Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет комп'ютерних наук
(повна назва)

Кафедра програмної інженерії
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

Дослідження методів класифікації зображень за контентом
(тема)

Виконав:
студент (ка) 2 курсу, групи ІПЗм-22-6

Чайковський С.О.
(прізвище, ініціали)

Спеціальність 121 – Інженерія програмного забезпечення
(код і повна назва спеціальності)
Тип програми освітньо-наукова

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

Допускається до захисту
Зав. кафедри _____ З.В.Дудар
(підпис) (прізвище, ініціали)

2024 р.

Харківський національний університет радіоелектроніки

Факультет комп'ютерних наук
Кафедра програмної інженерії
Рівень вищої освіти другий (магістерський)
Спеціальність 121 – Інженерія програмного забезпечення
Тип програми освітньо-наукова програма
Освітня програма Інженерія програмного забезпечення
(шифр і назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«___» _____ 2024 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Чайковському Сергію Олександровичу
(прізвище, ім'я, по батькові)

1. Тема роботи «Дослідження методів класифікації зображень за контентом»
Затверджена наказом по університету від 29.03.2024р. № 250 Ст
2. Термін подання студентом роботи до екзаменаційної комісії 17.06.2022
3. Вихідні дані до роботи інформація щодо нейронних мереж, галузі машинного навчання, інформація щодо згорткових нейронних мереж, архітектури згорткових нейронних мереж, мова програмування Python, бібліотека TensorFlow, інтерфейс Keras, середовище розробки PyCharm 2023
4. Перелік питань, що потрібно опрацювати в роботі
аналіз та вибір існуючих моделей нейронних мереж, підготовка даних для навчання обраних моделей нейронних мереж, проектування та розробка гібридного класифікатора для проведення експериментальних досліджень, написання програмних рішень, проведення експериментів та аналіз отриманих результатів, формулювання висновку щодо можливості використання гібридного нейромережевого класифікатора.

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1	Аналіз проблемної області та постановка задачі	01.10 – 30.12.23	*виконано*
2	Аналіз та вибір нейронних мереж для дослідження	30.12.23 – 17.01.24	*виконано*
3	Планування експерименту	17.01 – 25.01.24	*виконано*
4	Програмна реалізація експерименту	25.01 – 25.02.24	*виконано*
5	Експериментальні дослідження	25.02 – 01.03.24	*виконано*
6	Аналіз результатів експериментальних досліджень та розробка рекомендацій	01.03 – 15.03.24	*виконано*
7	Написання та оформлення статті та тез доповіді	25.02 – 15.03.24	*виконано*
8	Підготовка пояснювальної записки	15.03 – 07.05.24	*виконано*
9	Підготовка презентації та доповіді	07.05 – 07.06.24	*виконано*
10	Нормоконтроль	07.06 – 08.06.24	*виконано*
11	Рецензування	08.06 – 12.06.24	*виконано*
12	Занесення диплома в електронний архів	14.06.2024	*виконано*
13	Попередній захист	15.06.2024	*виконано*
14	Допуск до захисту у зав. кафедри	15.06.2024	*виконано*

Дата видачі завдання 01 жовтня 2023р.

Студент (ка) _____ Чайковський С.О.
(підпис)

Керівник роботи _____ проф. Смеляков С.В.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ / ABSTRACT

Пояснювальна записка до кваліфікаційної роботи магістра містить: 72 с., 20 рис., 2 табл., 41 джерел.

ВИЯВЛЕННЯ БУР'ЯНІВ, ГІБРИДНІ НЕЙРОМЕРЕЖІ, ЗГОРТКОВІ НЕЙРОННІ МЕРЕЖІ, КЛАСИФІКАТОР, МАШИННЕ НАВЧАННЯ, НЕЙРОННА МЕРЕЖА, ШТУЧНИЙ ІНТЕЛЕКТ, ШТУЧНА НЕЙРОЛННА МЕРЕЖА, ХВОРОБИ ТОМАТУ.

Тепличне господарство відіграє вирішальну роль у задоволенні попиту протягом року, незалежно від кліматичних умов. Однак підтримання здоров’я врожаю в теплицях є критично важливим і складним. Ця стаття присвячена аналізу ефективності класифікатора на основі гібридної нейронної мережі (HNN) для ефективного та своєчасного виявлення бур’янів і хвороб томатів у теплицях. Кожна хвороба рослин має унікальні особливості, які можна розпізнати та класифікувати. Також можна аналізувати зображення рослин на ранніх стадіях для виявлення бур’янів, оскільки бур’яни зазвичай значно відрізняються від сортових рослин. Таким чином, ми можемо ефективно аналізувати стан і тип рослин на кожному етапі вирощування. Моделі HNN можна використовувати для оптимізації використання ресурсів для вирощування одиниці культури. Це також дозволить краще контролювати стан рослин. Крім того, раннє виявлення захворювань значно зменшить надмірне використання агрохімікатів.

WEED DETECTION, HYBRID NEURAL NETWORKS, CONVERSIONAL NEURAL NETWORKS, CLASSIFIER, MACHINE LEARNING, NEURAL NETWORK, ARTIFICIAL INTELLIGENCE, ARTIFICIAL NEURAL NETWORK, TOMATO DISEASES.

Greenhouse farming plays a crucial role in satisfying the demand throughout the year, regardless of climatic conditions. However, maintaining crop health in

greenhouses is critical and challenging. This paper is devoted to analyzing the performance of a hybrid neural network (HNN) based classifier for effective and timely detection of weeds and tomato diseases in greenhouses. Each plant disease has unique features that can be recognized and classified. It is also possible to analyze images of plants in the early stages to detect weeds because weeds are usually significantly different from varietal plants. Thus, we can effectively analyze the condition and type of plants at each stage of cultivation. HNN models can be used to optimize the resource usage for growing a crop unit. It will also allow better monitoring of plant health. In addition, early detection of diseases will significantly reduce the excessive use of agrochemicals.

Заява щодо самостійного виконання кваліфікаційної роботи та можливості її публікації в електронному архіві відкритого доступу ElArKhNURE.

Я, Чайковський Сергій Олександрович, студент гр. ІПЗм-22-6, здобувач вищої освіти на другому (магістерському) рівні кафедри «Програмна інженерія», заявляю: моя кваліфікаційна робота на тему «Дослідження методів класифікації зображень за контентом», що буде представлена в екзаменаційну комісію для публічного захисту, виконана самостійно, в ній не містяться елементи плагіату і вона може бути опублікована в електронному архіві відкритого доступу ElArKhNURE. Всі запозичення з друкованих та електронних джерел мають відповідні посилання.

Я ознайомлений(на) з діючим положенням «Про протидію академічному плагіату в ХНУРЕ», згідно з яким виявлення плагіату є підставою для відмови в допуску кваліфікаційної роботи до захисту та застосування дисциплінарних заходів.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ

ML – Machine Learning

NLP – Natural Language Processing

CV – Computer Vision

AI – Artificial Intelligence

ANM – Artificial Neural Network

FNN – Feedforward Neural Network

MLP – Multilayer Perceptron

CNN – Convolutional Neural Network

RNN – Recurrent Neural Networks

R-CNN – Region-based Convolutional Neural Network

RPN – Reverse Polish Notation

FPN – Feature Pyramid Network

LSTM – Long Short-Term Memory

RBFN – Radial Basis Function Network

GAN – Generative Adversarial Network

HNN – Hybrid Neural Network

TP – True Positive

FN – False Negative

FP – False Positive

AP – Average Precise

SPA – Single Page Application

AWS – Amazon Web Service

API – Application Programming Interface

SDK – Software Development Kit

EKS – Kubernetes

DB – Database

RDS – Relational Data Storage

ALB – Application Load Balancer

CDN – Content Delivery Network

ML – Machine Learning

WAF – Web Application Firewall

CRR – Cross Region Replication

JSON – JavaScript Object Notation

SSO – Single Sign On

SaaS – Software as a service

ВСТУП

У сучасному інформаційному суспільстві, в якому обсяг візуальної інформації постійно зростає, проблеми аналізу та класифікації зображень набувають критичного значення. Справжнім викликом для сучасних технологій стає не лише обробка великого обсягу візуальних даних, але й їхнє розуміння та інтерпретація. В цьому контексті розробка методів класифікації зображень за контентом стає стратегічно важливою для подальшого розвитку цифрового середовища.

На сьогоднішній день, коли ми оточені великою кількістю візуальної інформації, проблема автоматизованого аналізу та класифікації зображень набуває вирішального значення. Швидкість надходження та збільшення обсягів цифрової інформації у різних сферах життя, вимагає новітніх підходів для ефективного управління цими великими потоками даних. Наразі існуючі методи класифікації залишають простір для покращень. Застосування інноваційних технологій, таких як штучний інтелект і нейронні мережі, може стати ефективним рішенням.

У даному дослідженні основна увага приділена тепличному господарству, яке відіграє вирішальну роль у задоволенні попиту протягом року, незалежно від кліматичних умов. Предметом даного дослідження є процеси виявлення хвороб та бур'янів у теплицях на прикладі вирощування томатів за допомогою гібридного нейронно мережевого класифікатора.

Метою роботи є експериментальна оцінка ефективності гібридного нейронно мережевого класифікатора для виявлення хвороб та росту бур’янів у теплиці на прикладі томатів. Задача полягатиме в створенні ефективної системи, здатної розпізнавати ключові аспекти зображень та високі вимоги до точності та швидкості.

Останні тенденції в галузі обробки зображень, зокрема використання глибокого навчання та нейронних мереж, свідчать про перспективність подальших досліджень в цьому напрямку. Актуальність теми обумовлена постійним попитом на томати в усьому світі та необхідністю оптимізації процесу

їх вирощування та відповідає сучасним викликам в обробці великих обсягів візуальних даних.

1 АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ

У цьому розділі буде сформульована проблематика, що розглядається у даній кваліфікаційні роботі, наведені основні відомості щодо предметної галузі дослідження, та сформульована постановка задачі даного дослідження.

1.1 Проблематика що розглядається

Культивований помідор, Solanum lycopersicum L., є найбільш споживаним овочем у світі завдяки його статусу основного інгредієнта багатьох сирих, варених або оброблених харчових продуктів [1]. Томат можна вирощувати в різних географічних зонах у відкритих грядках або теплицях і збирати врожай ручним або механічним способом [1, 2]. Переважно розглядається як однорічна культура, томат має значне комерційне значення в усьому світі [3].

Хоча кількісно визначити глобальне виробництво овочів у теплицях складно, деякі якісні дані можуть показати темпи виробництва томатів. Наприклад, імпорт із Мексики у 2017 році склав майже 84 відсотки (1,8 мільярда фунтів стерлінгів) обсягу тепличних помідорів, які надходять на ринок США. Тим часом імпорт канадських томатів, вирощених у теплицях, залишився на рівні близько 300 мільйонів фунтів [4].

Примітно, що інвестиції в теплиці продовжують зростати, особливо через COVID [5]. Наприклад, у період з 2015 по 2020 рік у Канаді було збільшено площу площ на 23,9%, що призвело до зростання потужностей тепличного виробництва фруктів і овочів до 1809 га у 2020 році. У результаті продажі помідорів зросли на 12,1% [6].

Бум тепличного господарства, однак, не позбавлений викликів. Важливою проблемою є підтримання оптимального здоров’я врожаю в цих контрольованих середовищах [7]. Більше того, враховуючи новий Європейський кліматичний закон, який накладає цілі щодо зменшення використання та ризику хімічних пестицидів на 50% у 2030 році [8], оптимізація використання ресурсів для кожної одиниці культури стає обов’язковою.

Усвідомлюючи ці проблеми та керуючись останніми технологічними досягненнями, у цій роботі представлено класифікатор гібридної нейронної мережі (HNN), керований штучним інтелектом, розроблений для ефективного виявлення хвороб і бур’янів у тепличному вирощуванні томатів [2, 5]. Цей автоматизований інструмент допоможе зробити виявлення хвороб і бур’янів своєчасним, економічно ефективним і безпечнішим для навколишнього середовища.

1.2 Аналіз сфери штучного інтелекту

Штучний інтелект – це галузь комп'ютерних наук, яка фокусується на створенні систем, які можуть виконувати завдання, які зазвичай вимагають інтелекту людини. Головна мета штучного інтелекту полягає в розробці алгоритмів та програм, які дозволяють комп'ютерам аналізувати дані, розуміти контекст, вчитися з досвіду, приймати рішення та вирішувати завдання в реальному часі [9].

Основні характеристики та напрями штучного інтелекту включають:

- машинне навчання (ML) – підгалузь штучного інтелекту, яка дозволяє системам навчатися на основі даних, розпізнавати закономірності та приймати рішення без явного програмування;
- обробка природних Мов (NLP) – системи, що володіють здатністю розуміти, інтерпретувати та генерувати людську мову;
- комп'ютерний зір (CV) – технологія, що надає комп'ютерам можливість розпізнавати, аналізувати та реагувати на візуальні дані, такі як зображення і відео;
- робототехніка (Robotics) – використання штучного інтелекту для програмування роботів та систем автоматизації для виконання різноманітних завдань;
- системи експертизи (Expert Systems) – системи, що моделюють знання та експертні навички у конкретній галузі для прийняття рішень або надання консультацій;

- ігровий штучний інтелект (AI in Gaming) – використання штучного інтелекту для створення ігрових персонажів, адаптивного геймплею та розробки інтелектуальних опонентів у відеоіграх;
- автономні системи – розробка систем, які можуть функціонувати та приймати рішення незалежно від зовнішнього впливу, наприклад, автономні автомобілі.

Данна кваліфікаційна робота сфокусована на напрямку машинного навчання, адже саме цей напрямок включає в себе один з типів машинного навчання, необхідний для даного дослідження, а саме нейронні мережі.

1.3 Аналіз сфери нейронних мереж

Нейронна мережа – це математична модель, яка імітує роботу нервової системи в біологічних організмах. Вона складається з великої кількості з'єднаних штучних нейронів, які організовані в шари і спроможні виконувати завдання машинного навчання [10].

Штучний нейрон – це базовий будівельний блок штучної нейронної мережі, математична модель, яка імітує роботу нейронів у біологічних системах. Він використовується для обробки інформації та прийняття рішень у задачах машинного навчання.

Штучний нейрон приймає вхідні сигнали, кожен із яких має вагу, і сумує їх. Потім до цієї суми застосовується активаційна функція, яка визначає вихідний сигнал нейрона. Цей вихід може бути переданий на вхід іншим нейронам в мережі.

Штучні нейрони організовані в шари, де вхідний шар приймає дані, вихідний шар видає результат, а проміжні шари (приховані) виконують обчислення. Кожен з'єднаний шар має свої ваги, які навчаються під час тренування нейронної мережі.

Ця абстрактна структура штучного нейрона дозволяє нейронним мережам вирішувати різноманітні завдання, такі як розпізнавання образів, класифікація тексту, прогнозування та інші.

Нейрони у нейронних мережах з'єднані з іншими нейронами за допомогою ланок. Кожне з'єднання має вагу, яка вказує, наскільки важливою є інформація, яку передає один нейрон іншому. Однак принцип з'єднань може варіюватися в залежності від типу нейронної мережі.

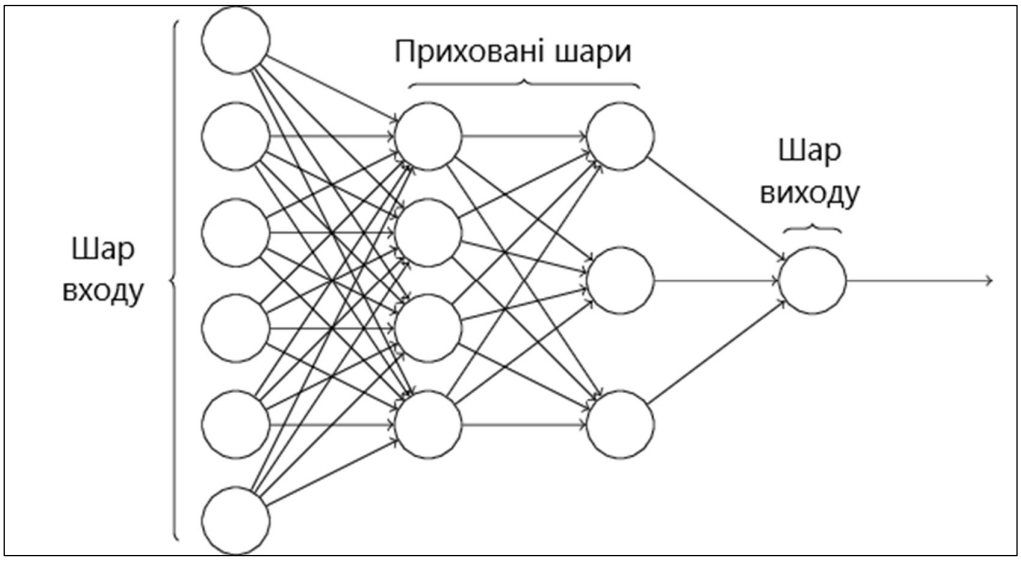

Рисунок 1.1 – Вигляд простої нейронної мережі (за даними [11])

Існує декілька типів штучних нейронних мереж (ANN), які зазвичай використовуються в різних програмах, зокрема [12]:

- нейронні мережі прямого зв’язку (FNN) – інформація рухається в одному напрямку, від вхідного до вихідного рівня. Багатошарові перцептрони (MLP) є поширеним типом прямої мережі;
- згорткові нейронні мережі (CNN) – оптимізовані для обробки зображень та використовують згорткові шари для виявлення локальних характеристик;
- рекурентні нейронні мережі (RNN) – використовуються для роботи з послідовними даними, такими як текст чи часові ряди, і мають здатність запам'ятовування попередніх станів;
- мережі довготривалої короткочасної пам’яті (LSTM) – розширення RNN з додатковими механізмами для управління довгостроковою та короткочасною пам'яттю;

- трансформери – розроблені для обробки природної мови та завдань послідовності. Не вимагають обробки послідовностей по порядку;
- мережі радіальних базисних функцій (RBFN) – використовуються як функції активації. Також часто використовуються в завданнях наближення функцій і класифікації;
- генеративні змагальні мережі (GAN) – містять мережу генератора та мережу дискримінатора, навчені одночасно. GAN використовуються для створення реалістичних синтетичних даних, наприклад зображень.

Для проведення дослідження сфокусуємось на згорткових нейронних мережах, адже саме цей тип нейронних мереж спеціально розроблений для обробки зображень і виявлення в них патернів та характеристик.

1.4 Аналіз сфери згорткових нейронних мереж

Згорткова нейронна мережа (CNN) є типом глибоких штучних нейронних мереж, спеціально розробленим для обробки та аналізу зображень. Цей тип мережі виявився ефективним в розпізнаванні образів і класифікації зображень, завдяки своїй спроможності автоматично виявляти різні характеристики та патерни на зображеннях.

CNN є ключовим інструментом у сфері комп'ютерного зору, розпізнавання образів та обробки зображень. Вона також має кілька особливостей, завдяки яким вона ефективна для обробки зображень [13]:

- згорткові шари для виявлення локальних характеристик – дозволяють мережі автоматично виявляти локальні характеристики та патерни на зображенні, такі як різні текстури, кольори і форми;
- області зі зменшеною розмірністю за допомогою пулінгу – використання шарів пулінгу дозволяє зменшити розмірність зображення, зберігаючи при цьому важливі характеристики. Це полегшує обчислення та скорочує кількість параметрів мережі;
- ієрархічна структура для аналізу об'єктів на різних рівнях – дозволяє виявляти складні об'єкти на різних рівнях абстракції. Наприклад, перші

шари можуть виявляти кольори та текстури, а подальші шари можуть об'єднувати ці характеристики для розпізнавання об'єктів;

- використання функцій активації для нелінійності – застосування функцій активації, таких як ReLU, дозволяє моделі навчатися нелінійним залежностям у даних, що є важливим для адекватного моделювання зображень;
- переносність функцій за допомогою попередньо навчених моделей, що дозволяє використовувати знання, отримане на великих даних, для конкретних завдань.

Ці особливості роблять CNN відмінним інструментом для задач обробки зображень, де важлива локальна ієрархічна інформація та виявлення зразків.

Існує велика кількість архітектур згорткових нейронних мереж, які були розроблені для різних завдань у області обробки зображень та комп'ютерного зору. Детальний огляд та вибір архітектури нейронних мереж для побудови гібридного нейронно мережевого класифікатора буде описано у наступному розділі даної кваліфікаційної роботи.

1.5 Постановка задачі

Основним завданням дослідження є розробка та експериментальна оцінка ефективності гібридного нейронно мережевого класифікатора для виявлення хвороб та росту бур’янів у теплиці томатів.

Для проведення дослідження необхідно:

- провести аналіз та обрати підходящі моделі нейронних мереж;
- провести підготовку даних для навчання обраних моделей нейронних мереж;
- обрати метрики та критерії оцінки ефективності обраних моделей нейронних мереж;
- розробити модель гібридної нейронної мережі для виявлення хвороб та бур'янів у теплицях на основі зібраних даних;

- провести тренування обраного набору моделей нейронних мереж на підготовленому наборі даних;
- провести експериментальну оцінку ефективності розробленого гібридного нейронно мережевого класифікатора на реальних даних;
- провести аналіз отриманих результатів, визначити точність та швидкість роботи гібридного нейронно мережевого класифікатора;
- провести порівняння обраних архітектур моделей нейронних мереж для виявлення переваг та обмежень кожної з них та розробити рекомендації щодо використання гібридного класифікатора;
- на основі отриманих результатів сформулювати висновки щодо можливості використання гібридного нейронно мережевого класифікатора для виявлення хвороб та бур'янів у теплицях при вирощуванні томатів.

2 ОПИС ПРИЙНЯТИХ ПРОЄКТНИХ РІШЕНЬ

У цьому розділі проведено аналіз та опис прийнятих проектних рішень для виконання кваліфікаційної роботи.

Дане дослідження має на меті експериментальну оцінку ефективності гібридного нейронно мережевого класифікатора для виявлення хвороб та росту бур'янів у теплиці томатів. Для побудови гібридного нейронно мережевого класифікатора потрібно виконати три завдання:

- виявити листя рослин;
- виявити потенційні бур'яни;
- виявити потенційно можливі захворювання.

2.1 Аналіз та вибір згорткових нейронних мереж для виявлення листя

Для завдання виявлення листя було розглянуто дві моделі нейронних мереж.

YOLO («You Look Only Once») – це новаторська архітектура згорткової нейронної мережі (CNN), розроблена для виявлення об'єктів у реальному часі.

Дана архітектура – дуже популярна архітектура CNN, яка зараз використовується для розпізнавання кількох об'єктів на зображенні. Головною особливістю цієї архітектури в порівнянні з іншими є те, що більшість систем застосовують CNN кілька разів до різних регіонів зображення.

У YOLO CNN застосовується один раз до всього зображення. Мережа ділить зображення на сітку та передбачає обмежувальні рамки та ймовірність того, що в кожній області є потрібний об'єкт.

Переваги цього підходу полягають у тому, що мережа розглядає все зображення одночасно та враховує контекст при ідентифікації та розпізнаванні об'єкта.

YOLO складається з серії згорткових шарів, включаючи з'єднання швидкого доступу та з'єднання пропуску, що робить її глибокою нейронною

мережею. Вона прогнозує об’єкти в трьох різних масштабах, дозволяючи обробляти малі та великі об’єкти на одному зображенні.

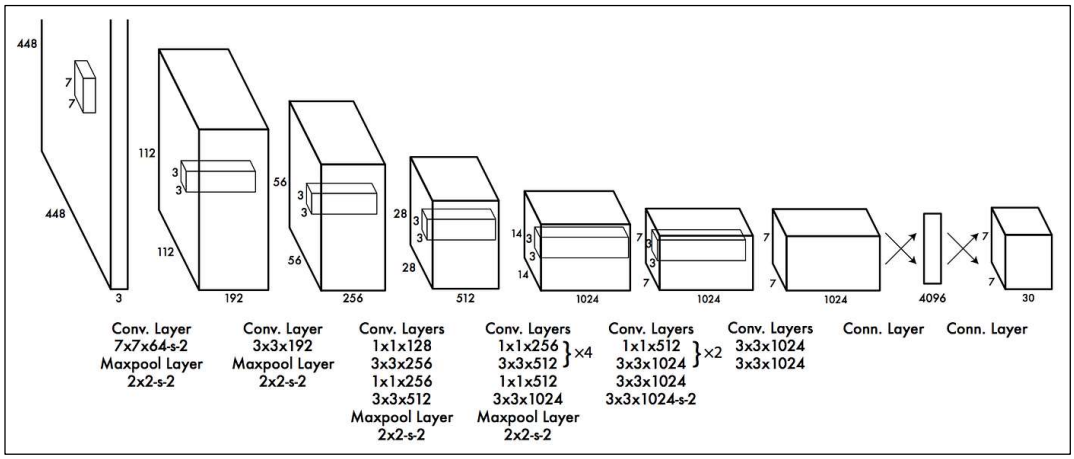

Рисунок 2.1 – Архітектура мережі YOLO, що має 24 згорткових шари (за даними [14])

Ця архітектура трохи жертвує точністю заради продуктивності в реальному часі, що робить її придатною для програм, які потребують швидкого виявлення об’єктів.

Загалом YOLO отримав широке застосування в різних програмах комп’ютерного бачення, включаючи спостереження, автономні транспортні засоби та відстеження об’єктів, завдяки своїй ефективності та ефективності в сценаріях реального часу.

RetinaNet – це найсучасніша модель виявлення об’єктів, призначена для вирішення проблем із виявленням об’єктів різного масштабу на зображеннях.

Ключем до її успіху є впровадження Focal Loss, який пом’якшує дисбаланс класів у наборах даних виявлення об’єктів.

RetinaNet включає мережу піраміди функцій (FPN) для ефективної обробки багато масштабних функцій. Вона використовує опорні блоки та двоетапний процес виявлення, що включає класифікацію та регресію.

Завдяки обчислювальній ефективності та високій точності RetinaNet є універсальною мережею, що знаходить застосування для загального виявлення об’єктів, спостереження та медичної візуалізації.

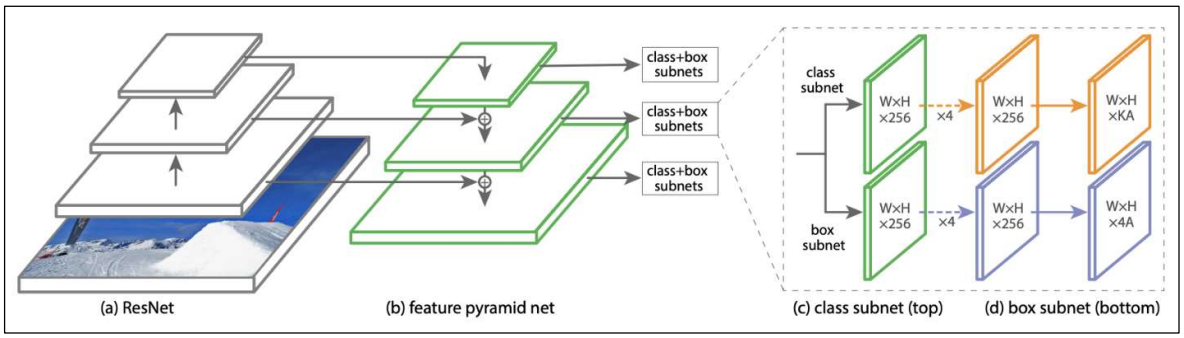

Рисунок 2.2 – Архітектура мережі RetinaNet, що використовує FPN (за даними [15])

YOLOv8 [16] CNN було обрано для виявлення листя через його високу швидкість, особливо порівняно з RetinaNet [17], і здатність працювати з відеофайлами.

2.2 Аналіз та вибір згорткових нейронних мереж для виявлення бур'янів та хвороб

Для класифікації бур'янів і хвороб ми порівняли такі архітектури, як VGG19 і MobileNet.

VGG19 є варіантом моделі VGG, архітектури згорткової нейронної мережі (CNN), представленої Visual Geometry Group з Оксфорда. VGG19 має загалом 19 шарів, у тому числі 16 шарів згортки, що дозволяє цій мережі виявляти складні особливості на зображеннях. Зокрема, послідовні згорткові шари у VGG19 можуть витягувати низько рівневі функції на початкових шарах і більш абстрактні, високо рівневі функції на глибших шарах. Це робить VGG19 особливо придатним для нашої мети виявлення тонких відмінностей у появі різних бур'янів і захворювань листя томатів [18].

MobileNet характеризується невеликим розміром і низькою затримкою, що робить його привабливим вибором для програм, які потребують швидкої обробки в режимі реального часу. Він використовує оптимізацію «стиску та збудження» та активацію жорсткого свисту для досягнення високої ефективності [19].

Обидві моделі були оцінені на основі їх обчислювальної ефективності, точності класифікації та стійкості до різноманітних і потенційно складних умов освітлення, ґрунту та погодних умов у тепличних умовах. Вибраний класифікатор повинен підтримувати високу продуктивність за всіма цими параметрами, щоб ефективно ідентифікувати бур’яни та хвороби, тим самим запобігаючи їх поширенню та забезпечуючи оптимальне здоров’я посівів.

2.3 Підготовка даних до проведення експерименту

Для виявлення листя та правильного обрізання зображення було взято 2000 зображень із спеціального набору даних із анотаціями. Він складається із зображень рослин у ґрунті. Набір даних складається з 1400 зображень поїздів і 600 зображень перевірки [20].

Для виявлення бур'янів використовувався набір даних V2 Plant Seedlings. Цей набір даних містить 5539 зображень сходів культур і бур’янів. Зображення згруповані в 12 класів, як показано на малюнках вище. Ці класи представляють поширені види рослин у сільському господарстві Данії. Кожен клас містить зображення RGB, які показують рослини на різних стадіях росту. Зображення мають різні розміри та формат PNG [21].

Для виявлення хвороб листя використовувався набір даних виявлення захворювань листя томатів. У даних є різні види хвороб листя томатів. Навчальний набір даних містить 10000 мічених зображень, а дані тестування/оцінки містять 1000 мічених зображень [22].

2.4 Вибір метрик та критеріїв порівняння

Щоб виміряти продуктивність запропонованого класифікатора HNN і конкретних моделей нейронних мереж, ми будемо використовувати кілька ключових показників і критеріїв, призначених для охоплення різних аспектів ефективності та ефективності. Критичні критерії порівняння включали точність, точність, пригадування та середню точність. Кожен із цих показників дає різний погляд на ефективність моделей, а разом вони забезпечують комплексну оцінку.

Точність є одним із найпростіших показників. Він вимірює частку правильних прогнозів, зроблених моделями. Хоча це дає загальне уявлення про продуктивність моделі, воно не розрізняє помилкові спрацьовування та помилкові негативи, що веде нас до точності та запам’ятовування.

Точність – це здатність моделі ідентифікувати лише відповідні об’єкти. Це відсоток правильних позитивних прогнозів і визначається за формулою 2.1.[23]

$$Precision = \frac{TP}{TP + FP} = \frac{TP}{AllDetections} \tag{2.1}$$

Істинно позитивний (TP): правильне виявлення. Виявлення з IOU >= порогове значення. Помилково негативний (FN): Основна істина не виявлена [23, 24].

Помилковий результат (FP): неправильне визначення. Виявлення за допомогою IOU < порогового значення. Поріг: залежно від метрики зазвичай встановлюється на 50%, 75% або 95% [23, 24].

Запам'ятовування або чутливість вимірюють частку фактичних позитивних результатів, правильно визначених. Це важливо для запобігання поширенню хвороб і бур’янів (формула 2.2).

$$Recall = \frac{TP}{TP + FN} = \frac{TP}{All\ Ground\ Truths} \tag{2.2}$$

Крива Precision x Recall використовується для оцінки продуктивності детектора об’єктів, оскільки впевненість змінюється шляхом побудови кривої для кожного класу об’єктів [23].

Середня точність (AP) – це числова метрика, яка також може допомогти нам порівняти різні детектори. На практиці AP – це точність, усереднена за всіма значеннями запам’ятовування від 0 до 1. Загальним визначенням середньої

точності є знаходження площі під кривою точності пригадування (формули 2.3 та 2.4) [25].

$$\sum_{n=0} (r_{n+1} - r_n)\rho_{int\ erp}(r_n + 1) \tag{2.3}$$

де

$$\rho_{int\ erp}(r_n + 1) = \max\ \rho(\tilde{r})_{\tilde{r}:\tilde{r}\geq r_{n+1}} \tag{2.4}$$

Це виміряна точність при відкликанні [25].

У цьому випадку замість використання точності, що спостерігається лише в кількох точках, AP тепер отримується шляхом інтерполяції точності на кожному рівні, беручи максимальну точність, значення відкликання якої більше або дорівнює. Таким чином ми обчислюємо розрахункову площу під кривою [25].

2.5 Засоби проведення дослідження

Дане магістерське дослідження буде виконане з використанням мови програмування Python, програмний код буде написаний за допомогою IDE Pycharm.

У ході проведення дослідження будуть використані деякі бібліотеки, зокрема:

TensorFlow – відкрите програмне забезпечення для машинного навчання і глибокого навчання, розроблене Google. Він надає широкий спектр інструментів для розробки та навчання моделей, включаючи гнучкі API для побудови та навчання різноманітних типів нейронних мереж. У даному магістерському дослідженні буде використана для створення, компіляції та навчання обраних мереж [26].

NumPy – бібліотека, яка надає підтримку для роботи з числовими масивами та матрицями. У даному магістерському дослідженні буде використана для спрощення та полегшення процесу розробки.

OpenCV – це відкрите програмне забезпечення для комп'ютерного зору та обробки зображень. Вона розроблена для надання інструментів для створення програм, які можуть аналізувати та оброблювати зображення та відео. У даному магістерському дослідженні буде використана для обробки вхідних зображень.

Keras – високорівневий інтерфейс для роботи з нейронними мережами, який працює поверх бібліотеки глибокого навчання TensorFlow. У даному магістерському дослідженні буде використана для полегшення та спрощення роботи з нейронними мережами [27].

Pandas – бібліотека, яка надає ефективні та легко використовувані структури даних для обробки та аналізу даних. У даному магістерському дослідженні буде використана для аналізу отриманих даних.

Roboflow Inference – це платформа з відкритим кодом, призначена для спрощення розгортання моделей комп’ютерного зору. Вона дає змогу розробникам виконувати виявлення об’єктів, класифікувати та сегментувати екземпляри, а також використовувати базові моделі, як-от CLIP, Segment Anything і YOLO-World, за допомогою нативного пакета Python, самостійного сервера або повністю керованого API [28].

2.6 Планування експерименту

У нашому дослідженні ми використовували попередньо навчену модель YOLOv8 для виявлення листя, навченого на спеціальному наборі даних [20].

Для класифікації зображень для виявлення бур’янів і хвороб ми навчили моделі VGG19 і MobileNet на наборі даних бур’янів [21] і на наборі даних хвороб томатів [22]. Обидва етапи навчання використовували попередньо підготовлені ваги ImageNet.

Після цього, використовуючи тестовий набір даних, ми виміряли точність, прецизійність, запам’ятовування та середню точність. Дані, отримані під час тренінгів, відображалися на графіках.

3 ОПИС ПРОГРАМНОЇ РЕАЛІЗАЦІЇ

У цьому розділі буде наведено реалізацію гібридного нейронного класифікатора мовою програмування Python у програмному середовищі Pycharm з використанням бібліотеки TensorFlow та інтерфейсу Keras.

Для створення гібридного класифікатора потрібно реалізувати три його компоненти:

– реалізація для визначення листя;
– реалізація для визначення потенційних бур’янів;
– реалізація для визначення потенційних захворювань.

3.1 Програмна реалізація для визначення листя

Програмна реалізація для визначення листя потрібна для більш точної роботи класифікаторів. Принцип роботи класифікаторів оснований на виявлені особливостей та подібностей у зображеннях. Таким чином мета цієї частини гібридного класифікатора у тому, щоб надати найбільш детальне зображення для аналізу.

Для реалізації було обрано попередньо навчену модель YOLOv8 з платформи Roboflow Inference з відкритим кодом, призначеної для спрощення розгортання моделей комп’ютерного зору.

Програмна реалізація визначення листя складається з двох частин.

Перша частина – це власне підключення попередньо навченої моделі та введення параметрів для її роботи. У даному випадку модель була налаштована на розпізнавання будь-якого типу рослин.

Підключення YOLOv8 моделі, її налаштування та отримання результатів виявлення об’єктів:

```
# load a pre-trained yolov8n model
model = inference.get_roboflow_model(model_id="lettuce-weed-4/1")

# run inference on chosen image
results = model.infer(
    image,
```

```
    confidence=0.1,
    iou_threshold=0.0,
)
# load the results into the supervision Detections api
detections = sv.Detections.from_inference(results[0].dict(
    by_alias=True,
    exclude_none=True
))
```

Друга частина – отримавши координати об’єктів на вихідному зображенні, потрібно вирізати ці об’єкти. Таким чином ми отримаємо набір зображень, які містять мінімальний набір зайвих деталей. Цей набір зображень буде далі переданий моделям класифікаторів для аналізу на наявність бур’янів та захворювань у рослин.

Код реалізації обрізання зображень та підготовки їх для передачі класифікаторам:

```
x1 = int(detections.xyxy[0])
y1 = int(detections.xyxy[1])
x2 = int(detections.xyxy[2])
y2 = int(detections.xyxy[3])

# create supervision annotators
bounding_box_annotator = sv.BoundingBoxAnnotator()
label_annotator = sv.LabelAnnotator()

# annotate the image with our inference results
annotated_image = bounding_box_annotator.annotate(
    scene=image, detections=detections)
annotated_image = label_annotator.annotate(
    scene=annotated_image, detections=detections)

cropped_image = img.crop((x1, y1, x2, y2)).resize((224, 224))
```

3.2 Програмна реалізація для визначення потенційних бур’янів та захворювань

Програмна реалізація класифікаторів для визначення потенційних бур’янів та захворювань буде однакова, так як підхід до навчання класифікаторів однаковий. Різниця полягає лише у вихідних наборах зображень в залежності від цілей навчання класифікатора.

Спершу потрібно створити генератор даних для навчальних зображень, завантажити всі зображення та перетворити їх до однакового розміру:

```
train_datagen = ImageDataGenerator(
    rescale=1. / 255,
    validation_split=0.2,  # set the validation split
    rotation_range=15,
    shear_range=0.2,
    zoom_range=0.2,
    width_shift_range=0.1,
    height_shift_range=0.1,
    horizontal_flip=True)

train_generator = train_datagen.flow_from_directory(
    data_dir,
    target_size=(224, 224),
    batch_size=64,
    class_mode='categorical',
    subset='training')
```

Те саме потрібно зробити і для набору зображень перевірки.

Наступний крок – це підготовка моделі нейронної мережі, яка буде використана для класифікатора. Обидва класифікатора ініціалізовані попередньо підготовленими ваговими коефіцієнтами ImageNet з подальшим налаштуванням на обраних наборах даних.

У даному прикладі наведено код моделі VGG19:

```
base_model = VGG19(
        weights='imagenet',
        include_top=False,
        input_shape=(224, 224, 3))

    x = base_model.output
    x = GlobalAveragePooling2D()(x)
    x = Dense(1024, activation='relu')(x)
    outputs = Dense(train_generator.num_classes,
activation='softmax')(x)

    model = Model(inputs=base_model.input, outputs=outputs)
```

Першим кроком до перенесення навчання є заморожування всіх шарів і навчання лише верхніх шарів:

```
for layer in model.layers[:20]:
    layer.trainable = False
```

Другим кроком є розморожування ряду шарів і підгонка моделі з меншою швидкістю навчання:

```
def freeze_unfreeze(last_layer_trainable):
    # setting first layers to be untrainable
    for layer in model.layers[:last_layer_trainable]:
        if not isinstance(layer, tf.keras.layers.BatchNormalization):
            layer.trainable = False
    # setting last layers to be trainable
    for layer in model.layers[last_layer_trainable:]:
        if not isinstance(layer, tf.keras.layers.BatchNormalization):
            layer.trainable = True
```

Останній крок – це власне навчання та збереження моделі нейронної мережі для подальшого використання у гібридному класифікаторі:

```
opt = Nadam(learning_rate=0.001)
model.compile(optimizer=opt,
    loss='categorical_crossentropy',
    metrics=['accuracy'])
model.fit(
    train_generator,
    validation_data=validation_generator,
    epochs=10,
    verbose=1)

freeze_unfreeze(last_layer_trainable=-20)

model.fit(
    train_generator,
    validation_data=validation_generator,
    epochs=5,
    verbose=1)

model.save("model.disease_detection_VGG19.keras")
```

3.3 Програмна реалізація гібридного класифікатора

Програмна реалізація гібридного класифікатора полягає у використанні попередньо навчених моделей нейронних мереж для виконання відповідних задач. У даному випадку гібридний класифікатор поєднує у собі модель YOLOv8 для визначення листя на зображенні та моделі VGG19 або MobileNet для класифікації зображень за присутністю бур’янів та наявності захворювань у рослин.

Програмна реалізація визначення листя та обрізання зображень описана у пункті 3.1.

Реалізація гібридного класифікатора:

```
cropped_image = img.crop((x1, y1, x2, y2)).resize((224, 224))
disease_model_VGG19 = keras.saving.load_model(
      "model.disease_detection_VGG19.keras",
      compile=True)

weed_model_VGG19 = keras.saving.load_model(
      "model.weeds_detection_VGG19.keras",
      compile=True)

img_cropped = tf.keras.preprocessing.image.load_img(
      image_file,
      target_size=(224, 224))
img_cropped = tf.keras.preprocessing.image.img_to_array(img_cropped)
img_cropped = np.expand_dims(img_cropped, axis=0)
img_cropped = img_cropped / 255.

disease_prediction_VGG19 = disease_model_VGG19.predict(img_cropped)
disease_class_VGG19 = disease_classes[np.argmax(
      disease_prediction_VGG19)]
weed_prediction_VGG19 = weed_model_VGG19.predict(img_cropped)
weed_class_VGG19 = weed_classes[np.argmax(weed_ prediction_VGG19)]
```

3.4 Архітектурна реалізація на основі хостингової платформи AWS

Для архітектури прикладного рішення було використано стиль мікросервісів, який забезпечує гнучкість, зручність обслуговування, масштабованість і можливість ефективного розміщення в хмарі. В якості хостингової платформи було обрано AWS Cloud. Він забезпечує високодоступні та масштабовані економічно ефективні рішення для задоволення очікуваних вимог. Для досягнення високої продуктивності, а також для виконання вимог щодо доступності та аварійного відновлення було вибрано мультирегіональне налаштування.

Прикладне рішення складається з кількох взаємопов’язаних мікросервісів, які працюють разом, щоб забезпечити ефективне завантаження та аналіз фотографій. Кожен мікросервіс відповідає за певний набір функцій і може бути розроблений, розгорнутий і масштабований незалежно.

Архітектура включає такі ключові компоненти:

– інтерфейс зовнішнього користувача, що дозволяє користувачам взаємодіяти з системою. Він надає функції для реєстрації користувачів, входу та автентифікації;

- служба керування користувачами, що керує обліковими записами користувачів, автентифікацією та авторизацією. Це гарантує, що лише авторизовані користувачі можуть отримати доступ до системи та виконувати певні дії;
- служба аналізу фотографій, що відповідає за аналіз завантажених фотографій. Ця служба використовує алгоритми гібрибного нейромережевого класифікатора;
- служба імпорту/експорту діє як адаптер для інтеграції із зовнішніми службами зберігання, такими як Google Drive або інші хмарні платформи зберігання;
- служба сповіщень надсилає повідомлення користувачам за допомогою різних каналів зв’язку.

Зв’язок між мікросервісами виконаний через API або асинхронну передачу повідомлень, що забезпечує слабкий зв’язок і забезпечує незалежну розробку, розгортання та масштабованість кожного компонента. Архітектура цільового рішення використовує сучасні технології та фреймворки, придатні для кожного мікросервісу, наприклад Node.js, Express.js, Python, TensorFlow і MongoDB (див. рис.3.1-3.2).

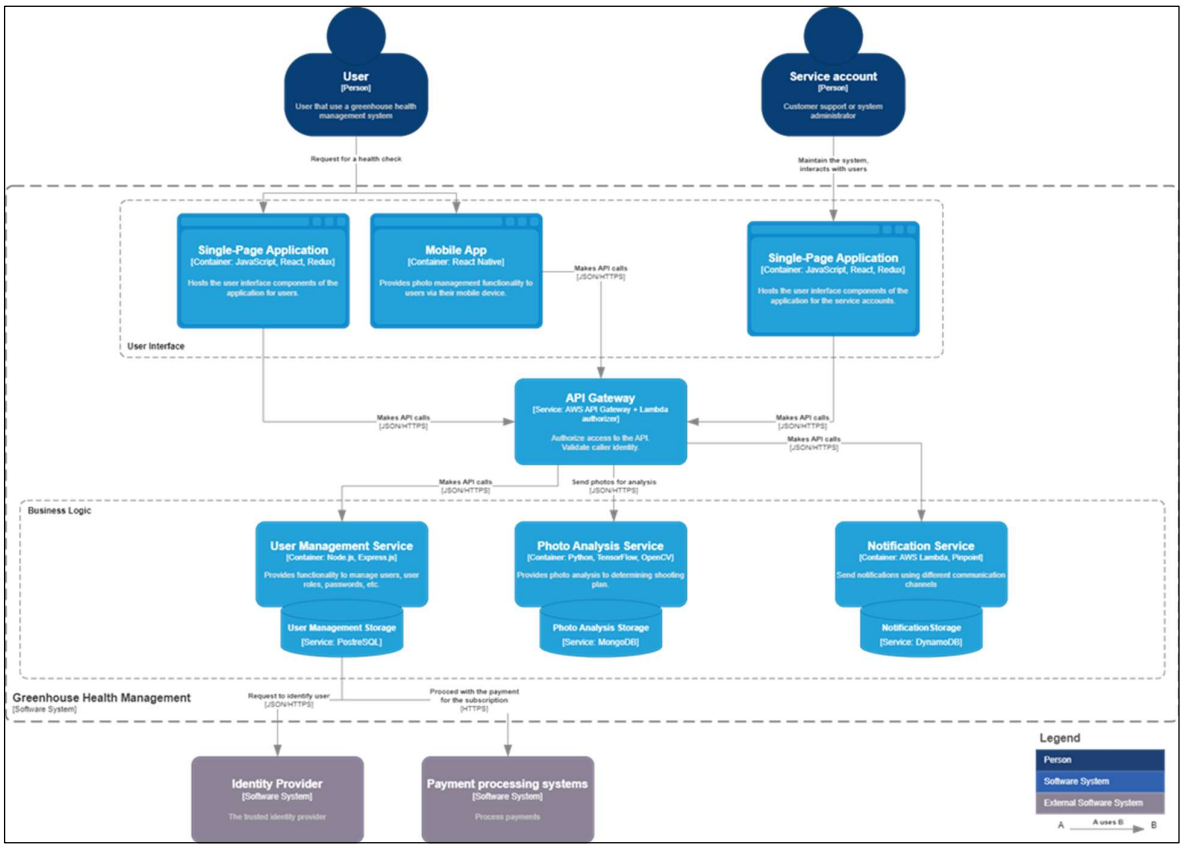

Рисунок 3.1 – Діаграма логічних компонентів запропонованого рішення на основі хостингової платформи AWS (рисунок створено самостійно)

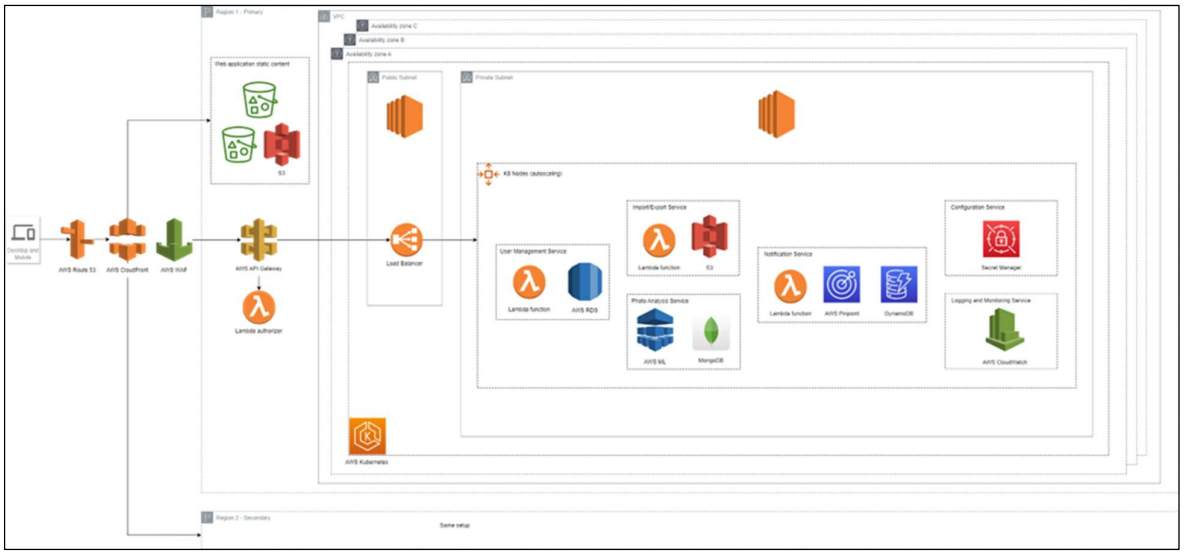

Рисунок 3.2 – Діаграма розгортання компонентів запропонованого рішення на основі хостингової платформи AWS (рисунок створено самостійно)

Діаграми розгортання включає в себе такі компоненти:

- AWS Route 53 надсилає користувача до найближчого доступного регіону;
- AWS CloudFront служить CDN для статичного вмісту, розміщеного в S3, і надсилає запити API до шлюзу API відповідного регіону;
- AWS WAF захищає програми від поширених веб-експлойтів і ботів, які можуть впливати на доступність, порушувати безпеку або споживати надмірні ресурси;
- AWS API із Lambda Authorizer перевіряє особу абонента та пересилає запити до відповідної серверної служби/функції лямбда;
- AWS Kubernetes (EKS) автоматизує розгортання, масштабування та керування контейнерними серверними програмами;
- AWS Application Load Balancer (ALB) інтегровано з Kubernetes. Він балансує навантаження вхідного трафіку на прикладному рівні між кількома цілями;
- AWS Lambda це безсерверний обчислювальний сервіс, керований подіями, який дозволяє запускати код без підготовки або керування серверами;
- AWS RDS Aurora забезпечує вбудовану безпеку, безперервне резервне копіювання, безсерверне обчислення, автоматизовану реплікацію в кількох регіонах та інтеграцію з іншими службами AWS;
- AWS DynamoDB безсерверна база даних NoSQL із ключем-значенням, розроблена для запуску високопродуктивних програм у будь-якому масштабі;
- MongoDB Atlas керована та гнучка база даних на AWS;
- AWS S3 використовується в усіх місцях, де потрібне зберігання об’єктів: статичні файли додатків інтерфейсу користувача, зберігання згенерованих звітів тощо;
- AWS Pinpoint це гнучкий і масштабований сервіс маркетингових комунікацій для вихідних і вхідних каналів. Він може зв’язуватися з

клієнтами за допомогою електронних листів, SMS, push-повідомлень, голосових служб або обміну повідомленнями в програмі;

- AWS Secrets Manager допомагає керувати конфігурацією програми, включаючи облікові дані бази даних, ключі API та інші секрети;
- AWS CloudWatch надає дані та практичну інформацію для моніторингу додатків, збирає моніторингові та робочі дані у вигляді журналів, показників і подій;
- AWS ML надає всі інструменти та бібліотеки для створення моделей ML.

Ця архітектура забезпечує необхідну гнучкість, масштабованість і модульність для керування великою кількістю користувачів і фотографій. Ця стратегія мінімізує час простою під час розгортання або оновлення. Цей підхід передбачає розгортання останньої версії системи паралельно з наявною версією та поступове спрямування трафіку до нової версії, щоб забезпечити плавні переходи та варіанти відкату, якщо це необхідно.

4 АНАЛІЗ РЕЗУЛЬТАТІВ ДОСЛІДЖЕННЯ

4.1 Результати розпізнавання листя

Модель YOLOv8 була попередньо навчена виявляти листя та виконувати належне кадрування зображення, 2000 зображень було відібрано з спеціального набору даних із анотаціями до зображення.

Як ми бачимо на кривій навчання, навчання нейронної мережі закінчено і подальше навчання не має сенсу. Нейронна мережа готова до використання (див.рис.4.1-4.3).

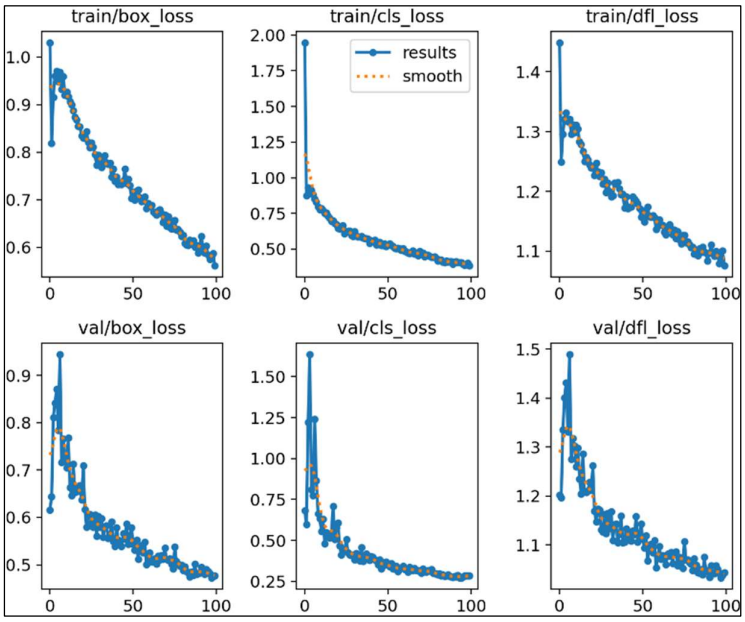

Рисунок 4.1 – Криві навчання виявлення листя [20]

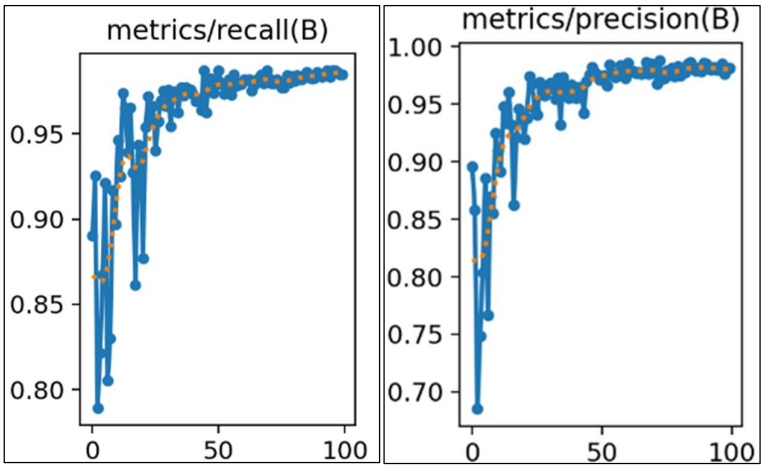

Рисунок 4.2 – Криві recall та precision YOLOv8 [20]

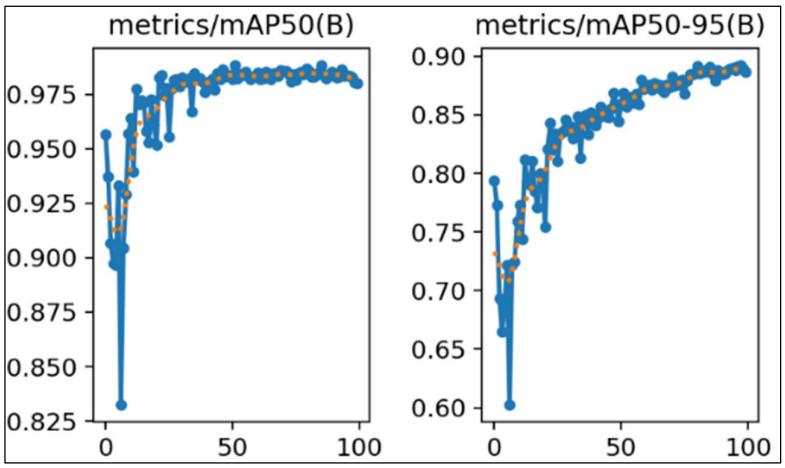

Рисунок 4.3 – Крива середньої точності YOLOv8 [20]

YOLOv8 показав високі показники швидкості обробки зображень, а також точність визначення листя (див.рис.4.4).

Рисунок 4.4 – Виявлення листя YOLOv8 (рисунок створено самостійно [29])

4.2 Результати класифікації бур'янів

Моделі VGG19 і MobileNet були навчені для класифікації бур’янів на наборі даних V2 Plant Seedlings [21]. Цей набір даних містить 5539 зображень сходів культур і бур’янів. Ми використовували попередньо навчені ваги ImageNet. Навчання проводилось у два етапи.

Першим кроком до перенесення навчання є заморожування всіх шарів і навчання лише верхніх шарів. Моделі тренувалися на десяти епохах.

Час навчання для моделі VGG19 займав у середньому 810 секунд на епоху із загальним часом навчання 3 години для першого кроку.

Час навчання для моделі MobileNet займав у середньому 200 секунд на епоху із загальним часом навчання 30 хвилин для першого кроку (див.рис.4.5-4.6).

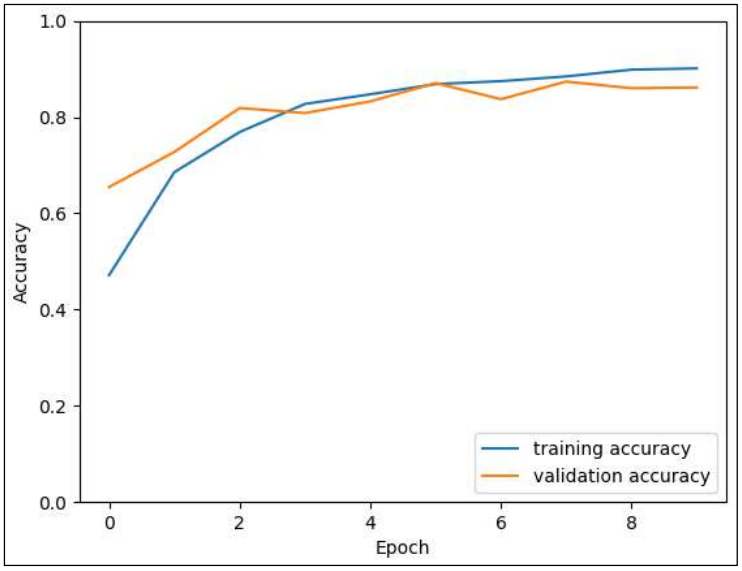

Рисунок 4.5 – Крива точності VGG19 для першого кроку навчання класифікації бур'янів (рисунок створено самостійно [30])

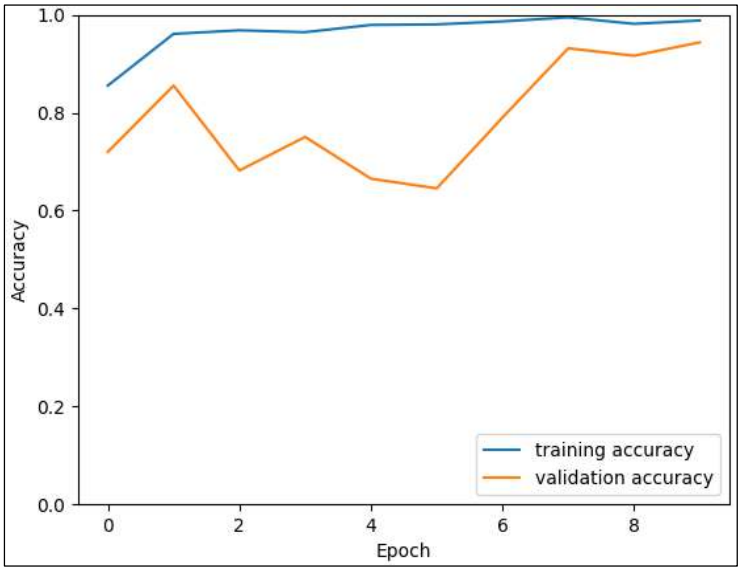

Рисунок 4.6 – Крива точності MobileNet для першого кроку навчання класифікації бур'янів (рисунок створено самостійно [32])

Другим кроком є розморожування 20 верхніх шарів і підгонка моделі з меншою швидкістю навчання. Моделі тренувалися на п'яти епохах.

Час навчання для моделі VGG19 займав у середньому 2050 секунд на епоху із загальним часом навчання 3 години для другого етапу.

Час навчання для моделі MobileNet займав у середньому 200 секунд на епоху із загальним часом навчання 30 хвилин для другого етапу (див.рис.4.7-4.8).

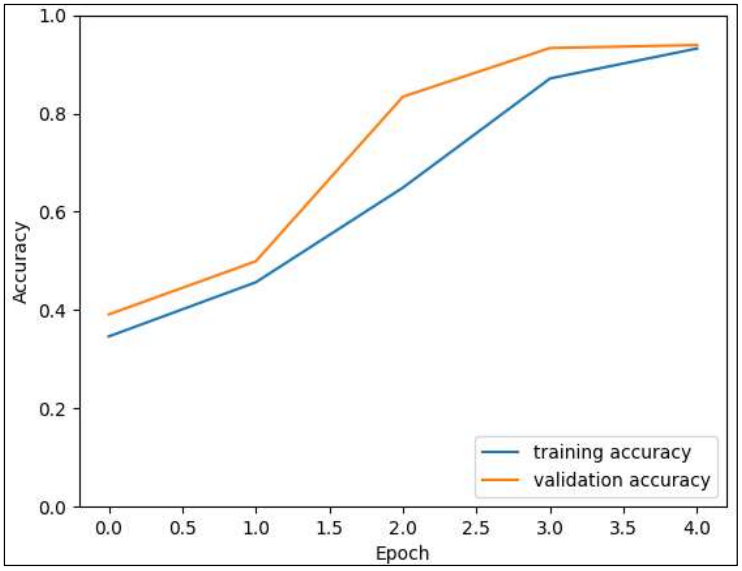

Рисунок 4.7 – Крива точності VGG19 для другого кроку навчання класифікації бур’янів (рисунок створено самостійно [31])

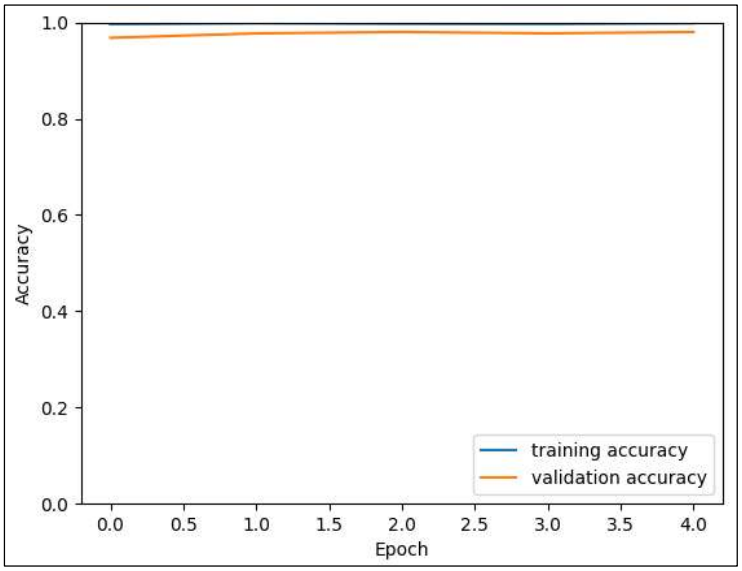

Рисунок 4.8 – Крива точності MobileNet для другого кроку навчання класифікації бур’янів (рисунок створено самостійно [33])

Проаналізувавши результати навчання, ми дійшли висновку, що обидві моделі мають високі показники точності класифікації бур’янів.

Однак після експерименту з фотографією, яка містить і паросток бур'яну, і паросток помідора, стає очевидним, що модель MobileNet дає хибнопозитивні результати. Ця проблема може призвести до неправильної загальної роботи класифікатора HNN (див.рис.4.9).

Рисунок 4.9 – Паросток помідора, обрізаний з оригінального фото (рисунок створено самостійно [38])

Keras 3 надає такі дані про моделі: MobileNet має 4,3 млн параметрів, а VGG19 – 143,7 млн параметрів [41].

Такий результат можна вважати очікуваним через різницю в кількості параметрів моделі (див.табл.4.1).

Таблиця 4.1 – Хибно позитивні результати класифікації бур’янів

Neural Network	**Number of Parameters**	**Accuracy, %**	**Weed Type**
VGG19	143.7M	37.32%	Black grass
MobileNet	4.3M	99.99%	Black grass

4.3 Результати класифікації захворювання

Моделі VGG19 і MobileNet були навчені для класифікації хвороб томатів на основі набору даних виявлення хвороб листя томатів [22]. Цей набір даних містить 10 000 зображень із мітками, а дані тестування та оцінки містять 1000 зображень із мітками. Ми використовували попередньо навчені ваги ImageNet.

Першим кроком до перенесення навчання є заморожування всіх шарів і навчання лише верхніх шарів. Моделі тренувалися на десяти епохах.

Час навчання для моделі VGG19 займав у середньому 4700 секунд за епоху із загальним часом навчання 13 годин для першого кроку.

Час навчання для моделі MobileNet займав у середньому 1100 секунд за епоху із загальним часом навчання 3 години для першого кроку (див.рис.4.10-.411).

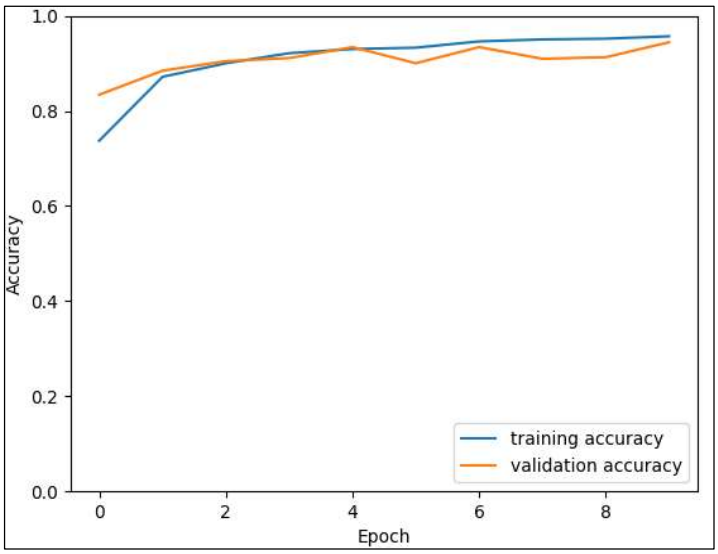

Рисунок 4.10 – Крива точності VGG19 для першого кроку навчання класифікації захворювань (рисунок створено самостійно [34])

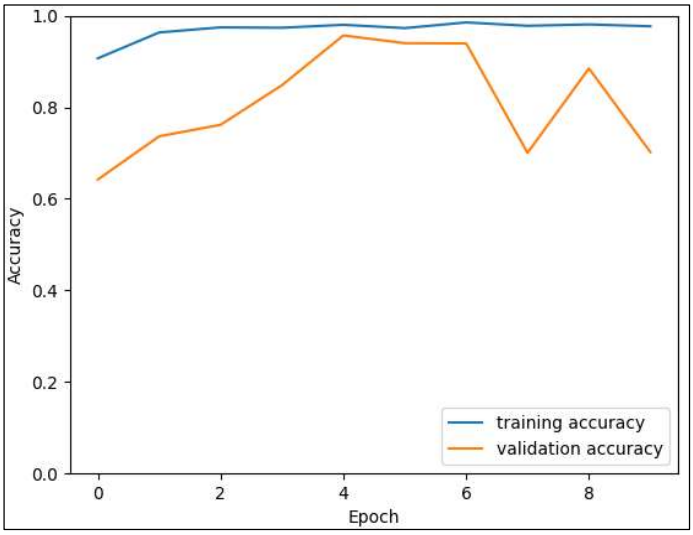

Рисунок 4.11 – Крива точності MobileNet для першого кроку навчання класифікації захворювання (рисунок створено самостійно [36])

Другим кроком є розморожування 20 верхніх шарів і підгонка моделі з меншою швидкістю навчання. Моделі тренувалися на п'яти епохах.

Час навчання для моделі VGG19 займав у середньому 12000 секунд на епоху із загальним часом навчання 17 годин для другого етапу.

Час навчання для моделі MobileNet займав у середньому 810 секунд за епоху із загальним часом навчання 1 година для другого етапу (див.рис.4.12-4.13).

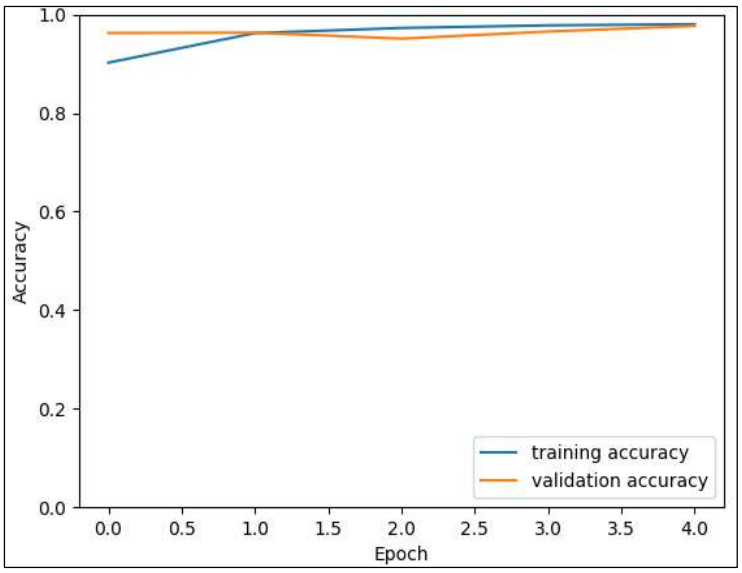

Рисунок 4.12 – Крива точності VGG19 для другого кроку навчання класифікації захворювання (рисунок створено самостійно [35])

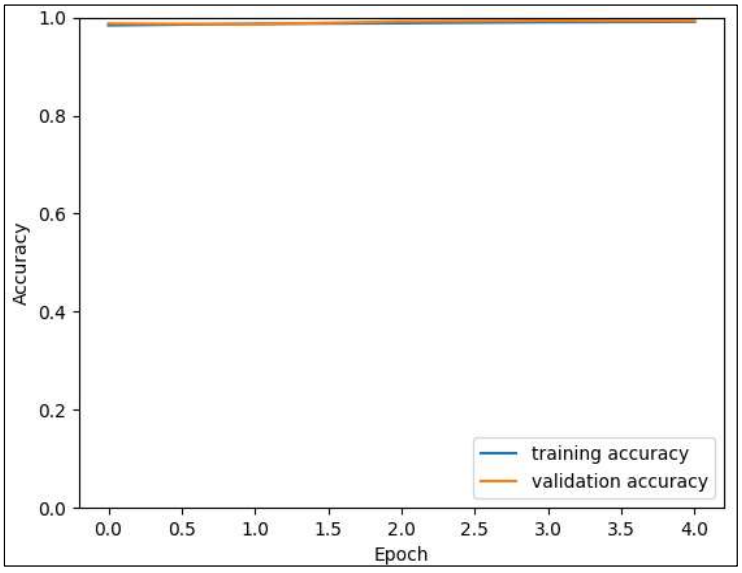

Рисунок 4.13 – Крива точності MobileNet для другого кроку навчання класифікації захворювань (рисунок створено самостійно [37])

Проаналізувавши результати навчання, можна побачити, що модель VGG19 має вищу точність класифікації захворювань.

Цей висновок підтверджено результатом експерименту з класифікації листя помідора, ураженого хворобою листової плісняви (див.рис.4.14 та табл.4.2).

Рисунок 4.14 – Лист помідора з хворобою листової плісняви (рисунок створено самостійно [39])

Таблиця 4.2 – Результати класифікації хвороб

Neural Network	Number of Parameters	Accuracy, %	Weed Type
VGG19	143.7M	99.99%	Leaf mold
MobileNet	4.3M	53.02%	Leaf mold

4.4 Алгоритм застосування класифікатора

Враховуючи результати нашого дослідження, ми можемо вивести інструкцію щодо ефективного використання класифікатора HNN. Ми пропонуємо покроковий алгоритм використання цієї технології для ефективного виявлення листя, класифікації бур'янів і ідентифікації хвороб.

Мета полягає в тому, щоб надати вичерпний посібник, яким можна керуватися, щоб використовувати класифікатор HNN для діагностики здоров'я рослин і подальшого прийняття рішень.

Цей процес включає кілька етапів, включаючи отримання зображень, попередню обробку, виявлення листя за допомогою моделі YOLOv8,

класифікацію бур’янів і хвороб за допомогою моделі VGG19 і, нарешті, об’єднання цих результатів для прийняття обґрунтованих рішень щодо управління здоров’ям рослин. Алгоритм також підкреслює важливість постійного вдосконалення моделі для підвищення точності та продуктивності.

Запропонований алгоритм застосування класифікатора HNN (див.рис.4.14) складається з наступних етапів:

а) виявлення листя:

1) отримайте зображення листя рослини або ураженої ділянки за допомогою камери або інших пристроїв обробки зображень;

2) попередньо обробіть отримане зображення для підвищення якості та стандартизуйте його формат для введення в моделі нейронної мережі. Це може включати зміну розміру, нормалізацію та інші відповідні методи для забезпечення узгодженості вхідних даних;

3) використовуйте попередньо навчену модель YOLOv8 для виявлення листя. Модель автоматично визначає та вирізає з вхідного зображення області, що містять листя, полегшуючи подальший аналіз.

б) класифікація бур’янів:

1) використовуйте модель VGG19, навчену для класифікації бур’янів. Передайте зображення обрізаного листя в моделі. Модель VGG19 особливо підходить для детального вилучення функцій;

2) оцініть результати класифікації, щоб визначити присутність бур’янів.

в) класифікація захворювання:

1) використовуйте модель VGG19, навчену для класифікації захворювань;

2) виявляйте та класифікуйте будь-які захворювання, присутні на листках, завантажуючи обрізані зображення в моделі.

г) остаточне рішення:

1) об’єднайте результати класифікації бур’янів і хвороб, щоб приймати обґрунтовані рішення щодо здоров’я рослин і стратегій управління;

2) Постійно оновлюйте та вдосконалюйте моделі на основі нових даних і відгуків від реальних додатків, щоб з часом підвищувати їх точність і продуктивність.

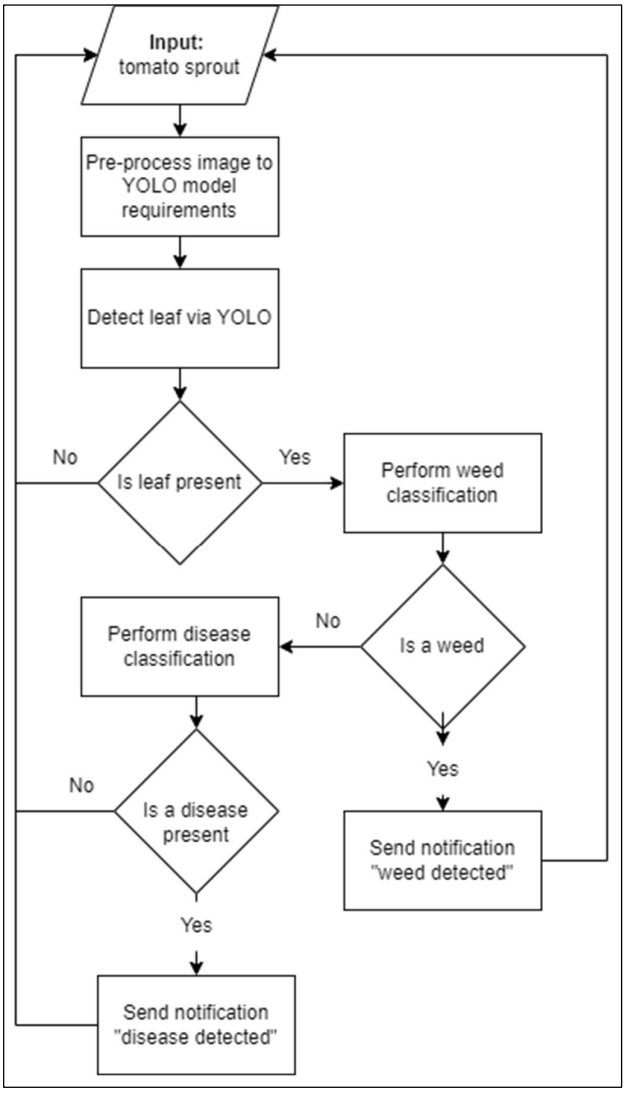

Рисунок 4.14 – Запропонований алгоритм застосування гібридного нейромережевого класифікатора (рисунок створено самостійно [40])

ВИСНОВКИ

За результатами нашого дослідження можна зробити наступні основні висновки.

У нашому дослідженні ми експериментально проаналізували найсучасніші моделі нейронних мереж для вирішення таких завдань, як виявлення листя, класифікація бур’янів і класифікація хвороб томатів.

Для виявлення листя ми використовували спеціальний набір даних і попередньо навчену модель YOLOv8 [20].

Ми використали набір даних про бур’яни [6] для класифікації бур’янів і набір даних про хвороби томатів [22] для класифікації хвороб томатів.

Моделі VGG19 і MobileNet були обрані як кандидати для класифікаторів бур'янів. Моделі тренувалися за допомогою попередньо підготовлених ваг ImageNet.

Після тренувань ми порівняли точність моделей VGG19 і MobileNet для завдань класифікації бур’янів і хвороб томатів.

Модель VGG19 показала високу точність у задачах прогнозування бур’янів і класифікації захворювань. Середній показник для моделі VGG19 у завданні класифікації захворювань становив 99,99%.

Модель VGG19 показала низький відсоток помилкових істинних результатів у задачі прогнозування бур’янів – 37,32% для тестового набору зображень.

Недоліком моделі VGG19 є час навчання: 33 години для повністю навченої моделі класифікації бур’янів плюс 147 годин для повністю навченої моделі класифікації хвороб.

З іншого боку, модель MobileNet пропонує швидший час навчання – 5,5 годин для повністю навченої моделі класифікації бур’янів плюс 31 година для повністю навченої моделі класифікації хвороб.

Модель MobileNet може бути більш придатною для середовища з обмеженими ресурсами через менший розмір програми – 16 МБ [41] порівняно з 549 МБ [41] для програми VGG19.

Недоліком моделі MobileNet є високий рівень помилкових істинних результатів у завданні класифікації бур’янів – 99,99% для тестового набору зображень. Також модель MobileNet показала недостатню точність у задачах виявлення захворювань – в середньому 53,02% проти 99,99% для VGG19.

Оскільки кожна модель має свої переваги та недоліки, залежно від умов використання, ми повинні використовувати відповідну модель. Використовуйте модель VGG19, коли немає обмежень у ресурсах. Для випадків середовища з обмеженими ресурсами можна використовувати модель MobileNet.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. H. I. Peyal et al., "Plant Disease Classifier: Detection of Dual-Crop Diseases Using Lightweight 2D CNN Architecture," in IEEE Access, vol. 11, pp. 110627-110643, 2023, doi: 10.1109/ACCESS.2023.3320686. (https://ieeexplore.ieee.org/document/10267978)

2. E. Özbılge, M. K. Ulukök, Ö. Toygar and E. Ozbılge, "Tomato Disease Recognition Using a Compact Convolutional Neural Network," in IEEE Access, vol. 10, pp. 77213-77224, 2022, doi: 10.1109/ACCESS.2022.3192428. (https://ieeexplore.ieee.org/document/9832884)

3. Geisenberg, C. and K. Stewart (1986), “Field crop management”, in Atherton, J.G. and J. Rudich (eds.), The Tomato Crop: A Scientific Basis for Improvement, Chapman & Hall, London, pp. 511-557.

4. USDA 2019, "Imported Greenhouse Tomatoes From Mexico Illustrate the Growing Diversity in Fresh-Market Tomatoes". (https://www.ers.usda.gov/data-products/chart-gallery/gallery/chart-detail/?chartId=93021)

5. C. Zhou, S. Zhou, J. Xing and J. Song, "Tomato Leaf Disease Identification by Restructured Deep Residual Dense Network," in IEEE Access, vol. 9, pp. 28822-28831, 2021, doi: 10.1109/ACCESS.2021.3058947. (https://ieeexplore.ieee.org/document/9353592)

6. Farm Credit Canada (2021), “Despite remarkable greenhouse sales in 2020 – uncertainty lies in the year ahead”, Canada. (https://www.fcc-fac.ca/en/knowledge/economics/greenhouse-sales)

7. P. D. Rosero-Montalvo, C. A. Gordillo-Gordillo and W. Hernandez, "Smart Farming Robot for Detecting Environmental Conditions in a Greenhouse," in IEEE Access, vol. 11, pp. 57843-57853, 2023, doi: 10.1109/ACCESS.2023.3283986. (https://ieeexplore.ieee.org/document/10146280)

8. European Commission 2022, "Sustainable use of pesticides". (https://food.ec.europa.eu/plants/pesticides/sustainable-use-pesticides_en)

9. Technology Industry 4.0, Artificial Intelligence, AI. URL: https://it-enterprise.com/knowledge-base/technology-innovation/artificial-intelligence

10. Technology Industry 4.0, Neural networks and deep learning. URL: https://it-enterprise.com/knowledge-base/technology-innovation/machine-learning

11. Using AI Methods to Evaluate a Minimal Model for Perception - Scientific Figure on ResearchGate. Available from: https://www.researchgate.net/figure/A-simple-ANN-featuring-two-hidden-layers-of-neurons-adapted-from-Nielsen-Neural_fig1_337052393 [accessed 6 Jun, 2024]

12. Types of Neural Networks and Definition of Neural Network. URL: https://www.mygreatlearning.com/blog/types-of-neural-networks/

13. What are convolutional neural networks? URL: https://www.ibm.com/topics/convolutional-neural-networks

14. YOLOv2 for Pigs Detection in Industrial Farming - Scientific Figure on ResearchGate. Available from: https://www.researchgate.net/figure/Figure-courtesy-38-The-network-consist-of-24-convolutional-layers-followed-by-2-fully_fig5_344645063 [accessed 7 Jun, 2024]

15. Traffic Sign Detection and Recognition for Autonomous Driving in Virtual Simulation Environment - Scientific Figure on ResearchGate. Available from: https://www.researchgate.net/figure/RetinaNet-network-architecture-Lin-et-al-2017_fig2_337241380 [accessed 7 Jun, 2024]

16. YOLOv8 (https://blog.roboflow.com/whats-new-in-yolov8/)

17. RetinaNet (https://paperswithcode.com/method/retinanet)

18. Very Deep Convolutional Networks for Large-Scale Image Recognition (https://paperswithcode.com/paper/very-deep-convolutional-networks-for-large)

19. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications (https://arxiv.org/pdf/1704.04861.pdf)

20. Leaf detection dataset (https://universe.roboflow.com/project-kuuew/lettuce-weed-4/dataset/1).

21. V2 Plant Seedlings Dataset. (https://www.kaggle.com/datasets/vbookshelf/v2-plant-seedlings-dataset?resource=download)

22. Tomato leaf disease detection. (https://www.kaggle.com/datasets/kaustubhb999/tomatoleaf/data)

23. K. Smelyakov, A. Chupryna, O. Bohomolov and N. Hunko, "The Neural Network Models Effectiveness for Face Detection and Face Recognition," 2021 IEEE Open Conference of Electrical, Electronic and Information Sciences (eStream), Vilnius, Lithuania, 2021, pp. 1-7, doi: 10.1109/eStream53087.2021.9431476. (https://ieeexplore.ieee.org/document/9431476)

24. Object Detection Metrics: (https://github.com/rafaelpadilla/Object-Detection-Metrics)

25. K. Smelyakov, D. Tovchyrechko, I. Ruban, A. Chupryna and O. Ponomarenko, "Local Feature Detectors Performance Analysis on Digital Image," 2019 IEEE International Scientific-Practical Conference Problems of Infocommunications, Science and Technology (PIC S&T), Kyiv, Ukraine, 2019, pp. 644-648, doi: 10.1109/PICST47496.2019.9061331. (https://ieeexplore.ieee.org/document/9061331)

26. TensorFlow (https://www.tensorflow.org/learn)

27. Keras (https://keras.io/about/)

28. Roboflow Inference (https://inference.roboflow.com/quickstart/what_is_inference/)

29. Picture YOLOv8 leaf detection result. URL: https://drive.google.com/file/d/1d3qM_llqgXXaTBoBGBpIKaIGVCwmCq3X/view?usp=drive_link

30. Picture VGG19 weed detection training first step. URL: https://drive.google.com/file/d/1NmfrdO0Lo4sUb_hci-1h4wwsVL5yswY_/view?usp=drive_link

31. Picture VGG19 weed classification training second step. URL: https://drive.google.com/file/d/1KG0JWKqaWvdd7HhafXmEMsYMEL_YWP37/view?usp=drive_link

32. Picture MobileNet weed classification training first step. URL: https://drive.google.com/file/d/1YHZVU-VglHS5ZakTyx2DF_KaOhGi8huA/view?usp=drive_link

33. Picture MobileNet weed classification training second step. URL: https://drive.google.com/file/d/1nIS6yeCgLqSlx8PT0roKcqWpiZR4UfTJ/view?usp=dri

ve_link

34. Picture VGG19 disease classification training first step. URL: https://drive.google.com/file/d/1BKCYD2YfupBTd4qaqd3kE8O1cxLNwBdm/view?usp=drive_link

35. Picture VGG19 disease classification training second step. URL: https://drive.google.com/file/d/1UhZ78EbFLkHBblXvbbtMe4lp2RgKtL1s/view?usp=drive_link

36. Picture MobileNet disease classification training first step. URL: https://drive.google.com/file/d/1ecgC3Pjo451qvhTapvflMBmvYJANGrJp/view?usp=drive_link

37. Picture MobileNet disease classification training second step. URL: https://drive.google.com/file/d/11oNXxpyyeJnfYEXGmwxeQRNcjYt-SR_Q/view?usp=drive_link

38. Picture tomato sprout after leaf detection. URL: https://drive.google.com/file/d/1BykvAEVQ7OUgtb-Wt9H1k8qL8o3Ys498/view?usp=drive_link

39. Picture leaf mold sample. URL: https://drive.google.com/file/d/1C-D7ojheCucOhtCD-mrIerAlB1EPncEP/view?usp=drive_link

40. Application algorithm. URL: https://drive.google.com/file/d/19rYMSDWoatXSGER-WVUf8SUDm3YQZhVh/view?usp=sharing

41. Keras Aplication characteristics. URL: https://keras.io/api/applications/