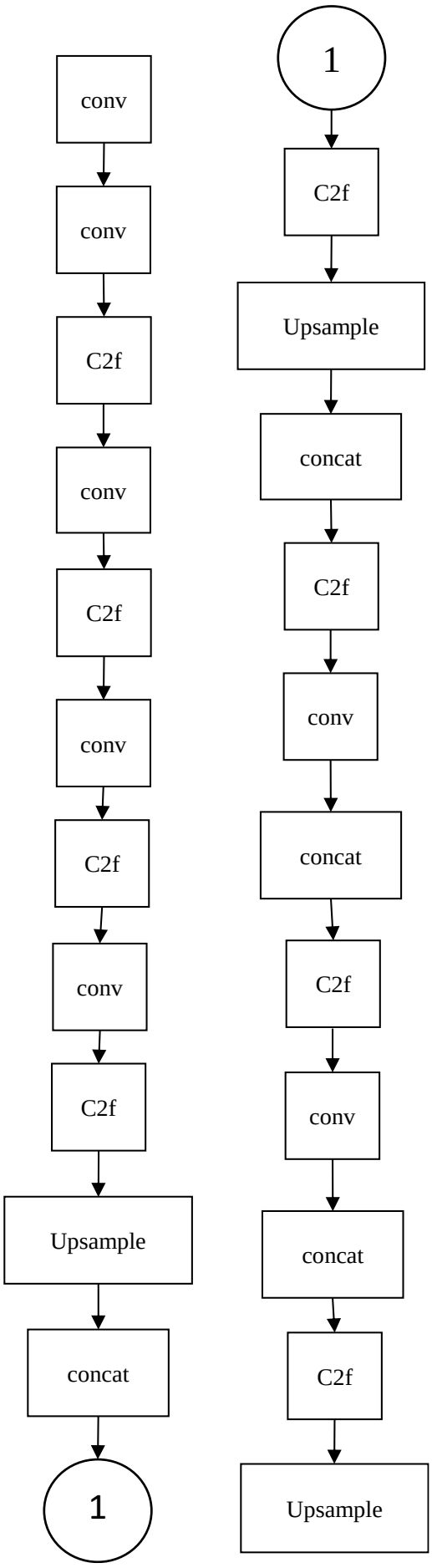


Рисунок 3.18 – Використана архітектура нейронної мережі YOLO

У відображенні моделі використовуються наступні позначення:

– conv – вказує на шар згортки (convolutional layer). Шар згортки використовує фільтри для обробки зображення і вилучення різних характеристик зображення, таких як кольори, форми і текстури;

– C2f – об'єднання виходів з різних шарів згортки (convolutional layers). Виходи від кількох шарів згортки з різними розмірами фільтрів, можуть бути об'єднані, щоб створити більш комплексні характеристики або поєднати різні аспекти зображення;

– concat – процес об'єднання виходів з різних шарів або функцій (виходи з різних шарів або функцій з’єднуються в один вектор або матрицю, що дозволяє об'єднати інформацію з різних джерел).

Описання архітектури нейронної мережі:

1. Conv: Перша згорткова операція.

2. Conv: Друга згорткова операція.

3. C2f: Перехід від 3-х попередніх шарів до одного шару шляхом об'єднання їх в одне представлення.

4. Conv: Згорткова операція.

5. C2f: Перехід від попередніх шарів до одного шару.

6. Conv: Згорткова операція.

7. C2f: Перехід від попередніх шарів до одного шару.

8. Conv: Згорткова операція.

9. C2f: Перехід від попередніх шарів до одного шару.

10. Upsample: Операція збільшення розміру зображення.

11. Concat: Об'єднання даних з попередніх шарів з результатом операції Upsample.

12. C2f: Перехід від попередніх шарів до одного шару.

13. Upsample: Операція збільшення розміру зображення.

14. Concat: Об'єднання даних з попередніх шарів з результатом операції Upsample.

15. C2f: Перехід від попередніх шарів до одного шару.

16. Conv: Згорткова операція.

17. Concat: Об'єднання даних з попередніх шарів з результатом операції Conv.

18. C2f: Перехід від попередніх шарів до одного шару.

19. Conv: Згорткова операція.

20. Concat: Об'єднання даних з попередніх шарів з результатом операції Conv.

21. Upsample: Операція збільшення розміру зображення.

Функція втрат: для навчання моделі використовується підбір оптимальної функції втрат. Ця функція дозволяє виміряти різницю між прогнозованими значеннями та очікуваними мітками класів. Метою навчання є мінімізація цієї функції втрат шляхом коригування ваг моделі.

Оптимізатор: для оптимізації навчання використовуються алгоритми оптимізації, такі як стохастичний градієнтний спуск (SGD) та Adam.

Оцінка результатів: після навчання моделі необхідно оцінити її ефективність, що включає оцінку точності, відповідності метрикам, таким як точність, витрати, F-міра (комбінує дві основні метрики: точність (precision) і повноту (recall) в одне число, що відображає компроміс між цими двома метриками.). Оцінка результатів дозволяє виявити можливі проблеми та вдосконалити модель.

На рис. 3.19 наведено тільки початкові і кінцеві епохи навчання (наведено тільки початкові і останні епохи.

Для оцінки навчання використовуються наступні метрики:

– box loss (втрата позиції або координат) – вимірює втрати або помилки в точності розташування або координат об'єктів на зображенні. Вона відображає, наскільки точно модель може визначити позицію і розмір об'єктів на зображенні порівняно з правильною міткою;

```
Epoch    GPU_mem   box_loss   cls_loss   dfl_loss
1/100      13.7G      1.684      2.321      1.934
           Class     Images  Instances      Box(P
             all         76        432       0.72

Epoch    GPU_mem   box_loss   cls_loss   dfl_loss
2/100      13.9G      1.408      1.602      1.647
           Class     Images  Instances      Box(P
             all         76        432      0.481

Epoch    GPU_mem   box_loss   cls_loss   dfl_loss
3/100        14G      1.383        1.5      1.592
           Class     Images  Instances      Box(P
             all         76        432      0.331

Epoch    GPU_mem   box_loss   cls_loss   dfl_loss
4/100      14.2G      1.398      1.446      1.592
           Class     Images  Instances      Box(P
             all         76        432      0.671

Epoch    GPU_mem   box_loss   cls_loss   dfl_loss
5/100        14G      1.388      1.385      1.568
           Class     Images  Instances      Box(P
             all         76        432      0.679

Epoch    GPU_mem   box_loss   cls_loss   dfl_loss
6/100      13.9G      1.346      1.344      1.543
           Class     Images  Instances      Box(P
             all         76        432      0.641

Epoch    GPU_mem   box_loss   cls_loss   dfl_loss
7/100      14.2G      1.349      1.325      1.548
           Class     Images  Instances      Box(P
             all         76        432      0.679
```

А)

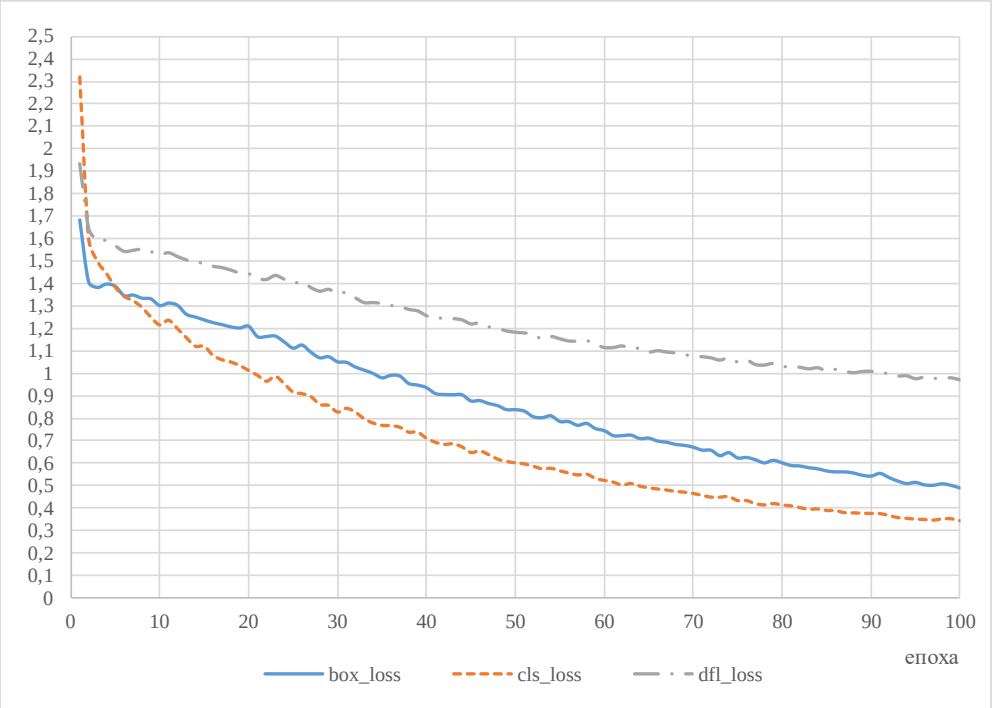
```
   Epoch    GPU_mem   box_loss   cls_loss   dfl_loss
  95/100      14.1G     0.5136     0.3526     0.9761
              Class     Images  Instances      Box(P
                all         76        432      0.768

   Epoch    GPU_mem   box_loss   cls_loss   dfl_loss
  96/100        14G     0.5024     0.3512      0.983
              Class     Images  Instances      Box(P
                all         76        432      0.807

   Epoch    GPU_mem   box_loss   cls_loss   dfl_loss
  97/100      14.1G     0.5006     0.3482     0.9773
              Class     Images  Instances      Box(P
                all         76        432      0.787

   Epoch    GPU_mem   box_loss   cls_loss   dfl_loss
  98/100      13.9G     0.5074     0.3538     0.9786
              Class     Images  Instances      Box(P
                all         76        432      0.801

   Epoch    GPU_mem   box_loss   cls_loss   dfl_loss
  99/100        14G     0.5002     0.3545     0.9801
              Class     Images  Instances      Box(P
                all         76        432      0.819

   Epoch    GPU_mem   box_loss   cls_loss   dfl_loss
 100/100      14.1G     0.4885     0.3458      0.971
              Class     Images  Instances      Box(P
                all         76        432        0.8
```

Б)

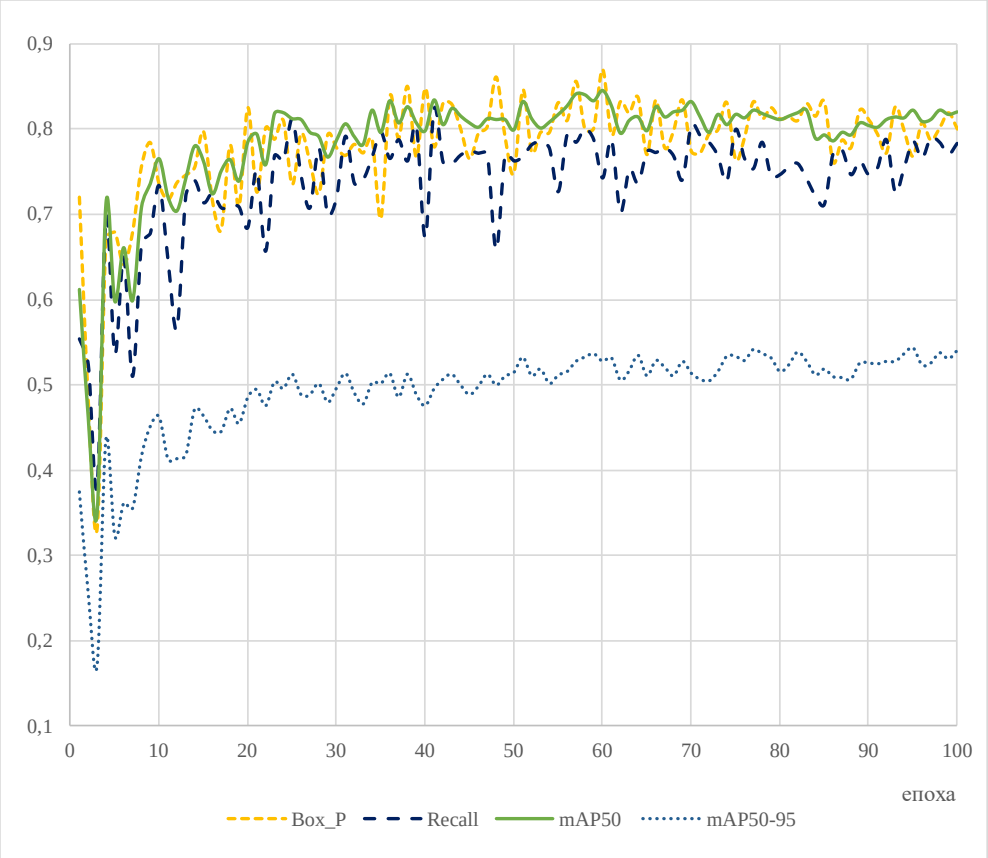
Рисунок 3.19 – Оцінка результатів навчання НМ:

А) епохи 1-7, Б) епохи 95-100

Більш наочно дані рис. 3.21 представлено на графіку (рис. 3.22)).



А)



Б)

Рисунок 3.20 – Метрики начання НМ: А) box-loss, cls-loss, dfl-loss , Б) Box-P, Recall, mAP50, mAP50-95

– cls loss (втрата класифікації) – вимірює втрати або помилки в класифікації об'єктів на зображенні. Вона відображає, наскільки точно модель може визначити клас або категорію об'єкта на зображенні порівняно з правильною міткою;

– dfl loss (втрата зсуву маски або ключових точок) – вимірює втрати або помилки в точності визначення зсуву маски або ключових точок об'єкта на зображенні. Вона оцінює, наскільки точно модель може визначити зсув або зміщення маски об'єкта або ключових точок порівняно з правильною міткою;

– Box_P (Box Precision) – вимірює точність знаходження меж об'єктів. Вона вказує, наскільки точно модель визначає межі об'єктів.

– Recall – вимірює покриття (здатність виявити) об'єктів моделлю. Це відношення кількості виявлених об'єктів до загальної кількості існуючих об'єктів.

– mAP50 (Mean Average Precision at IoU 0.5) – оцінює середню точність виявлення об'єктів при пороговому значенні перекриття (IoU) 0.5 (враховує точність виявлення об'єктів для різних класів та обчислює середнє значення);

– mAP50-95 (Mean Average Precision from IoU 0.5 to 0.95) оцінює середню точність виявлення об'єктів для діапазону порогових значень перекриття (IoU) від 0.5 до 0.95 (враховує точність виявлення об'єктів для різних класів та обчислює середнє значення).

Модель тренувалась майже 2 години в ідеальних умовах для тренування НМ. Результатом тренування є моделі last.pt (остання епозха) та best.pt (найкраща епоха).

Last.pt має останні параметри тренування, але її мінус в тому що модель може бути перетренована, а best.pt має ваги тієї епохи що показала найкращі результати, тобто мінімальний validation loss. У нашому розпізнаванні використовується саме ця модель, так як вона показала себе у точності краще ніж остання.

Перенавчання (overfitting) є типовою проблемою при навчанні нейронних мереж і виникає, коли модель надто точно відтворює тренувальні дані, але

показує погані результати на нових, невиданих даних. Це могло призвести до втрати загальної здатності до узагальнення та неправильних прогнозів.

Для вирішення цієї проблеми в YOLO використовується регуляризація (управління складністю моделі шляхом додавання додаткових обмежень до ваг моделі). Для розробленої НМ використано метод L1 регуляризації, який додавав штраф до функції втрати для обмеження значень ваг.

У методі L1, штрафним членом є сума абсолютних значень ваг моделі, помножена на коефіцієнт регуляризації alpha. Цей штраф змушує деякі ваги стати нульовими, що призводить до розрідженості (sparse) моделі, тобто деякі ознаки не беруться до уваги в процесі прийняття рішень.

3.4 Тестування програмного модуля

Вся робота була покрита автотестами, коди яких наведено в Додатку В.

У тестах перевірялись наступні параметри:

– правильність підготовки та обробки зображення;

– вірність визначення ступеня спорченості фрукта;

– правильність класифікації фруктів

– відповідність обробки сторінкою запитів GET та POST (чи повертають вони очікуваний HTTP-статус та чи містять вони необхідні дані у відповіді).

Тестування на додаткових даних показало можливості помилки системи (рис. 3.21).

В ході тестувань виявлено, що система визначила мочені яблука як стиглі, що може бути викликано недостатньою інформацією для правильного визначення стиглості і відсутність прикладів у навчанні для мочених яблук.

Щоб поліпшити точність визначення стиглості, необхідно зробити наступні кроки:

Рисунок 3.21 – Помилкове визначення кількості і свіжості

– розширення набору даних: збільшення розмаїття навчальних даних може допомогти системі навчитися розрізняти між стиглими та моченими яблуками, а додавання зображень до навчального набору може поліпшити здатність системи розпізнавання;

– використання більш складної моделі: більш складні моделі можуть мати кращу здатність розрізняти між різними станами яблук;

– перевірка параметрів навчання: необхіжно перевірит параметри навчання моделі, такі як швидкість навчання, регуляризація та інші;

– використання додаткових характеристик: включення додаткових характеристик об’єктів, таких як текстура, в якості вхідних даних для системи розпізнавання.

3.5 Висновки до розділу 3

Розроблена система розпізнавання фруктів на основі згорткової нейронної мережі має можливість навчання, що дозволяє досягти високої

точності та гнучкості у розпізнаванні та класифікації фруктів. Особливості роботи системи навчання включають наступне:

– набір даних: для навчання системи розпізнавання фруктів необхідний великий набір зображень фруктів різних видів, що включають різні пози, розміри та особливості. Цей набір даних використовується для тренування нейронної мережі та налаштування її вагів.

– архітектура згорткової нейронної мережі: Використовується архітектура згорткової нейронної мережі, яка має спеціальні шари для виявлення різних ознак фруктів, таких як форма, текстура, колір тощо. Ця архітектура дозволяє впевнено класифікувати та розпізнавати фрукти на зображеннях.

– процес тренування: система навчається шляхом підгонки вагів нейронної мережі до оптимальних значень за допомогою набору тренувальних даних. Під час тренування ваги змінюються таким чином, щоб мережа навчилася розпізнавати та класифікувати фрукти з високою точністю.

– зворотне поширення помилки: для тренування системи використовується алгоритм зворотного поширення помилки, який дозволяє коригувати ваги мережі залежно від різниці між прогнозованими та очікуваними результатами.

ВИСНОВКИ

Обрано тему розроблення модулю розпізнавання фруктів, що являтиме собою підхід до вирішення завдань визначення стиглості та кількості фруктів у промислових фруктових садах. Аналіз загальної задачі розпізнавання фруктів підтверджує актуальність даної розробки, оскільки існуючі програми надають обмежені можливості і не враховують всіх необхідних факторів, таких як виявлення стану фруктів та інші додаткові функції.

При розробці модуля обрано оптимальні сучасні методи згорткових нейронних мереж, що дозволяють отримати високу точність розпізнавання та узагальнення на різні види фруктів.

Отже, розробка даного модуля є актуальною і важливою для покращення процесу розпізнавання фруктів, визначення їх стану та підрахунку кількості, що сприятиме ефективному управлінню промисловими фруктовими садами та покращенню якості збору врожаю.

У розділі було проведено розробку архітектури системи розпізнавання плодів. Під час функціонального аналізу були визначені головна мета функціонування системи і побудована карта процесів роботи застосунку. Для детальнішого опису функціональності була створена діаграма процесу ЯК-БУДЕ, де виділені фрагменти, які раніше не розглядались.

Архітектура інформаційної системи розпізнавання плодів має просту структуру, засновану на базі даних, що включає таблиці з інформацією про плоди. За допомогою концептуальної та логічної моделей, розроблених у цьому розділі, можна визначити структуру бази даних. Для системи необхідні таблиці, що містять дані про плоди, такі як назви, опис, властивості тощо.

Також система повинна мати таблицю для зберігання результатів розпізнавання плодів, де будуть міститися дані про плоди, які були аналізовані, та відповідні результати. Ця таблиця дозволить зберігати і використовувати інформацію про розпізнані плоди для подальшого аналізу та відображення результатів.

Описані таблиці можуть бути використані для реалізації різних функцій системи, таких як пошук плодів за їх характеристиками, фільтрація за певними параметрами, створення звітів про розпізнавання плодів тощо. Це дозволить користувачам системи здійснювати ефективний пошук та аналіз розпізнаних плодів з урахуванням їх властивостей.

Розроблена система розпізнавання фруктів на основі згорткової нейронної мережі має можливість навчання, що дозволяє досягти високої точності та гнучкості у розпізнаванні та класифікації фруктів. Особливості роботи системи навчання включають наступне:

– набір даних: Для навчання системи розпізнавання фруктів необхідний великий набір зображень фруктів різних видів, що включають різні пози, розміри та особливості. Цей набір даних використовується для тренування нейронної мережі та налаштування її вагів.

– архітектура згорткової нейронної мережі: Використовується архітектура згорткової нейронної мережі, яка має спеціальні шари для виявлення різних ознак фруктів, таких як форма, текстура, колір тощо. Ця архітектура дозволяє впевнено класифікувати та розпізнавати фрукти на зображеннях.

– процес тренування: Система навчається шляхом підгонки вагів нейронної мережі до оптимальних значень за допомогою набору тренувальних даних. Під час тренування ваги змінюються таким чином, щоб мережа навчилася розпізнавати та класифікувати фрукти з високою точністю.

– зворотне поширення помилки: Для тренування системи використовується алгоритм зворотного поширення помилки, який дозволяє коригувати ваги мережі залежно від різниці між прогнозованими та очікуваними результатами.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Andriyanov, N. Development of Apple Detection System and Reinforcement Learning for Apple Manipulator. Electronics 2023, 12, 727. https://doi.org/10.3390/electronics12030727

2. Azizah, L., Umayah, S., Riyadi,S., Damarjati, C., Utama, N. Deep learning implementation using convolutional neural network in mangosteen surface defect detection. 7th IEEE Int. Conf. Control Syst., Comput. Eng. (ICCSCE), 2017. – pp. 242–246.

3. Mohapatra,A.,Shanmugasundaram, S., Malmathanraj R. Grading of ripening stages of red banana using dielectric properties changes and image processing approach. Comput. Electron. Agricult., 2017. – 143(382) – pp. 100–110.

4. Lu,J., Hu, J., Zhao, G., Mei, F. & Zhang, C. (2017). An in-field automatic wheat disease diagnosis system. Comput. Electron. Agricult., 2017. – 142(1). – pp. 369–379.

5. Zhu, H., Liu, Q., Qi, Y., Huang, X., Jiang, F., Zhang, S. Plant identification based on very deep convolutional neural networks. Multimed Tools Appl., 2018. – 77(1). – pp. 27779-29797.

6. N. A. Muhammad, A. A. Nasir, Z. Ibrahim, and N. Sabri, “Evaluation of CNN , Alexnet and GoogleNet for FruitRecognition,” IJEECS , 2018. – vol. 12, no. 2. – pp. 468–475.

7. R. D. Safiyah, Z. A. Rahim, S. Syafiq, Z. Ibrahim, and N. Sabri, “Performance Evaluation for Vision-BasedVehicle Classification Using Convolutional Neural Network,” IJET , vol. 7, pp. 86–90, 2018.

8. Y. Lecun, Y. Bengio, and G. Hinton, “Deep learning,” Nature , vol. 521, no. 7553, pp. 436–444, 2015.

9. J. Ubbens, M. Cieslak, P. Prusinkiewicz, and I. Stavness, “The use of plant models in deep learning: An applicationto leaf counting in rosette plants,” Plant Methods, vol. 14, no. 1, pp. 1–10, 2018.

10. Kutyrev A., Kiktev N., Kalivoshko O., Rakhmedov R. Recognition and Classification Apple Fruits Based on a Convolutional Neural Network Model. – Information Technology and Implementation (IT&I-2022), November 30 - December 02, 2022, pp. 90-101.

11. Andriyanov N., Khasanshin I., Utkin D., Gataullin T., Ignar S., Shumaev V., Soloviev V. Intelligent System for Estimation of the Spatial Position of Apples Based on YOLOv3 and Real Sense Depth Camera D415. Symmetry 2022, 14(1), 148; https://doi.org/10.3390/sym14010148.

12. Koval B., Khlevna I. Anomalies Analysis and Detection Using Computer Vision for Finding Defects in Plant Leaves Images. – Information Technology and Implementation (IT&I-2022), November 30 - December 02, 2022, pp. 69-79.

13. Sabri N., Abdul Aziz Z., Ibrahim Z., Akmal Rasydan M.a.R.б Abd Ghani A.H. Comparing Convolution Neural Network Models for Leaf Recognition, International Journal of Engineering and Technology (IJET) , vol.7, pp. 141–144, 2018.

14. A. Kortylewski, B. Egger, A. Schneider, T. Gerig, A. Morel-Forster, and T. Vetter, “Empirically Analyzing theEffect of Dataset Biases on Deep Face Recognition Systems,” 2017.

15. N. Ateqah, B. Mat, N. Hidayah, B. Abd, and Z. Ibrahim, “Celebrity Face Recognition using Deep Learning,”, IJEECS , vol. 12, no. 2, pp. 476–481, 2018.

16. I. Sutskever and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks.", In Advancesin Neural Information Processing Systems , pp. 1097-1105, 2012.

17. K. S. George and S. Joseph, “Text Classification by Augmenting Bag of Words (bow) Representation with Co-Occurrence Feature”, IOSR Journal of Computer Engineering (IOSR-JCE), Vol 16, No 1, pp 34-38., 2014.[10]

18. J. Sivic and A. Zisserman, “Video Google: A text retrieval approach to object matching in videos,” Towar. Categ.Object Recognit. , no. Iccv, pp. 1470–1477, 2003.

19. C. S. Venegas-Barrera and J. Manjarrez, “Visual Categorization with Bags of Keypoints,” Rev. Mex. Biodivers. ,vol. 82, no. 1, pp. 179–191, 2011.

20. C. Hiba, Z. Hamid, and A. Omar, “Bag of Features Model Using the New Approaches : A Comprehensive Study”, International Journal Advances Computer Science and Applications , vol. 1, no. 7, pp. 226–234, 2016.

21. Z. Ibrahim, N. Sabri and D. Isa, “Palm Oil Fresh Fruit Bunch Ripeness Grading Recognition Using Convolutional Neural Network”, Journal of Telecommunication, Electronic & Computer Engineering , Vol 9, No. 3-2, 2018, pp.109-113.

22. Mureșan, Horea and Oltean, Mihai. "Fruit recognition from images using deep learning". Acta UniversitatisSapientiae, Informatica. 10. 26-42. 10.2478/ausi-2018-0002, 2018.

23. E. Okafor, P. Pawara, F. Karaaba, and O. Surinta, “Comparative Study Between Deep Learning and Bag of VisualWords for Wild-Animal Recognition.”, In IEEE Symposium Series for Computational Intelligence (SSCI) , pp. 1-8,2016.

24. S. Hwang, “Bag-of-visual-words approach based on SURF features to polyp detection in wireless capsuleendoscopy videos,” Proc. 2011 Int. Conf. Image Process. Comput. Vision, Pattern Recognition, IPCV 2011, vol. 2,no. i, pp. 941–944, 2011.

25. Z. Ibrahim, N. Sabri, and N. N. A. Mangshor, “Leaf Recognition using Texture Features for Herbal PlantIdentification”. Indonesian Journal of Electrical Engineering and Computer Science 9(1), 152-156,2018

26. S. O’Hara and B.A. Draper, “Introduction to the bag of features paradigm for image classification and retrieval”.arXiv preprint arXiv:1101.3354, 2011.

27. C. Zhang, P. Wang, , K. Chen, and J. K. Kämäräinen, “Identity-aware convolutional neural networks for facialexpression recognition”. Journal of Systems Engineering and Electronics, 28 (4), 784-792, 2017.

ДОДАТКИ

Додаток А

Приклад json-файлу сегментації зображення


```
{
  "image": "DSC04593.JPG",
  "annotations": [
    {
      "label": "fresh",
      "coordinates": {
        "x": 2563.67626953125,
        "y": 2261.777099609375,
        "width": 2916.789306640625,
        "height": 2619.0869140625
      }
    },
    {
      "label": "fresh",
      "coordinates": {
        "x": 5027.462890625,
        "y": 3627.76220703125,
        "width": 5310.74169921875,
        "height": 3874.03125
      }
    },
    {
      "label": "fresh",
      "coordinates": {
        "x": 4886.35009765625,
        "y": 4348.85498046875,
        "width": 5148.96337890625,
        "height": 4596.73583984375
```

```
  }
},
{
  "label": "fresh",
  "coordinates": {
    "x": 3444.68505859375,
    "y": 1174.22802734375,
    "width": 3726.768310546875,
    "height": 1409.190673828125
  }
},
{
  "label": "fresh",
  "coordinates": {
    "x": 3853.667236328125,
    "y": 3858.127685546875,
    "width": 4184.00146484375,
    "height": 4044.41455078125
  }
},
{
  "label": "fresh",
  "coordinates": {
    "x": 1503.227783203125,
    "y": 4908.41552734375,
    "width": 2007.071533203125,
    "height": 5282.1845703125
  }
},
{
```

```
      "label": "fresh",
      "coordinates": {
        "x": 7615.62060546875,
        "y": 1043.2939453125,
        "width": 7864.3798828125,
        "height": 1258.516845703125
      }
    }
  ]
}
```

Додаток Б

Програмний код завантжувального модуля

```
import os
import sys
import numpy as np
import numpy
import cv2
from flask import Flask, render_template, request
from werkzeug.utils import secure_filename

from keras.preprocessing import image
from keras.models import load_model
from PIL import Image, ImageFile
import my_tf_mod
import get_counts
from io import BytesIO
import matplotlib.pyplot as plt
import base64

app = Flask(__name__)

@app.route('/')
def home():
    return render_template('home.html')

@app.route('/Prediction', methods=['GET','POST'])
def pred():
    if request.method=='POST':
```

```
        file=request.files['file']
        #  image = request.files['file']

        org_img, img= my_tf_mod.preprocess(file)
        file.seek(0)
        image=file
        image.save(image.filename)

        print(img.shape)
        fruit_dict=my_tf_mod.classify_fruit(img)
        rotten=my_tf_mod.check_rotten(img)
        counts=get_counts.get_count(cv2.imread(image.filename))

        os.remove(image.filename)
        print(counts)
        img_x=BytesIO()
        plt.imshow(org_img/255.0)
        plt.savefig(img_x,format='png')
        plt.close()
        img_x.seek(0)
        plot_url=base64.b64encode(img_x.getvalue()).decode('utf8')

    return render_template('Pred3.html', fruit_dict=fruit_dict, rotten=rotten,
plot_url=plot_url,counts=counts)
if __name__=='__main__':
    app.run(debug=True)
```

Додаток В

Програмні коди автотестів

Тест перевірки функції preprocess

```
# Підготовка зображення
with open('test_image.jpg', 'rb') as file:
    org_img, processed_img = preprocess(file)
# Перевірка типу та розмірності зображення
assert isinstance(org_img, np.ndarray)
assert isinstance(processed_img, np.ndarray)
assert org_img.shape == (height, width, channels)
assert processed_img.shape == (1, 100, 100, 3)
```

Тест перевірки функції check_rotten

```
# Підготовка зображення
with open('test_image.jpg', 'rb') as file:
    org_img, processed_img = preprocess(file)
# Перевірка визначення ступеня спорченості
probabilities = check_rotten(processed_img)
assert isinstance(probabilities, list)
assert len(probabilities) == 2
assert all(0 <= prob <= 100 for prob in probabilities)
assert sum(probabilities) == 100
```

Тест перевірки сторінки /Prediction з методом POST

```
def test_pred_post():
    with app.test_client() as client:
        with open('test_image.jpg', 'rb') as file:
            data = {'file': (file, 'test_image.jpg')}
            response = client.post('/Prediction', data=data,
content_type='multipart/form-data')
            assert response.status_code == 200
            assert b"Fruit Prediction" in response.data
```

```
assert b"fruit_dict" in response.data
assert b"rotten" in response.data
assert b"plot_url" in response.data
assert b"counts" in response.data
```