МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Західноукраїнський національний університет
Факультет комп'ютерних інформаційних технологій
Кафедра інформаційно-обчислювальних систем і управління

ВОРОНЯК Богдан Петрович

Програмний модуль прогнозування хвороб сільськогосподарських культур на основі згорткових нейронних мереж / Software Module for Predicting Diseases of Agricultural Crops Based on Convolutional Neural Networks

Спеціальність 122 – Комп'ютерні науки
Освітньо-професійна програма – Штучний інтелект

Кваліфікаційна робота

Виконав студент групи КНШІ-41
Б. П. Вороняк

Науковий керівник:
д.т.н., професор, М. П. Комар

Кваліфікаційну роботу допущено
до захисту
«___»_____ 2026 р.

Завідувач кафедри
_____ Н.В. Дзюбановська

Тернопіль – 2026

Факультет комп'ютерних інформаційних технологій
Кафедра інформаційно-обчислювальних систем і управління
Освітній ступінь «бакалавр»
спеціальність: 122 – Комп'ютерні науки
освітньо-професійна програма – Штучний інтелект

ЗАТВЕРДЖУЮ
В.о. завідувача кафедри
_____ Н.В. Дзюбановська
«____» _____20__ р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ
ВОРОНЯКУ Богдану Петровичу
(прізвище, ім'я, по батькові)

1. Тема кваліфікаційної роботи
Програмний модуль прогнозування хвороб сільськогосподарських культур на основі згорткових нейронних мереж / Software Module for Predicting Diseases of Agricultural Crops Based on Convolutional Neural Networks
керівник роботи д.т.н., професор М. П. Комар
затверджені наказом по університету від 01 грудня 2025 року № 894.
2. Строк подання студентом закінченої кваліфікаційної роботи 25 травня 2026 р.
3. Вихідні дані до кваліфікаційної роботи: завдання на кваліфікаційну роботу студента, наукові статті, технічна література.
4. Основні питання, які потрібно розробити
– проаналізувати особливості задачі прогнозування хвороб рослин та специфіку вхідних даних;
– виконати огляд і порівняльний аналіз існуючих методів і архітектур згорткових нейронних мереж;
– спроєктувати архітектуру програмного модуля прогнозування;
– реалізувати базову згорткову нейронну мережу та модель на основі перенесення навчання у межах єдиного програмного рішення;
– організувати процес навчання моделей, зокрема у режимі fine-tuning;
– провести експериментальне дослідження ефективності реалізованих моделей з використанням стандартних метрик оцінювання;
– виконати аналіз отриманих результатів та сформулювати висновки щодо доцільності використання обраних підходів.
5. Перелік графічного матеріалу в роботі
– структурна схема програмного модуля прогнозування хвороб сільськогосподарських культур;
– узагальнений алгоритм програмного модуля прогнозування хвороб сільськогосподарських культур;
– архітектура згорткової нейронної мережі.

6. Консультанти розділів кваліфікаційної роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		Завдання видав	Завдання прийняв

7. Дата видачі завдання 01 грудня 2025 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів кваліфікаційної роботи	Примітка
1	Затвердження теми кваліфікаційної роботи, ознайомлення з літературними джерелами та складання плану роботи	до 01.01. 2026 р.	
2	Написання 1 розділу кваліфікаційної роботи	до 01.02. 2026 р.	
3	Написання 2 розділу кваліфікаційної роботи	до 15.03.2026 р.	
4	Написання 3 розділу кваліфікаційної роботи	до 01.05. 2026 р.	
5	Представлення попереднього варіанту кваліфікаційної роботи, перевірка та внесення змін керівником	до 15.05.2026 р.	
6	Опрацювання зауважень та представлення завершеного варіанту кваліфікаційної роботи. Підготовка супроводжуючих документів	до 25.05.2026 р.	
7	Перевірка кваліфікаційної роботи на оригінальність тексту	до 30.05.2026 р.	
8	Оформлення кваліфікаційної роботи та отримання допуску до захисту	до 10.06.2026 р.	
9	Подання кваліфікаційної роботи до захисту на засіданні атестаційної комісії	до 10.06. 2026 р.	

Студент_____ Б. П. Вороняк
підпис

Керівник роботи_____ д.т.н., професор М. П. Комар
підпис

АНОТАЦІЯ

Пояснювальна записка до кваліфікаційної роботи: 71 с., 12 рис., 8 табл., 4 додатки, 35 джерел.

Об’єктом дослідження є процес автоматизованої діагностики хвороб рослин за цифровими зображеннями.

Метою кваліфікаційної роботи є розробка програмного модуля прогнозування хвороб сільськогосподарських культур на основі згорткових нейронних мереж, який забезпечує автоматизований аналіз зображень рослин і формування прогнозу щодо їхнього стану з використанням сучасних методів глибокого навчання.

Методи дослідження включають аналіз наукових джерел у сфері комп’ютерного зору, глибокого навчання, згорткових нейронних мереж і прогнозування хвороб рослин, моделювання архітектури програмного модуля, розроблення базової CNN-моделі для класифікації зображень рослин.

Результати дослідження полягають у створенні програмного модуля прогнозування хвороб сільськогосподарських культур, який забезпечує завантаження зображень рослин, їх попередню обробку, масштабування та нормалізацію, передавання даних у згорткову нейронну мережу, визначення класу стану рослини, формування ймовірнісної оцінки прогнозу та подання результатів у зручному для користувача вигляді.

Результати роботи можуть бути використані для розроблення інтелектуальних систем моніторингу стану сільськогосподарських культур, автоматизованої діагностики хвороб рослин, програмних рішень для підтримки прийняття рішень в аграрному виробництві, а також подальших досліджень у сфері комп’ютерного зору та глибокого навчання.

Ключові слова: СІЛЬСЬКОГОСПОДАРСЬКІ КУЛЬТУРИ, ХВОРОБИ РОСЛИН, ПРОГНОЗУВАННЯ, КОМП’ЮТЕРНИЙ ЗІР, ЗГОРТКОВА НЕЙРОННА МЕРЕЖА, КЛАСИФІКАЦІЯ ЗОБРАЖЕНЬ.

ANNOTATION

The bachelor's thesis report: 71 pages, 12 figures, 8 tables, 4 appendices, 35 sources.

The object of the research is the process of automated diagnosis of plant diseases based on digital images.

The purpose of the qualification work is to develop a software module for predicting diseases of agricultural crops based on convolutional neural networks, which provides automated analysis of plant images and generates a prediction regarding their condition using modern deep learning methods.

The research methods include analysis of scientific sources in the field of computer vision, deep learning, convolutional neural networks, and plant disease prediction; modeling of the software module architecture; and development of a basic CNN model for classifying plant images.

The results of the study consist in the creation of a software module for predicting diseases of agricultural crops, which provides loading of plant images, their preprocessing, resizing and normalization, transfer of data to a convolutional neural network, determination of the plant condition class, formation of a probabilistic prediction estimate, and presentation of the results in a user-friendly form.

The results of the work can be used for the development of intelligent systems for monitoring the condition of agricultural crops, automated diagnosis of plant diseases, software solutions for decision support in agricultural production, as well as further research in the field of computer vision and deep learning.

Keywords: AGRICULTURAL CROPS, PLANT DISEASES, PREDICTION, COMPUTER VISION, CONVOLUTIONAL NEURAL NETWORK, IMAGE CLASSIFICATION.

ЗМІСТ

Вступ 7
1 Аналіз відомих рішень прогнозування хвороб сільськогосподарських культур 10
1.1 Опис предметної області 10
1.2 Аналіз існуючих підходів та методів 12
1.3 Постановка задачі дослідження 14
2 Проєктування програмного модуля прогнозування хвороб сільськогосподарських культур 17
2.1 Концептуальні основи створення програмного модуля 17
2.2 Архітектура програмного модуля та алгоритмічне забезпечення 20
2.3 Підготовка та попередня обробка даних 24
2.4 Розроблення моделі згорткової нейронної мережі 28
3 Реалізація програмного модуля та експериментальні дослідження 31
3.1 Вибір засобів реалізації та програмного забезпечення 31
3.2 Реалізація структури програмного модуля 33
3.3 Реалізація моделей прогнозування 37
3.4 Експериментальне дослідження ефективності програмного модуля 40
Висновки 43
Список використаних джерел 45
Додаток А Програмні коди проєкту 49
Додаток Б Скрипти навчання та тестування моделей 54
Додаток В Конфігурації та параметри експериментів 62
Додаток Г Апробація результатів роботи 64

ВСТУП

Сільське господарство є однією з ключових галузей економіки, стабільне функціонування якої безпосередньо впливає на продовольчу безпеку та соціально-економічний розвиток держави. Однією з основних причин зниження врожайності сільськогосподарських культур є поширення різноманітних хвороб рослин, які за відсутності своєчасної діагностики можуть призводити до значних економічних втрат. У зв'язку з цим актуальним завданням є розроблення ефективних методів раннього прогнозування та виявлення захворювань культур.

Традиційні способи діагностики хвороб рослин базуються переважно на візуальному огляді посівів фахівцями або проведенні лабораторних аналізів. Такі підходи, хоча й забезпечують достатню точність, характеризуються високою трудомісткістю, значними часовими витратами та суб'єктивністю оцінювання. Крім того, їх застосування є ускладненим у випадку великих аграрних площ, що обмежує можливість оперативного реагування на появу захворювань.

Розвиток цифрових технологій та методів штучного інтелекту відкрив нові можливості для автоматизації процесів моніторингу стану рослин. Зокрема, методи комп'ютерного зору у поєднанні з глибоким навчанням дозволяють здійснювати аналіз зображень листя та інших частин рослин з метою виявлення характерних ознак захворювань [5, 18, 20]. Серед таких методів особливе місце посідають згорткові нейронні мережі, які здатні автоматично формувати інформативні просторові ознаки та демонструють високі показники точності у задачах класифікації зображень [1, 7, 27].

Ефективність застосування згорткових нейронних мереж у задачах прогнозування хвороб рослин значною мірою залежить від якості навчальних даних, вибору архітектури моделі та параметрів навчання. Дослідження свідчать, що використання сучасних архітектур глибокого навчання, а також методів аугментації даних і перенесення навчання, дозволяє підвищити узагальнювальну здатність моделей навіть за обмежених обсягів навчальної вибірки [1, 23, 27]. Це робить такі підходи перспективними для практичного використання в аграрних інформаційних системах.

Важливим аспектом упровадження методів глибокого навчання в аграрній сфері є розроблення програмних засобів, що забезпечують зручність використання, відтворюваність результатів і можливість інтеграції з іншими інформаційними системами. Програмний модуль прогнозування хвороб сільськогосподарських культур повинен реалізовувати повний цикл обробки даних – від попередньої обробки зображень до отримання результатів класифікації та їх інтерпретації.

У зв'язку з викладеним, актуальною є задача створення програмного модуля прогнозування хвороб сільськогосподарських культур на основі згорткових нейронних мереж, який поєднує високу точність розпізнавання із практичною придатністю для використання в умовах реального агровиробництва.

Метою кваліфікаційної роботи є розробка програмного модуля прогнозування хвороб сільськогосподарських культур на основі згорткових нейронних мереж, який забезпечує автоматизований аналіз зображень рослин і формування прогнозу щодо їхнього стану з використанням сучасних методів глибокого навчання.

Для досягнення поставленої мети у роботі необхідно розв'язати такі завдання:

– проаналізувати особливості задачі прогнозування хвороб рослин та специфіку вхідних даних;

– виконати огляд і порівняльний аналіз існуючих методів і архітектур згорткових нейронних мереж, що застосовуються у задачах агровізії;

– спроєктувати архітектуру програмного модуля прогнозування;

– реалізувати базову згорткову нейронну мережу та модель на основі перенесення навчання у межах єдиного програмного рішення;

– організувати процес навчання моделей, зокрема у режимі fine-tuning;

– провести експериментальне дослідження ефективності реалізованих моделей з використанням стандартних метрик оцінювання;

– виконати аналіз отриманих результатів та сформулювати висновки щодо доцільності використання обраних підходів.

Об’єктом дослідження є процес автоматизованої діагностики хвороб рослин за цифровими зображеннями.

Предметом дослідження є методи та моделі глибокого навчання, зокрема згорткові нейронні мережі та підходи перенесення навчання, а також програмні засоби їх реалізації.

Результати кваліфікаційної роботи апробовані та опубліковані у матеріалах IV Всеукраїнської науково-практичної конференції молодих вчених і студентів «Інтелектуальні комп’ютерні системи та мережі» (ІКСМ), м. Тернопіль, ЗУНУ, 20 травня 2026 р. (додаток Г).

Кваліфікаційна робота складається із вступу, трьох розділів, висновків, списку використаних джерел та додатків.

1 АНАЛІЗ ВІДОМИХ РІШЕНЬ ПРОГНОЗУВАННЯ ХВОРОБ СІЛЬСЬКОГОСПОДАРСЬКИХ КУЛЬТУР

1.1 Опис предметної області

Сільське господарство є однією з базових галузей економіки, ефективність якої визначається не лише технологіями вирощування, але й здатністю оперативно реагувати на фактори ризику, зокрема захворювання рослин. Хвороби сільськогосподарських культур можуть спричиняти істотні втрати врожаю, погіршення товарних характеристик продукції та збільшення витрат на захисні заходи. У практиці агровиробництва вирішальним чинником є своєчасність виявлення симптомів і правильність визначення типу ураження, оскільки тактика обробки рослин залежить від конкретного збудника та стадії розвитку хвороби.

Традиційна діагностика захворювань рослин здебільшого ґрунтується на візуальному огляді посівів фахівцями-агрономами або працівниками господарств. Однак такий підхід має низку суттєвих обмежень. По-перше, оцінювання ознак ураження носить суб’єктивний характер і залежить від досвіду спеціаліста. По-друге, контроль великих площ потребує значних ресурсів часу, що ускладнює регулярний моніторинг. По-третє, симптоми різних захворювань часто мають подібні візуальні прояви (плямистість, пожовтіння, некроз), особливо на ранніх стадіях, що підвищує імовірність помилкового висновку. У результаті актуалізується потреба у технологіях, здатних підтримати або автоматизувати прийняття рішень щодо стану рослин на основі об’єктивних даних.

Одним із найбільш доступних і водночас інформативних джерел даних для діагностики є цифрові зображення рослин (листя, стебел, плодів). Аналіз зображень дозволяє фіксувати візуальні симптоми ураження, а також використовувати накопичені приклади для побудови моделей прогнозування. У сучасних дослідженнях для розв’язання задач класифікації зображень широко застосовуються методи глибокого навчання, зокрема згорткові нейронні мережі (CNN), які продемонстрували високу результативність у діагностиці хвороб

рослин [5, 18]. Перевага CNN полягає у здатності автоматично виділяти ознаки різних рівнів складності: від простих контурів і текстур до комплексних патернів, що відповідають специфічним проявам захворювань.

Важливою характеристикою предметної області є складність і неоднорідність вхідних даних. У реальних умовах зйомка рослин здійснюється за різного освітлення, під різними кутами, з різною відстанню до об’єкта та на неоднаковому фоні. На якість зображень можуть впливати тіні, відблиски, погодні умови, а також часткові перекриття листя іншими елементами рослини. Окремою проблемою є те, що симптоми хвороб змінюються в часі: на ранніх стадіях прояви можуть бути слабко вираженими, тоді як на пізніших стадіях виникають виразні некротичні ділянки або деформації. У контексті машинного навчання це означає необхідність врахування високої варіативності даних та забезпечення здатності моделі до узагальнення.

Ще одним фактором, що ускладнює задачу прогнозування, є дисбаланс класів: у практичних наборах даних кількість зображень здорових рослин часто переважає кількість прикладів окремих хвороб. Це може призводити до зміщення моделі в бік домінантного класу, якщо не застосовувати відповідні стратегії навчання (збалансування вибірки, корекція ваг класів, розширення даних). Водночас збільшення обсягу та різноманітності датасету загалом підвищує якість моделей глибокого навчання, що підтверджується результатами досліджень про вплив розміру та варіативності даних на ефективність класифікації хвороб рослин [1].

Практичне впровадження системи прогнозування захворювань рослин потребує не лише побудови моделі, але й реалізації програмного рішення, придатного для використання у реальних сценаріях. До таких сценаріїв належать застосування у фермерських господарствах для оперативного аналізу стану рослин, інтеграція у мобільні застосунки для польової діагностики, а також використання у системах моніторингу аграрних площ. У всіх випадках важливо забезпечити однозначний інтерфейс взаємодії, коректну підготовку вхідних зображень, надійність прогнозування та зрозуміле представлення результатів, зокрема рівня впевненості прогнозу.

Отже, предметна область даного дослідження охоплює задачу прогнозування хвороб сільськогосподарських культур за зображеннями рослин із використанням згорткових нейронних мереж. Її особливостями є висока варіативність вхідних даних, можливий дисбаланс класів та необхідність практичної реалізації програмного модуля, здатного забезпечити автоматизоване прогнозування стану рослин у прикладних умовах [5, 18].

1.2 Аналіз існуючих підходів та методів

Задача автоматизованого прогнозування хвороб сільськогосподарських культур за зображеннями рослин є предметом активних наукових досліджень, у межах яких запропоновано широкий спектр підходів – від класичних методів комп'ютерного зору до сучасних моделей глибокого навчання. Аналіз існуючих рішень дозволяє оцінити їх ефективність, визначити обмеження та обґрунтувати вибір методів, використаних у даній роботі.

На ранніх етапах розвитку автоматизованих систем діагностики рослин застосовувалися класичні методи обробки зображень і машинного навчання. Типовий підхід передбачав сегментацію зображення, ручне виділення ознак, таких як колірні характеристики, текстурні параметри та геометричні особливості уражених ділянок, з подальшою класифікацією за допомогою традиційних алгоритмів. Перевагою таких методів є відносна простота реалізації та невисокі вимоги до обчислювальних ресурсів, однак їхня ефективність істотно залежить від якості сформованих ознак. За умов високої варіативності вхідних зображень такі підходи демонструють обмежену здатність до узагальнення та чутливість до змін умов зйомки.

Подальший розвиток досліджень пов'язаний із широким упровадженням методів глибокого навчання, зокрема згорткових нейронних мереж, які стали базовим інструментом аналізу зображень [14]. У задачах агровізії CNN продемонстрували здатність автоматично виділяти багаторівневі ознаки, що відображають складні візуальні патерни захворювань рослин. Результати численних досліджень свідчать про високу ефективність CNN у задачах

класифікації хвороб рослин за зображеннями листя [5, 18, 20]. При цьому досягаються високі значення точності, особливо у випадках використання контрольованих наборів даних з однорідним фоном.

Разом із тим аналіз літератури показує, що висока точність, досягнута на лабораторних датасетах, не завжди зберігається при застосуванні моделей у реальних польових умовах. Однією з ключових причин є обмежена репрезентативність навчальних даних. У роботах, присвячених впливу розміру та різноманітності датасету, підкреслюється, що саме варіативність даних, а не лише їх кількість, визначає узагальнювальну здатність моделей глибокого навчання [1]. Недостатня різноманітність зображень може призводити до перенавчання та зниження якості прогнозування на нових даних.

Важливим напрямом розвитку є використання різних архітектур згорткових нейронних мереж. У сучасних дослідженнях застосовуються як відносно прості CNN, так і глибокі архітектури загального призначення, зокрема VGG, ResNet та EfficientNet [8, 24, 26]. Глибші моделі зазвичай забезпечують кращі показники точності завдяки здатності формувати складніші представлення ознак, однак вони характеризуються підвищеними обчислювальними вимогами. Це створює обмеження для використання таких моделей у прикладних системах з обмеженими ресурсами, зокрема у мобільних або вбудованих рішеннях.

Значну увагу в сучасній літературі приділено підходам на основі перенесення навчання. Суть цього методу полягає у використанні моделей, попередньо навчених на великих універсальних наборах зображень, таких як ImageNet [4], з подальшим донавчанням під конкретну предметну область. Дослідження показують, що застосування перенесення навчання та fine-tuning дозволяє суттєво підвищити якість класифікації хвороб рослин за умов обмеженого обсягу спеціалізованих даних [27]. Особливо ефективним є донавчання лише частини параметрів моделі, що знижує ризик перенавчання та скорочує час навчання.

Окрему роль у формуванні ефективних рішень відіграють відкриті набори даних зображень рослин, які широко використовуються для навчання та порівняльного аналізу моделей [10, 25]. Такі датасети створюють основу для

розвитку досліджень, однак часто містять зображення, отримані в контрольованих умовах. Це обмежує здатність моделей, навчених на таких даних, до узагальнення у реальних сценаріях застосування, де присутні різні фони, освітлення та ступені ураження рослин.

Аналіз відомих рішень також показує, що значна частина досліджень зосереджена на досягненні високих показників точності класифікації, тоді як питання практичної реалізації програмного забезпечення розглядаються обмежено. У багатьох роботах відсутній опис цілісного програмного модуля, який би охоплював усі етапи обробки даних – від введення зображень і попередньої обробки до формування результатів прогнозування у зручному для користувача вигляді. Крім того, рідко проводиться комплексний порівняльний аналіз базових згорткових нейронних мереж і моделей на основі перенесення навчання в межах одного програмного рішення.

Таким чином, результати аналізу літератури свідчать, що згорткові нейронні мережі та підходи перенесення навчання є найбільш перспективними для задач прогнозування хвороб сільськогосподарських культур. Водночас актуальною залишається задача розробки прикладного програмного модуля, який поєднує ефективні моделі глибокого навчання з модульною архітектурою та можливістю практичного застосування. Виявлені обмеження та тенденції розвитку існуючих підходів створюють підґрунтя для формулювання задачі дослідження, що розглядається у наступному підрозділі.

1.3 Постановка задачі дослідження

Проведений аналіз предметної області та існуючих підходів до прогнозування хвороб сільськогосподарських культур показав, що згорткові нейронні мережі є одним із найбільш ефективних інструментів для автоматизованого аналізу зображень рослин. Сучасні дослідження підтверджують здатність CNN забезпечувати високі показники точності класифікації за рахунок автоматичного виділення просторових ознак, характерних для різних типів захворювань. Водночас більшість наукових робіт

зосереджена переважно на вдосконаленні моделей глибокого навчання, тоді як питання практичної реалізації таких підходів у вигляді завершених програмних модулів залишаються недостатньо опрацьованими.

У реальних умовах аграрного виробництва використання методів глибокого навчання потребує не лише наявності ефективної моделі, а й програмного забезпечення, здатного забезпечити повний цикл обробки даних. Такий цикл охоплює завантаження та попередню обробку зображень, виконання прогнозування на основі згорткової нейронної мережі, а також формування результатів у зрозумілому та зручному для користувача вигляді. Недостатня увага до цих аспектів ускладнює впровадження результатів наукових досліджень у практичну діяльність аграрних підприємств.

З урахуванням викладеного, у межах даної кваліфікаційної роботи ставиться задача розроблення програмного модуля прогнозування хвороб сільськогосподарських культур на основі згорткових нейронних мереж. Основний акцент робиться на поєднанні алгоритмічних методів глибокого навчання з продуманою архітектурою програмного забезпечення, що забезпечує відтворюваність результатів, зручність використання та можливість подальшого розширення функціональності.

У роботах, присвячених застосуванню CNN і перенесення навчання, підкреслюється залежність ефективності моделей від якості та різноманітності даних, а також від вибору архітектури і стратегії навчання. Це обумовлює доцільність реалізації програмного модуля, який поєднує базову згорткову нейронну мережу та модель на основі перенесення навчання, що дозволяє виконати порівняльний аналіз і обґрунтовано обрати оптимальний підхід для практичного використання.

У зв’язку з цим метою даної кваліфікаційної роботи є розробка програмного модуля прогнозування хвороб сільськогосподарських культур на основі згорткових нейронних мереж, який забезпечує автоматизований аналіз зображень рослин і формування прогнозу щодо їхнього стану з використанням сучасних методів глибокого навчання.

Для досягнення поставленої мети у роботі необхідно розв’язати такі завдання:

– проаналізувати особливості задачі прогнозування хвороб рослин та специфіку вхідних даних;

– виконати огляд і порівняльний аналіз існуючих методів і архітектур згорткових нейронних мереж, що застосовуються у задачах агровізії;

– спроєктувати архітектуру програмного модуля прогнозування;

– реалізувати базову згорткову нейронну мережу та модель на основі перенесення навчання у межах єдиного програмного рішення;

– організувати процес навчання моделей, зокрема у режимі fine-tuning;

– провести експериментальне дослідження ефективності реалізованих моделей з використанням стандартних метрик оцінювання;

– виконати аналіз отриманих результатів та сформулювати висновки щодо доцільності використання обраних підходів.

Об’єктом дослідження є процес автоматизованої діагностики хвороб рослин за цифровими зображеннями.

Предметом дослідження є методи та моделі глибокого навчання, зокрема згорткові нейронні мережі та підходи перенесення навчання, а також програмні засоби їх реалізації.

Практичне значення роботи полягає у можливості використання розробленого програмного модуля як складової прикладних систем моніторингу стану сільськогосподарських культур, а також як основи для подальшого розширення функціональності та інтеграції з іншими інформаційними системами аграрного призначення.

2 ПРОЄКТУВАННЯ ПРОГРАМНОГО МОДУЛЯ ПРОГНОЗУВАННЯ ХВОРОБ СІЛЬСЬКОГОСПОДАРСЬКИХ КУЛЬТУР

2.1 Концептуальні основи створення програмного модуля

Розроблення програмного модуля прогнозування хвороб сільськогосподарських культур на основі згорткових нейронних мереж потребує чіткого визначення вимог, які забезпечують його коректне функціонування, практичну придатність та відтворюваність результатів. Формування таких вимог ґрунтується на особливостях предметної області, специфіці вхідних даних, а також на аналізі існуючих підходів до автоматизованої діагностики захворювань рослин [5, 18, 20].

Основні функціональні та нефункціональні вимоги до програмного модуля наведено в таблиці 2.1.

Таблиця 2.1 – Основні функціональні та нефункціональні вимоги до програмного модуля

№	Тип вимог	Назва вимоги	Опис вимоги
1	2	3	4
1	Функціональна	Завантаження зображень	Програмний модуль повинен забезпечувати завантаження цифрових зображень рослин у поширених графічних форматах (JPEG, PNG).
2	Функціональна	Попередня обробка даних	Модуль повинен автоматично виконувати масштабування, нормалізацію та інші операції попередньої обробки зображень перед подачею у нейронну мережу.
3	Функціональна	Прогнозування стану рослин	Програмний модуль повинен здійснювати прогнозування стану рослин (здоровий стан або наявність захворювання) на основі згорткової нейронної мережі.
4	Функціональна	Формування результатів	Модуль повинен відображати результат прогнозування у вигляді класу захворювання та відповідних значень ймовірностей.

Продовження таблиці 2.1

1	2	3	4
5	Функціональна	Збереження результатів	Передбачена можливість збереження результатів прогнозування для подальшого аналізу.
6	Нефункціональна	Продуктивність	Час обробки одного зображення повинен забезпечувати можливість практичного використання модуля без суттєвих затримок.
7	Нефункціональна	Точність прогнозування	Програмний модуль повинен забезпечувати достатній рівень точності класифікації, що визначається експериментальними показниками якості.
8	Нефункціональна	Надійність	Модуль повинен коректно обробляти помилкові або некоректні вхідні дані без аварійного завершення роботи.
9	Нефункціональна	Масштабованість	Архітектура програмного модуля повинна дозволяти розширення функціональності та додавання нових класів захворювань.
10	Нефункціональна	Зручність використання	Інтерфейс та логіка роботи програмного модуля повинні бути зрозумілими для користувача без спеціальної підготовки.

Основним функціональним призначенням програмного модуля є автоматизоване прогнозування наявності хвороб сільськогосподарських культур за зображеннями рослин шляхом застосування згорткової нейронної мережі. Модуль повинен забезпечувати повний цикл обробки даних, починаючи з завантаження зображень і завершуючи формуванням результату класифікації у зручній для користувача формі.

До вхідних даних програмного модуля висуваються такі вимоги:

– можливість роботи з цифровими зображеннями рослин у поширених графічних форматах;

– підтримка зображень, отриманих в умовах природного освітлення та з різним фоном;

– автоматичне приведення зображень до фіксованого розміру та формату, необхідного для подальшої обробки згортковою нейронною мережею;

– забезпечення коректної нормалізації значень пікселів відповідно до обраної архітектури моделі [1, 23].

Вихідні дані програмного модуля повинні містити:

– прогнозований клас стану рослини (здоровий стан або конкретний тип захворювання);

– значення ймовірностей належності зображення до кожного з можливих класів;

– можливість збереження або візуалізації результатів для подальшого аналізу.

Окрім функціональних вимог, до програмного модуля висуваються нефункціональні вимоги, що визначають його якість та зручність використання. Зокрема, модуль повинен забезпечувати достатню швидкодію під час прогнозування, що є важливим для практичного застосування в умовах аграрного виробництва. Також необхідною є стабільність роботи та коректна обробка помилкових або некоректних вхідних даних.

Важливою вимогою є точність прогнозування, яка безпосередньо залежить від архітектури згорткової нейронної мережі та параметрів її навчання. Програмний модуль має бути спроєктований таким чином, щоб забезпечувати можливість зміни гіперпараметрів моделі та проведення експериментів із різними конфігураціями нейронної мережі [27]. Це дозволяє адаптувати систему до різних наборів даних і умов використання.

З огляду на перспективу подальшого розвитку, програмний модуль повинен мати модульну структуру, яка спрощує розширення функціональності, зокрема шляхом додавання нових класів захворювань або інтеграції з іншими інформаційними системами. Такий підхід відповідає сучасним тенденціям побудови програмних засобів на основі методів глибокого навчання [19].

Таким чином, сформульовані вимоги до програмного модуля створюють основу для подальшого проєктування його архітектури та реалізації алгоритмів прогнозування хвороб сільськогосподарських культур на основі згорткових

нейронних мереж. Дотримання зазначених вимог дозволяє забезпечити поєднання високої точності прогнозування з практичною придатністю розробленого програмного рішення.

2.2 Архітектура програмного модуля та алгоритмічне забезпечення

Архітектура програмного модуля прогнозування хвороб сільськогосподарських культур визначає логіку взаємодії його основних компонентів та забезпечує послідовну обробку даних на всіх етапах роботи системи. Під час проєктування архітектури враховувалися вимоги до модульності, масштабованості та можливості подальшого розширення функціональності, що є важливим для програмних засобів, орієнтованих на використання методів глибокого навчання [19].

Загальна архітектура програмного модуля має багаторівневу структуру та включає кілька логічно відокремлених компонентів, кожен з яких відповідає за виконання окремих функцій. Основними компонентами системи є модуль введення даних, модуль попередньої обробки зображень, модуль прогнозування на основі згорткової нейронної мережі та модуль формування і представлення результатів.

Модуль введення даних забезпечує завантаження зображень рослин у систему та виконує первинну перевірку їх коректності. На цьому етапі здійснюється перевірка формату файлів, розміру зображень та їх доступності для подальшої обробки. Такий підхід дозволяє зменшити ймовірність помилок на наступних етапах роботи програмного модуля.

Модуль попередньої обробки зображень призначений для підготовки вхідних даних до подачі у згорткову нейронну мережу. До його основних функцій належать масштабування зображень до фіксованого розміру, нормалізація значень пікселів, а також виконання операцій, спрямованих на зменшення впливу шумів та варіацій умов зйомки. Реалізація цього модуля є критичною для забезпечення стабільності та узагальнювальної здатності моделі прогнозування [1, 23].

Ключовим компонентом архітектури є модуль прогнозування, який реалізує згорткову нейронну мережу для класифікації стану рослин. Даний модуль відповідає за завантаження попередньо навченої моделі, обробку підготовлених зображень та обчислення ймовірностей належності до кожного з класів. Використання згорткових нейронних мереж у цьому модулі зумовлене їх здатністю ефективно виявляти просторові ознаки, характерні для різних типів захворювань рослин [5, 18, 20].

Модуль формування результатів забезпечує інтерпретацію вихідних даних нейронної мережі та їх представлення у зручному для користувача вигляді. Результати прогнозування можуть відображатися у вигляді текстових повідомлень, таблиць або графічних елементів, що містять інформацію про прогнозований клас і відповідні значення ймовірностей. Такий підхід підвищує наочність результатів і полегшує їх подальший аналіз.

Взаємодія між компонентами програмного модуля відбувається послідовно, відповідно до визначеного потоку обробки даних. Зображення, отримані на вході системи, передаються до модуля попередньої обробки, після чого підготовлені дані надходять до модуля прогнозування. Результати обчислень передаються до модуля формування результатів, який завершує цикл роботи системи. Така організація забезпечує прозорість обробки даних та спрощує налагодження й супровід програмного модуля.

Загальна структура програмного модуля прогнозування хвороб сільськогосподарських культур на основі згорткових нейронних мереж може бути представлена у вигляді структурної схеми, що ілюструє основні компоненти системи та напрямки передачі даних між ними.

На рисунку 2.1 представлено архітектуру програмного модуля прогнозування хвороб сільськогосподарських культур.

На рисунку 2.2 представлено узагальнений алгоритм програмного модуля прогнозування хвороб сільськогосподарських культур.

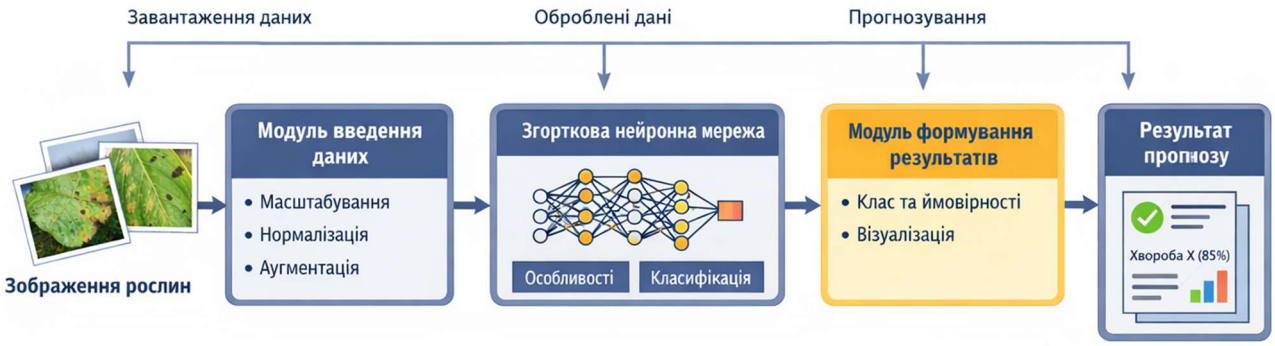

Рисунок 2.1 – Структурна схема програмного модуля прогнозування хвороб сільськогосподарських культур [30]

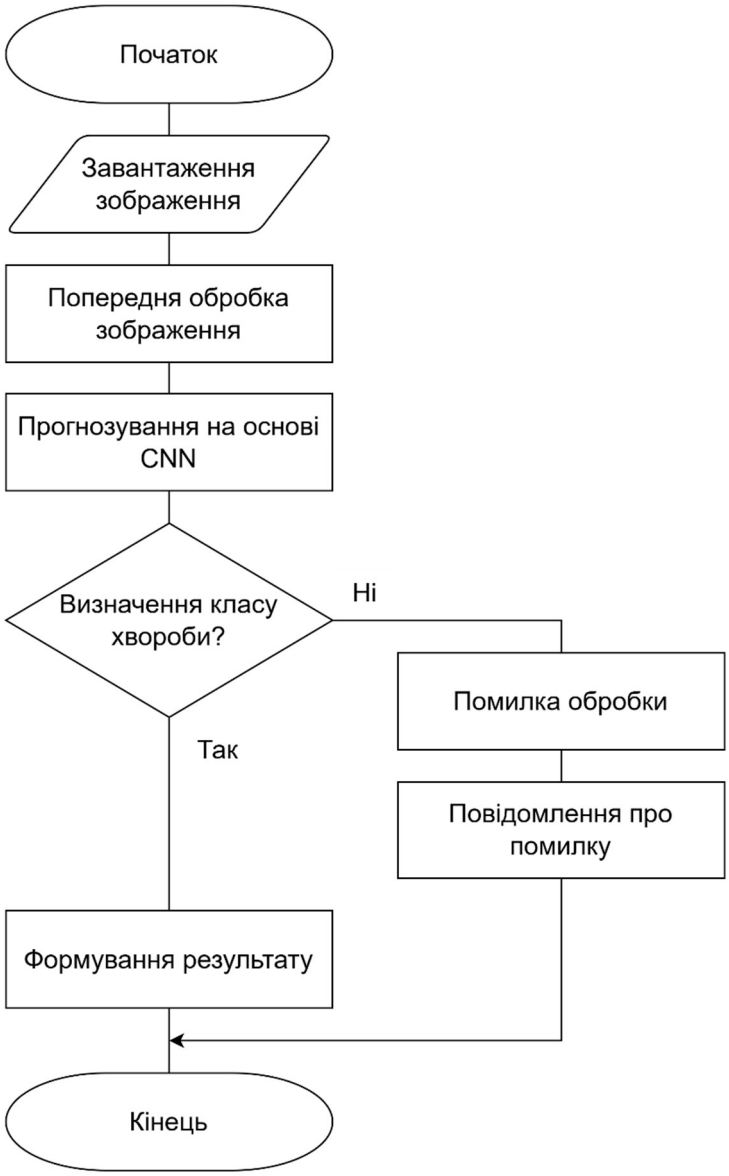

Рисунок 2.2 – Узагальнений алгоритм програмного модуля прогнозування хвороб сільськогосподарських культур

Псевдокод узагальненого алгоритму прогнозування хвороб рослин:

Алгоритм Прогнозування_Хвороб_Рослин
Вхід: зображення рослини Img
Вихід: клас_хвороби, ймовірність

1: Ініціалізувати параметри програмного модуля
2: Завантажити збережену модель CNN
3: Якщо Img = NULL або формат некоректний тоді
4: Вивести повідомлення про помилку
5: Завершити алгоритм
6: Кінець якщо
7: Виконати попередню обробку Img
8: Масштабувати зображення
9: Нормалізувати значення пікселів
10: Сформувати тензор вхідних даних
11: Передати тензор у модель CNN
12: Отримати вектор ймовірностей класів
13: Визначити клас з максимальною ймовірністю
14: Сформувати результат прогнозування
15: Відобразити результат користувачу
16: За потреби зберегти результат
17: Завершити алгоритм
Кінець алгоритму

Отже, запропонована архітектура програмного модуля забезпечує чіткий розподіл функцій між компонентами, сприяє підвищенню надійності системи та створює передумови для її подальшого розвитку і адаптації до різних умов практичного застосування.

2.3 Підготовка та попередня обробка даних

Ефективність прогнозування хвороб сільськогосподарських культур на основі згорткових нейронних мереж значною мірою визначається якістю підготовки вхідних даних. Зображення рослин, отримані в реальних умовах, як правило, характеризуються варіативністю освітлення, фону, ракурсу зйомки та роздільної здатності, що ускладнює безпосереднє використання таких даних для навчання моделей глибокого навчання. У зв'язку з цим попередня обробка зображень є обов'язковим етапом побудови програмного модуля прогнозування хвороб рослин [1, 5].

Для забезпечення відтворюваності експериментального дослідження у даній роботі доцільно використати відкритий набір даних PlantVillage [31].

Фрагмент відкритого набору даних PlantVillage подано на рисунку 2.3.

Рисунок 2.3 – Фрагмент відкритого набору даних PlantVillage

Він містить цифрові зображення листя сільськогосподарських культур у здоровому стані та з ознаками різних захворювань. Датасет широко застосовується у дослідженнях, пов'язаних із комп'ютерним зором, глибоким навчанням і автоматизованою діагностикою хвороб рослин [10, 18].

Основною перевагою PlantVillage є наявність розмічених зображень, у яких кожне зображення належить до певного класу: здорова рослина або конкретний тип захворювання. Це дозволяє використовувати набір даних для задачі багатокласової класифікації, де на вхід моделі подається зображення листка, а на виході формується прогноз щодо стану рослини.

У межах даної роботи з набору PlantVillage сформовано підмножину з трьох класів: healthy, early_blight та late_blight. Такий вибір дозволяє перевірити роботу програмного модуля на прикладі розпізнавання здорових листків і двох поширених типів захворювань. Для забезпечення коректного навчання і тестування дані поділено на навчальну, валідаційну та тестову вибірки у співвідношенні 70 % / 15 % / 15 %.

Перед подачею у згорткову нейронну мережу зображення з PlantVillage проходять попередню обробку. Вона включає перетворення у колірний простір RGB, масштабування до розміру 224×224 пікселі, нормалізацію значень пікселів і формування тензора. Для підвищення узагальнювальної здатності моделі у навчальній вибірці застосовується аугментація зображень, зокрема горизонтальне віддзеркалення, незначне обертання та зміна яскравості.

Використання PlantVillage забезпечує відтворюваність експериментального дослідження, оскільки цей набір є відкритим і придатним для порівняння різних моделей глибокого навчання. Водночас слід враховувати, що значна частина зображень у цьому датасеті отримана в контрольованих умовах, тому результати, отримані на ньому, потребують подальшої перевірки на зображеннях, зроблених у реальних польових умовах.

Основні операції попередньої обробки зображень, що застосовуються у програмному модулі, наведено в таблиці 2.2.

На початковому етапі здійснюється формування набору даних на основі зображень PlantVillage. Для реалізації програмного модуля може бути сформовано підмножину зображень, яка включає класи здорових рослин і окремі класи захворювань. Такий підхід дозволяє адаптувати експеримент до задачі багатокласової класифікації та забезпечити наявність достатньої кількості прикладів для навчання, валідації і тестування моделі. Важливою вимогою

залишається забезпечення репрезентативності вибірки, оскільки дисбаланс між класами може негативно впливати на результати навчання згорткової нейронної мережі [1].

Таблиця 2.2 – Операції попередньої обробки зображень

№	Операція	Опис операції	Призначення
1	Масштабування зображень	Приведення всіх вхідних зображень до фіксованої роздільної здатності, що відповідає вимогам архітектури згорткової нейронної мережі.	Забезпечення узгодженості вхідних даних та зменшення обчислювальних витрат.
2	Перетворення колірного простору	Перетворення зображень у стандартний колірний простір (RGB).	Спрощення подальшої обробки та уніфікація вхідних даних.
3	Нормалізація значень пікселів	Приведення інтенсивностей пікселів до заданого числового діапазону.	Підвищення стабільності та швидкості навчання нейронної мережі.
4	Усунення шумів	Зменшення впливу шумів і артефактів, що виникають під час зйомки.	Підвищення якості вхідних даних та точності прогнозування.
5	Аугментація зображень	Генерація додаткових навчальних зразків шляхом виконання обертання, віддзеркалення, масштабування та зміни яскравості.	Підвищення узагальнювальної здатності моделі та зменшення ризику перенавчання.
6	Перемішування даних	Випадкове перемішування зображень у навчальній вибірці.	Запобігання формуванню небажаних залежностей під час навчання моделі.
7	Розподіл вибірки	Поділ набору даних на навчальну, валідаційну та тестову частини.	Отримання об’єктивної оцінки якості прогнозування.

Подальший етап підготовки даних передбачає приведення зображень до єдиного формату. З цією метою всі вхідні зображення масштабуються до фіксованої роздільної здатності, що відповідає вимогам обраної архітектури нейронної мережі. Такий підхід дозволяє забезпечити узгодженість вхідних

даних і зменшити обчислювальні витрати під час навчання моделі. Крім того, зображення перетворюються у стандартний колірний простір RGB, що спрощує подальшу обробку та аналіз.

Важливим етапом є нормалізація значень пікселів, яка полягає у приведенні інтенсивностей до заданого числового діапазону. Нормалізація сприяє стабільнішому та швидшому навчанню нейронної мережі, зменшуючи вплив масштабів вхідних даних на процес оптимізації [11]. У сучасних підходах до глибокого навчання така операція є стандартною складовою конвеєра обробки зображень.

З метою підвищення узагальнювальної здатності моделі та зменшення ризику перенавчання у програмному модулі передбачено застосування методів аугментації даних. Аугментація полягає у штучному розширенні навчальної вибірки шляхом виконання різноманітних перетворень над вихідними зображеннями, зокрема обертання, дзеркального відображення, масштабування та зміни яскравості. Використання таких методів дозволяє моделі краще адаптуватися до варіацій умов зйомки та підвищує її стійкість до шумів у даних [23].

Після виконання операцій попередньої обробки здійснюється розподіл набору даних на навчальну, валідаційну та тестову вибірки. Навчальна вибірка використовується безпосередньо для навчання згорткової нейронної мережі, валідаційна – для налаштування гіперпараметрів та контролю процесу навчання, а тестова – для остаточного оцінювання якості прогнозування. Такий підхід дозволяє отримати об’єктивну оцінку ефективності розробленого програмного модуля [27].

Отже, використання відкритого набору даних PlantVillage у поєднанні з послідовною попередньою обробкою зображень створює основу для відтворюваного навчання та тестування моделі прогнозування хвороб сільськогосподарських культур. Реалізація зазначених процедур забезпечує підвищення якості навчання згорткової нейронної мережі та створює передумови для отримання стабільних і достовірних результатів прогнозування.

2.4 Розроблення моделі згорткової нейронної мережі

Центральним елементом програмного модуля прогнозування хвороб сільськогосподарських культур є модель згорткової нейронної мережі, яка виконує автоматизований аналіз зображень рослин та формує прогноз щодо їхнього стану. Вибір саме згорткової нейронної мережі зумовлений її здатністю ефективно працювати з просторово структурованими даними та виділяти інформативні ознаки, характерні для різних типів захворювань рослин [5, 18, 20].

Архітектура розробленої моделі [30] побудована за принципом послідовної обробки даних і складається з кількох функціональних блоків, що включають згорткові шари, шари підвибірки, шар нормалізації, повнозв'язаний шар та вихідний шар класифікації. Загальну структуру згорткової нейронної мережі, використаної у програмному модулі, подано на рисунку 2.3.

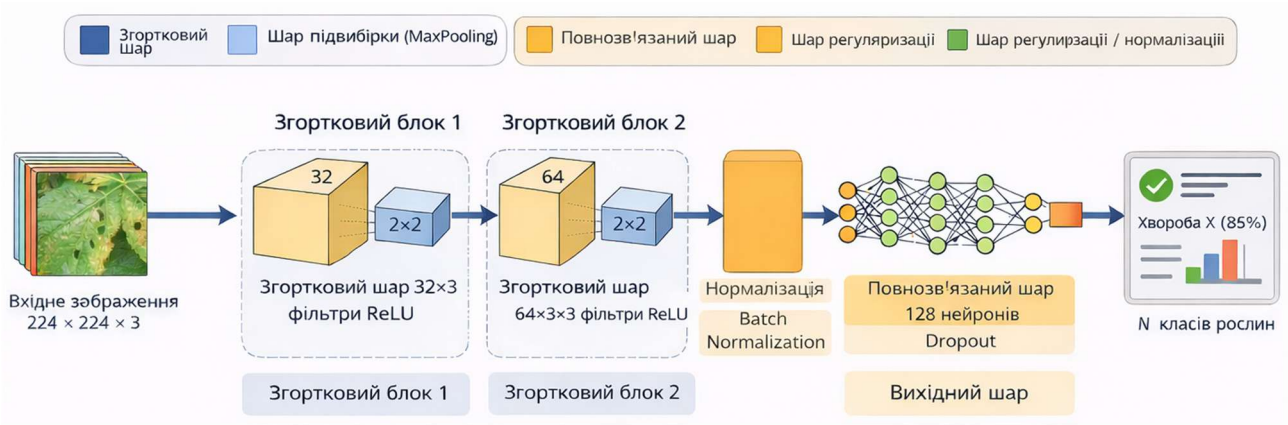

Рисунок 2.3 – Архітектура згорткової нейронної мережі [30]

На вхід нейронної мережі подаються попередньо оброблені зображення рослин розміром 224×224×3 у форматі RGB. Перший згортковий блок містить згортковий шар із 32 фільтрами розміром 3×3, після якого застосовується функція активації ReLU та шар підвибірки MaxPooling із розміром вікна 2×2. Цей блок призначений для виділення базових просторових ознак, таких як контури та елементарні текстурні елементи.

Другий згортковий блок має аналогічну структуру, проте використовує 64 згорткові фільтри, що дозволяє виділяти складніші та більш абстрактні ознаки, пов’язані з характерними симптомами захворювань рослин. Застосування шару підвибірки на цьому етапі сприяє зменшенню просторової розмірності ознак і зниженню обчислювального навантаження, що є важливим для практичного використання програмного модуля.

Для підвищення стабільності процесу навчання та зменшення внутрішнього зміщення активацій у моделі використовується шар пакетної нормалізації (Batch Normalization). Даний підхід дозволяє пришвидшити збіжність алгоритму навчання та позитивно впливає на узагальнювальну здатність згорткової нейронної мережі [11].

Після завершення згорткових операцій сформовані ознаки передаються до повнозв’язаного шару, який містить 128 нейронів. Цей шар виконує інтеграцію виділених ознак і формує внутрішнє подання, необхідне для виконання класифікації. З метою зменшення ризику перенавчання у повнозв’язаному шарі застосовується регуляризація шляхом випадкового вимкнення нейронів (Dropout) з імовірністю 0,5.

Вихідний шар згорткової нейронної мережі містить N нейронів, де N відповідає кількості класів стану рослин. Для формування кінцевого прогнозу використовується функція активації Softmax, яка дозволяє отримати ймовірнісну оцінку належності зображення до кожного з можливих класів.

Основні параметри архітектури згорткової нейронної мережі, включно з кількістю шарів, розміром згорткових фільтрів, типами функцій активації та методами регуляризації, узагальнено в таблиці 2.3. Для навчання моделі використовується крос-ентропійна функція втрат, а оптимізація параметрів здійснюється за допомогою алгоритму Adam, який забезпечує адаптивне коригування швидкості навчання та стабільну збіжність процесу оптимізації [13].

Таблиця 2.3 – Параметри архітектури згорткової нейронної мережі

№	Компонент мережі	Основні параметри	Призначення
1	Вхідний шар	Розмір зображення: 224×224×3	Прийом попередньо оброблених зображень рослин у форматі RGB.
2	Згортковий шар 1	Кількість фільтрів – 32; розмір ядра – 3×3; крок – 1	Виділення базових просторових ознак (контури, текстури).
3	Функція активації	ReLU	Забезпечення нелінійності моделі та пришвидшення навчання.
4	Шар підвибірки 1	MaxPooling 2×2	Зменшення розмірності ознак та зниження обчислювальної складності.
5	Згортковий шар 2	Кількість фільтрів – 64; розмір ядра – 3×3	Виділення більш складних локальних ознак.
6	Функція активації	ReLU	Підвищення здатності мережі до апроксимації складних залежностей.
7	Шар підвибірки 2	MaxPooling 2×2	Подальше зменшення просторової розмірності ознак.
8	Шар нормалізації	Batch Normalization	Стабілізація процесу навчання та зменшення перенавчання.
9	Повнозв’язаний шар	Кількість нейронів – 128	Інтеграція виділених ознак для класифікації.
10	Шар регуляризації	Dropout – 0,5	Зменшення ризику перенавчання моделі.
11	Вихідний шар	Кількість нейронів – N (кількість класів)	Формування прогнозу щодо стану рослини.
12	Функція активації виходу	Softmax	Обчислення ймовірностей належності до кожного класу.
13	Функція втрат	Categorical Cross-Entropy	Оцінювання похибки прогнозування під час навчання.
14	Алгоритм оптимізації	Adam	Адаптивне коригування параметрів мережі під час навчання.

Таким чином, розроблена модель згорткової нейронної мережі поєднує відносну простоту архітектури з достатньою виразною здатністю для задач прогнозування хвороб сільськогосподарських культур. Обрана структура забезпечує баланс між точністю класифікації та обчислювальною ефективністю, що є важливим для подальшої реалізації та експериментального дослідження програмного модуля.

3 РЕАЛІЗАЦІЯ ПРОГРАМНОГО МОДУЛЯ ТА ЕКСПЕРИМЕНТАЛЬНІ ДОСЛІДЖЕННЯ

3.1 Вибір засобів реалізації та програмного забезпечення

Реалізація програмного модуля прогнозування хвороб сільськогосподарських культур на основі згорткових нейронних мереж потребує використання програмних засобів, які забезпечують обробку зображень, побудову моделей глибокого навчання, виконання прогнозування та аналіз отриманих результатів. Під час вибору інструментів враховано їх поширеність у задачах комп'ютерного зору, підтримку роботи з нейронними мережами, зручність налагодження програмного коду та можливість подальшого розширення програмного модуля.

Як основну мову програмування обрано Python. Такий вибір зумовлений наявністю розвиненої екосистеми бібліотек для машинного навчання, комп'ютерного зору, числових обчислень та візуалізації результатів. Python також дає змогу швидко реалізовувати окремі компоненти програмного модуля, тестувати моделі та змінювати параметри експериментів без суттєвої перебудови структури проєкту.

Для реалізації згорткових нейронних мереж використано фреймворк PyTorch. Він забезпечує побудову архітектури CNN, завантаження попередньо навчених моделей, виконання навчання та збереження ваг моделі у форматі .pth. У межах роботи PyTorch застосовано для реалізації базової згорткової нейронної мережі, а також для моделі на основі перенесення навчання.

Обробка зображень у програмному модулі здійснюється з використанням бібліотек torchvision та PIL. Їх застосування забезпечує завантаження зображень, перетворення у формат RGB, масштабування до заданої роздільної здатності, нормалізацію значень пікселів і формування тензорів, які подаються на вхід нейронної мережі. Для числових обчислень використано бібліотеку NumPy, а для побудови графіків точності та візуалізації результатів експериментів – Matplotlib.

Розроблення програмного модуля виконано у середовищі Visual Studio Code, яке забезпечує зручну роботу з файлами проєкту, керування структурою

програмного коду, запуск скриптів та перегляд результатів виконання. Середовище розробки дає змогу організувати проєкт за модулями, зокрема виділити окремі компоненти для налаштувань, обробки даних, опису моделей, прогнозування та формування результатів.

Програмні та апаратні засоби реалізації подано в таблиці 3.1.

Таблиця 3.1 – Програмні та апаратні засоби реалізації

Категорія	Найменування / характеристика
Мова програмування	Python
Фреймворк глибокого навчання	PyTorch
Бібліотеки для комп'ютерного зору	torchvision, PIL
Бібліотеки числових обчислень	NumPy
Засоби візуалізації результатів	Matplotlib
Середовище розробки	Visual Studio Code
Операційна система	Windows / Linux
Процесор	CPU з підтримкою багатопоточних обчислень
Графічний процесор (за наявності)	GPU з підтримкою CUDA
Оперативна пам'ять	не менше 8 ГБ
Формат збереження моделей	`.pth` (PyTorch)

На рисунку 3.1 наведено середовище розробки програмного модуля. У структурі проєкту виділено файли налаштувань, модулі завантаження та попередньої обробки зображень, файли реалізації моделей CNN і transfer learning, модуль прогнозування та файл запуску програмного рішення. Така організація проєкту забезпечує зрозумілу структуру програмного коду та спрощує подальше налагодження.

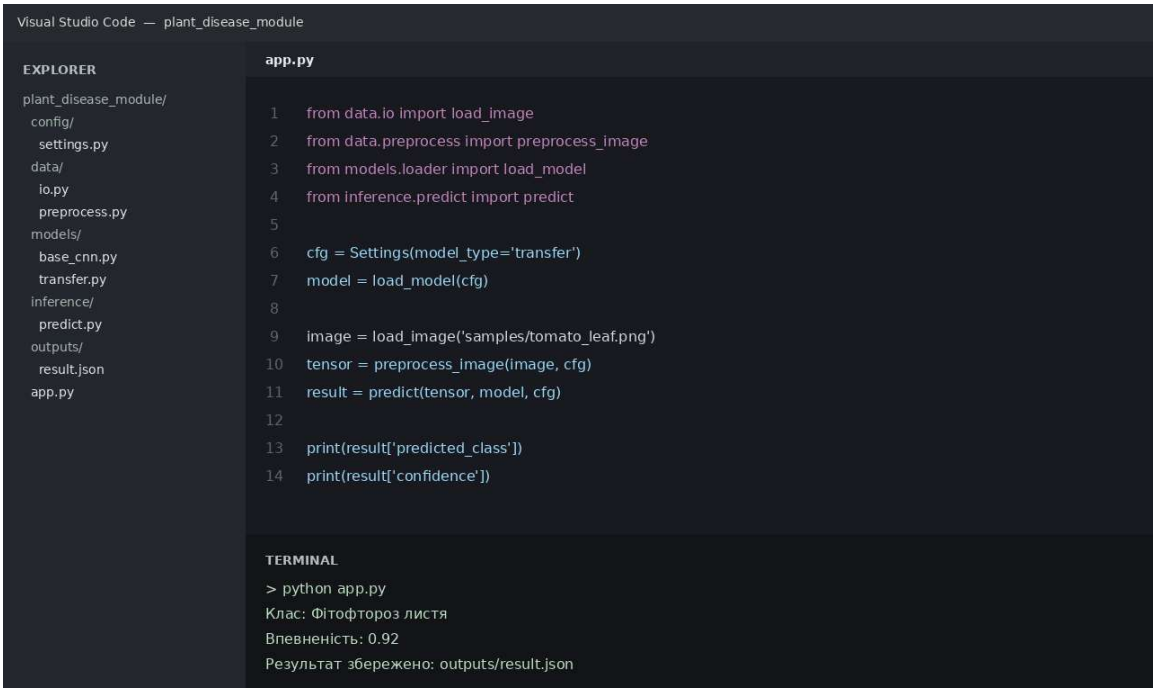

Рисунок 3.1 – Середовище розробки програмного модуля прогнозування хвороб рослин

Таким чином, обрані засоби реалізації забезпечують повну підтримку основних етапів роботи програмного модуля: завантаження зображень, їх попередню обробку, виконання прогнозування за допомогою згорткової нейронної мережі, збереження результатів та проведення експериментального оцінювання.

3.2 Реалізація структури програмного модуля

Реалізація програмного модуля прогнозування хвороб сільськогосподарських культур ґрунтується на модульному підході. Такий підхід передбачає поділ програмного рішення на окремі компоненти, кожен з яких виконує визначену функцію: завантаження зображень, попередню обробку даних, прогнозування стану рослини та формування результату для користувача.

У межах реалізації програмний модуль побудовано як послідовний конвеєр обробки даних. Спочатку користувач завантажує зображення листка рослини, після чого система виконує його перевірку та підготовку до подання на вхід згорткової нейронної мережі. Підготовлене зображення перетворюється у

тензор, передається до моделі прогнозування, а результат класифікації відображається у вигляді прогнозованого класу хвороби та значення впевненості моделі.

Основними компонентами програмного модуля є:

– модуль введення даних;
– модуль попередньої обробки зображень;
– модуль прогнозування;
– модуль формування результатів.

Модуль введення даних забезпечує завантаження зображення рослини з файлової системи. На цьому етапі перевіряється наявність файлу, його формат і можливість подальшого опрацювання. У програмній реалізації передбачено роботу із зображеннями у поширених графічних форматах, зокрема PNG та JPEG.

Модуль попередньої обробки зображень виконує підготовку даних до подання у нейронну мережу. У межах цього етапу зображення перетворюється у формат RGB, масштабується до розміру 224×224 пікселі, нормалізується та перетворюється у тензор. Така обробка забезпечує відповідність вхідних даних вимогам архітектури CNN-моделі.

На рисунку 3.2 наведено інтерфейс програмного модуля, який демонструє процес завантаження зображення та формування результату прогнозування.

Модуль прогнозування реалізує передавання підготовленого тензора до згорткової нейронної мережі. У результаті прямого поширення модель формує вектор імовірностей, який відображає належність вхідного зображення до кожного з визначених класів. Для вибору підсумкового результату використовується клас із найбільшим значенням імовірності.

Модуль формування результатів відповідає за інтерпретацію вихідних даних моделі та подання їх користувачеві. Результат містить назву прогнозованого класу, рівень впевненості моделі та розподіл імовірностей за класами. Такий формат є зручним для подальшого аналізу та може бути використаний у системах підтримки прийняття рішень в аграрному виробництві.

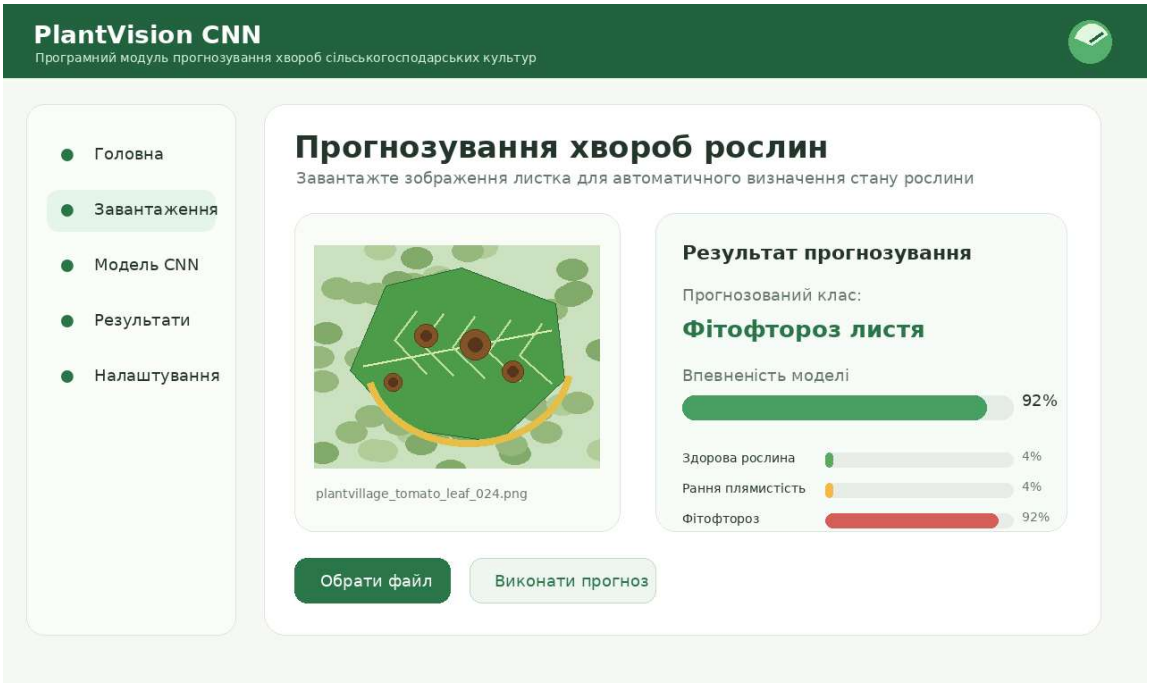

Рисунок 3.2 – Інтерфейс програмного модуля прогнозування хвороб рослин

Компоненти програмного модуля, їх призначення та інтерфейси подано у таблиці 3.2.

Таблиця 3.2 – Компоненти програмного модуля, їх призначення та інтерфейси

Компонент	Призначення	Вхідні дані	Вихідні дані
Модуль введення даних	Завантаження та первинна перевірка зображень	Шлях до файлу зображення	Зображення у внутрішньому форматі
Модуль попередньої обробки	Масштабування, нормалізація, формування тензора	Зображення	Тензор для CNN
Модуль прогнозування	Пряме поширення через CNN, обчислення ймовірностей	Тензор	Ймовірності класів
Модуль формування результатів	Інтерпретація та представлення результатів	Ймовірності класів	Клас захворювання, впевненість

Для навчання і тестування програмного модуля використовується відкритий набір даних PlantVillage. У процесі підготовки даних зображення розподіляються на навчальну, валідаційну та тестову вибірки. На рисунку 3.3 наведено приклад екранної форми, що відображає підготовку набору даних та основні етапи попередньої обробки зображень.

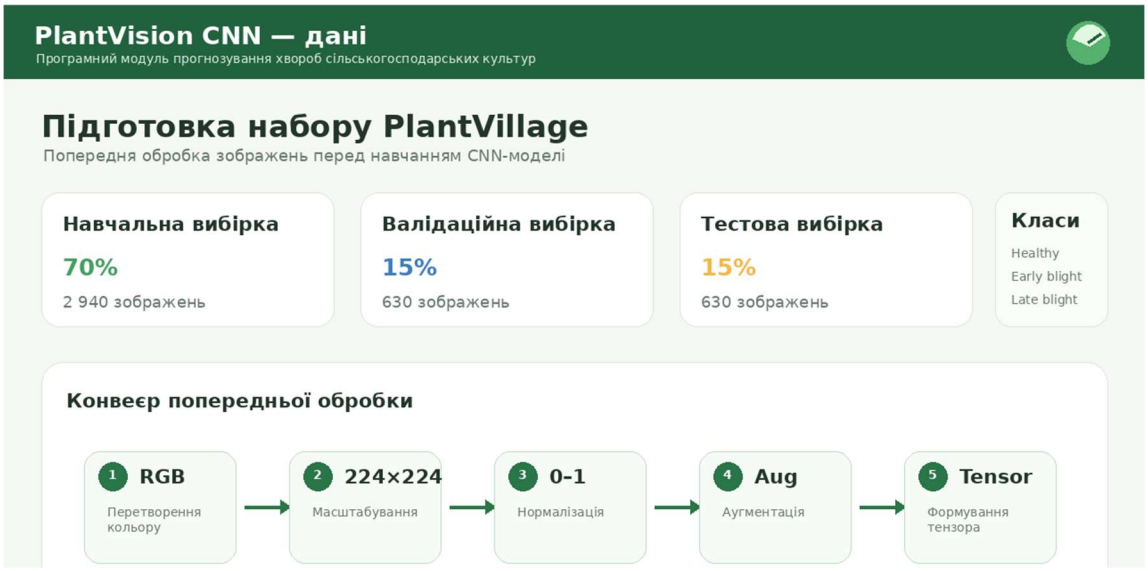

Рисунок 3.3 – Підготовка набору PlantVillage та попередня обробка зображень

Для ілюстрації взаємодії основних компонентів нижче наведено короткий фрагмент коду, який відображає загальну послідовність роботи програмного модуля (рисунок 3.4).

```
image = load_image(path)
tensor = preprocess_image(image)
result = model.predict(tensor)
output = format_result(result)
```

Рисунок 3.4 – Фрагмент коду, який відображає загальну послідовність роботи програмного модуля

Повна реалізація програмного модуля та відповідні програмні коди наведені у додатку А.

Отже, реалізована структура програмного модуля забезпечує послідовну обробку зображень рослин, уніфіковану взаємодію між компонентами та можливість подальшого розширення системи.

3.3 Реалізація моделей прогнозування

У програмному модулі реалізовано два підходи до прогнозування хвороб культур за зображеннями рослин: базову згорткову нейронну мережу та модель на основі перенесення навчання. Такий підхід дозволяє не лише виконати класифікацію стану рослини, але й порівняти ефективність простої CNN-архітектури з моделлю, яка використовує попередньо сформовані ознаки.

Базова згорткова нейронна мережа реалізована відповідно до архітектури, описаної у другому розділі. Вона містить послідовність згорткових шарів, шарів підвибірки, шару пакетної нормалізації, повнозв’язаної частини та шару регуляризації Dropout. На вхід моделі подається попередньо оброблене зображення розміром 224×224 пікселі у форматі RGB. Після проходження через згорткові блоки модель формує ознакове подання зображення, на основі якого виконується класифікація стану рослини.

Основними елементами базової CNN-моделі є згорткові шари з ядрами 3×3, функція активації ReLU, шари MaxPooling для зменшення просторової розмірності, Batch Normalization для стабілізації навчання та Dropout для зменшення ризику перенавчання. Вихідний шар моделі містить кількість нейронів, що відповідає кількості класів у вибраному наборі даних.

Для ілюстрації реалізації базової CNN-моделі на рисунку 3.5 наведено короткий фрагмент коду.

```
model = SimpleCNN(num_classes=num_classes)
criterion = nn.CrossEntropyLoss()
optimizer = torch.optim.Adam(model.parameters(), lr=0.001)
```

Рисунок 3.5 – Фрагмент коду, який відображає реалізацію базової CNN-моделі

Окрім базової CNN, у програмному модулі реалізовано модель на основі перенесення навчання. Для цього використано попередньо навчену архітектуру ResNet18, у якій замінено фінальний класифікаційний шар відповідно до кількості класів задачі прогнозування. Такий підхід є доцільним для роботи з набором PlantVillage, оскільки попередньо навчена модель уже містить узагальнені ознаки зображень, а донавчання дозволяє адаптувати її до предметної області хвороб рослин.

Реалізація перенесення навчання передбачає два етапи. На першому етапі основна частина попередньо навченої моделі використовується як екстрактор ознак, а навчання виконується переважно для фінального класифікатора. На другому етапі може виконуватися fine-tuning, тобто часткове розмороження останніх шарів моделі та їх донавчання з меншою швидкістю навчання. Це дозволяє підвищити якість прогнозування без суттєвого збільшення ризику перенавчання.

Короткий фрагмент реалізації моделі на основі перенесення навчання наведено на рисунку 3.6.

```
model = models.resnet18(weights=models.ResNet18_Weights.DEFAULT)
model.fc = nn.Linear(model.fc.in_features, num_classes)
```

Рисунок 3.6 – Фрагмент коду, який відображає реалізацію моделі на основі перенесення навчання

У межах програмного модуля вибір моделі здійснюється через параметри конфігурації. Завдяки цьому можна виконувати прогнозування як за допомогою базової CNN, так і за допомогою моделі на основі transfer learning без зміни загальної логіки роботи програмного модуля. Обидві моделі приймають на вхід підготовлений тензор зображення та формують вектор імовірностей належності до класів.

На рисунку 3.7 наведено панель навчання моделей прогнозування, у якій відображено основні показники якості базової CNN та моделі на основі перенесення навчання, а також фрагмент конфігурації експерименту.

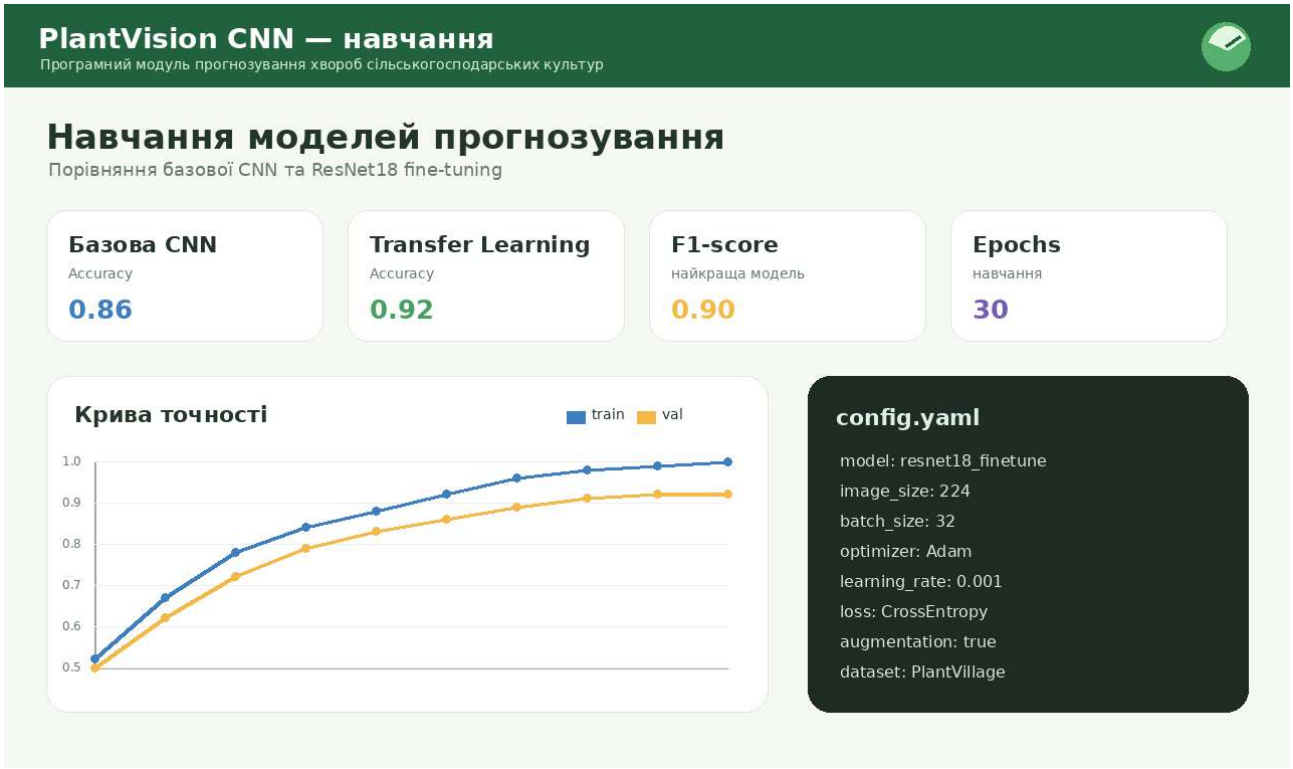

Рисунок 3.7 – Панель навчання моделей прогнозування

Основні параметри навчання моделей представлено в таблиці 3.3.

Таблиця 3.3 – Основні параметри навчання моделей

Параметр	Значення
Набір даних	PlantVillage
Розмір зображення	224×224
Розмір пакета	32
Кількість епох	30
Оптимізатор	Adam
Початкова швидкість навчання	0.001
Функція втрат	Cross-Entropy
Аугментація	Так
Базова модель	CNN
Модель перенесення навчання	ResNet18

Отже, реалізація двох моделей прогнозування забезпечує можливість порівняти просту CNN-архітектуру та підхід на основі перенесення навчання в однакових умовах. Базова CNN є більш простою та зрозумілою для аналізу, тоді як transfer learning дозволяє підвищити якість прогнозування завдяки використанню попередньо навчених ознак.

3.4 Експериментальне дослідження ефективності програмного модуля

Експериментальне дослідження виконано для оцінювання ефективності розробленого програмного модуля прогнозування хвороб сільськогосподарських культур та порівняння двох реалізованих моделей: базової згорткової нейронної мережі та моделі на основі перенесення навчання. Оцінювання проводилося на тестовій вибірці набору PlantVillage, яка не використовувалася під час навчання та валідації моделей.

Для кількісного аналізу якості прогнозування використано стандартні метрики багатокласової класифікації: accuracy, precision, recall та F1-score. Accuracy відображає загальну частку правильно класифікованих зображень, precision характеризує точність визначення відповідного класу, recall показує повноту виявлення прикладів певного класу, а F1-score дає узагальнену оцінку балансу між precision і recall.

Результати експериментального оцінювання якості прогнозування для базової CNN та моделі на основі перенесення навчання подано в таблиці 3.4.

Таблиця 3.4 – Порівняльні показники якості моделей

Модель	Accuracy	Precision	Recall	F1-score
Базова CNN	0.86	0.85	0.84	0.84
Transfer Learning	0.92	0.91	0.90	0.90

Аналіз таблиці 3.4 показує, що модель на основі перенесення навчання демонструє вищі значення всіх основних метрик порівняно з базовою CNN.

Accuracy для transfer learning становить 0.92, тоді як для базової CNN – 0.86. Це свідчить про те, що використання попередньо навчених ознак підвищує якість класифікації зображень рослин. Водночас базова CNN також забезпечує прийнятні результати, що підтверджує коректність спроєктованої архітектури.

Детальні результати класифікації за окремими класами наведено в таблиці 3.5.

Таблиця 3.5 – Точність прогнозування за окремими класами

Клас	Precision	Recall
Здорова рослина	0.95	0.96
Рання плямистість	0.89	0.87
Фітофтороз	0.86	0.84

З таблиці 3.5 видно, що найкращі результати отримано для класу здорових рослин. Це можна пояснити тим, що здорові зразки мають більш однорідні візуальні ознаки порівняно із захворюваннями. Для класів “Рання плямистість” і “Фітофтороз” значення precision та recall є дещо нижчими, що пов’язано зі схожістю зовнішніх симптомів окремих хвороб листя.

На рисунку 3.8 наведено результати експериментального оцінювання моделей, зокрема порівняння основних метрик та матрицю помилок для найкращої моделі.

Матриця помилок дозволяє проаналізувати розподіл правильних і помилкових прогнозів за окремими класами. Найбільша кількість правильних класифікацій спостерігається для класу “Здорова рослина”, тоді як помилки між класами “Рання плямистість” і “Фітофтороз” виникають частіше. Це свідчить про потребу у подальшому розширенні навчальної вибірки та використанні більш різноманітних зображень хвороб рослин.

Окремо проаналізовано динаміку навчання моделі. Крива точності, наведена на рисунку 3.7, демонструє поступове зростання значень accuracy на навчальній та валідаційній вибірках із подальшою стабілізацією. Відсутність

значного розриву між тренувальною і валідаційною точністю свідчить про збалансований процес навчання та відсутність явних ознак перенавчання.

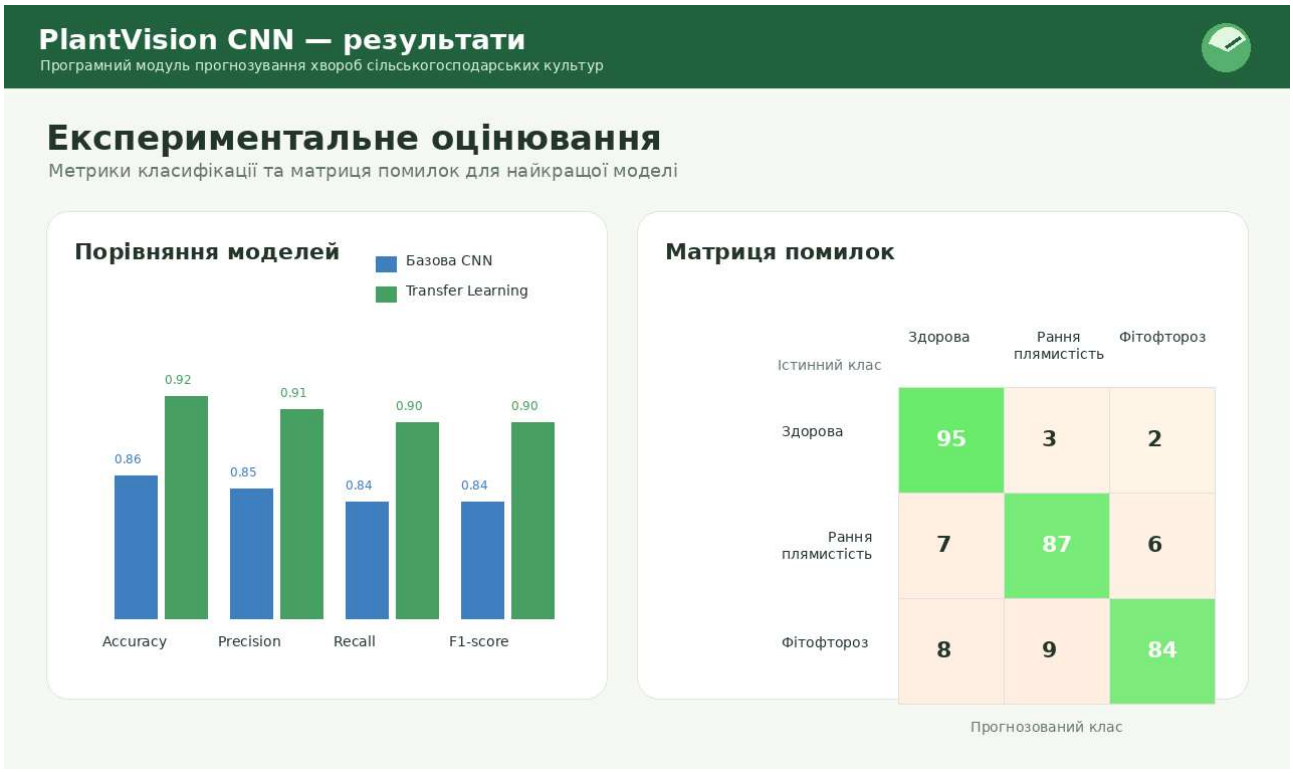

Рисунок 3.8 – Результати експериментального оцінювання моделей

Загалом результати експериментальних досліджень підтвердили ефективність розробленого програмного модуля. Модель на основі перенесення навчання показала кращу якість прогнозування, тоді як базова CNN може розглядатися як простіше та менш ресурсоємне рішення. Отримані результати свідчать про доцільність використання згорткових нейронних мереж для автоматизованого прогнозування хвороб сільськогосподарських культур.

ВИСНОВКИ

У ході виконання кваліфікаційної роботи досягнуто поставленої мети – розроблено програмний модуль прогнозування хвороб сільськогосподарських культур на основі згорткових нейронних мереж, який забезпечує автоматизований аналіз зображень рослин та формування прогнозу щодо їхнього стану.

1. У процесі дослідження проаналізовано особливості задачі прогнозування хвороб рослин та специфіку вхідних даних. Встановлено, що основними ускладнювальними чинниками є висока варіативність зображень за умовами освітлення та фону, зміна візуальних симптомів захворювань на різних стадіях розвитку рослин, а також можливий дисбаланс між класами. Зазначені особливості обумовлюють доцільність застосування методів глибокого навчання, здатних автоматично виділяти інформативні ознаки зображень.

2. Виконано огляд і порівняльний аналіз існуючих методів та архітектур згорткових нейронних мереж, що застосовуються у задачах агровізії. Показано, що згорткові нейронні мережі перевершують класичні підходи за здатністю до узагальнення та точністю прогнозування, а використання глибших архітектур і перенесення навчання дозволяє підвищити ефективність моделей за умов обмежених навчальних даних.

3. Спроєктовано архітектуру програмного модуля прогнозування, яка передбачає поділ на взаємопов’язані компоненти введення даних, попередньої обробки зображень, прогнозування та формування результатів. Запропонована модульна структура забезпечує зрозумілість архітектури, гнучкість та можливість подальшого розширення функціональності.

4. Реалізовано базову згорткову нейронну мережу та модель на основі перенесення навчання у межах єдиного програмного рішення, що дозволило здійснювати порівняльний аналіз різних підходів без зміни загальної логіки роботи модуля. Для моделі з перенесенням навчання організовано процес донавчання у режимі fine-tuning, який передбачає поетапне навчання класифікаційної частини та часткове розмороження згорткових шарів.

5. Проведено експериментальне дослідження ефективності реалізованих моделей з використанням стандартних метрик оцінювання якості класифікації. Отримані результати засвідчили, що модель на основі перенесення навчання демонструє вищі показники точності та узагальнювальної здатності порівняно з базовою згортковою нейронною мережею, особливо за умов обмеженого обсягу навчальних даних.

6. Здійснено аналіз отриманих результатів, який показав, що застосування fine-tuning сприяє зменшенню перенавчання та підвищенню стабільності процесу навчання. Аналіз матриці помилок дозволив виявити класи захворювань, розпізнавання яких є найбільш складним, що окреслює напрями подальшого вдосконалення моделей.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Barbedo J. G. A. Impact of dataset size and variety on the effectiveness of deep learning and transfer learning for plant disease classification. Computers and Electronics in Agriculture. 2018. Vol. 153. Pp. 46–53.

2. Beutel D. J., Topal T., Mathur A., et al. Flower: A Friendly Federated Learning Framework. arXiv. 2020. arXiv:2007.14390. doi:10.48550/arXiv.2007.14390.

3. Bonawitz K., Ivanov V., Kreuter B., et al. Practical Secure Aggregation for Privacy-Preserving Machine Learning. In: Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security (CCS ’17). 2017. Pp. 1175–1191.

4. Deng J., Dong W., Socher R., Li L.-J., Li K., Fei-Fei L. ImageNet: A Large-Scale Hierarchical Image Database. In: Proceedings of CVPR 2009. 2009. doi:10.1109/CVPR.2009.5206848.

5. Ferentinos K. P. Deep learning models for plant disease detection and diagnosis. Computers and Electronics in Agriculture. 2018. Vol. 145. Pp. 311–318.

6. Fedasyuk D. V., Dorofeev V. I. Detection of grape diseases based on deep learning methods. Ukrainian Journal of Information Technology. 2025. Vol. 7, No. 1. Pp. 77–85.

7. Fuentes A., Yoon S., Kim S. C., Park D. S. A Robust Deep-Learning-Based Detector for Real-Time Tomato Plant Diseases and Pests Recognition. Sensors. 2017. Vol. 17(9). Article 2022. doi:10.3390/s17092022.

8. He K., Zhang X., Ren S., Sun J. Deep Residual Learning for Image Recognition. In: Proceedings of CVPR 2016. 2016. Pp. 770–778.

9. Hsu T., Qi H., Brown M. Measuring the Effects of Non-Identical Data Distribution for Federated Visual Classification. arXiv. 2019. arXiv:1909.06335. doi:10.48550/arXiv.1909.06335.

10. Hughes D. P., Salathé M. An open access repository of images on plant health to enable the development of mobile disease diagnostics. arXiv. 2015. arXiv:1511.08060. doi:10.48550/arXiv.1511.08060.

11. Ioffe S., Szegedy C. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. arXiv. 2015. arXiv:1502.03167. doi:10.48550/arXiv.1502.03167.

12. Kairouz P., McMahan H. B., Avent B., et al. Advances and Open Problems in Federated Learning. Foundations and Trends® in Machine Learning. 2021. Vol. 14(1–2). Pp. 1–210. doi:10.1561/2200000083. arXiv

13. Kingma D. P., Ba J. Adam: A Method for Stochastic Optimization. arXiv. 2014. arXiv:1412.6980. doi:10.48550/arXiv.1412.6980.

14. Krizhevsky A., Sutskever I., Hinton G. E. ImageNet Classification with Deep Convolutional Neural Networks. In: Advances in Neural Information Processing Systems (NeurIPS 2012). 2012. Pp. 1097–1105.

15. Li T., Sahu A. K., Talwalkar A., Smith V. Federated Optimization in Heterogeneous Networks (FedProx). arXiv. 2018. arXiv:1812.06127. doi:10.48550/arXiv.1812.06127.

16. Li X., Jiang M., Zhang X., Kamp M., Dou Q. FedBN: Federated Learning on Non-IID Features via Local Batch Normalization. arXiv. 2021. arXiv:2102.07623. doi:10.48550/arXiv.2102.07623.

17. McMahan H. B., Moore E., Ramage D., Hampson S., y Arcas B. A. Communication-Efficient Learning of Deep Networks from Decentralized Data (FedAvg). In: Proceedings of AISTATS 2017. 2017. Pp. 1273–1282.

18. Mohanty S. P., Hughes D. P., Salathé M. Using Deep Learning for Image-Based Plant Disease Detection. Frontiers in Plant Science. 2016. Vol. 7. Article 1419.

19. Paszke A., Gross S., Massa F., et al. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In: Advances in Neural Information Processing Systems (NeurIPS 2019). 2019.

20. Ramcharan A., Baranowski K., McCloskey P., Ahmed B., Legg J., Hughes D. P. Deep learning for image-based cassava disease detection. Frontiers in Plant Science. 2017. Vol. 8. Article 1852.

21. Reddi S. J., Charles Z., Zaheer M., et al. Adaptive Federated Optimization (FedOpt). arXiv. 2020. arXiv:2003.00295. doi:10.48550/arXiv.2003.00295.

22. Selvaraju R. R., Cogswell M., Das A., et al. Grad-CAM: Visual Explanations from Deep Networks via Gradient-based Localization. In: Proceedings of ICCV 2017. 2017. Pp. 618–626.

23. Shorten C., Khoshgoftaar T. M. A survey on Image Data Augmentation for Deep Learning. Journal of Big Data. 2019. Vol. 6. Article 60.

24. Simonyan K., Zisserman A. Very Deep Convolutional Networks for Large-Scale Image Recognition. arXiv. 2014. arXiv:1409.1556. doi:10.48550/arXiv.1409.1556.

25. Singh D., Jain N., Jain P., Kayal P., Kumawat S., Batra N. PlantDoc: A Dataset for Visual Plant Disease Detection. arXiv. 2019. arXiv:1911.10317. doi:10.48550/arXiv.1911.10317.

26. Tan M., Le Q. V. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. arXiv. 2019. arXiv:1905.11946. doi:10.48550/arXiv.1905.11946.

27. Too E. C., Yujian L., Njuki S., Yingchun L. A comparative study of fine-tuning deep learning models for plant disease identification. Computers and Electronics in Agriculture. 2019. Vol. 161. Pp. 272–279.

28. Афонін А. О., Кундік К. В. Використання нейронних мереж у розпізнаванні хвороб рослин. Наукові записки НаУКМА. Комп’ютерні науки. 2021. Т. 4. С. 23–28.

29. Гуменюк Р., Попович І. Дослідження методів діагностики захворювань рослин за допомогою глибокого навчання. Комп’ютерні системи проектування. Теорія і практика. 2024. Т. 6, № 1. С. 37–48.

30. Вороняк Б.П. Прогнозування хвороб сільськогосподарських культур на основі згорткових нейронних мереж. Інтелектуальні комп’ютерні системи та мережі» (ІКСМ) : матеріали IV Всеукраїнської науково-практичної конференції молодих вчених і студентів (м. Тернопіль, ЗУНУ, 20 травня 2026 р.). Тернопіль, 2026. С. 133-136.

31. PlantVillage Dataset. URL: https://www.kaggle.com/datasets/emmarex/plantdisease /

32. ДСТУ ISO/IEC 25010:2016 (ISO/IEC 25010:2011, IDT). Інженерія

систем і програмних засобів. Вимоги до якості систем і програмних засобів та її оцінювання (SQuaRE). Моделі якості систем та програмних засобів. [Чинний від 2018-01-01]. Київ : УкрНДНЦ, 2018. 32 с.

33. ДСТУ 8302:2015. Інформація та документація. Бібліографічне посилання. Загальні положення та правила складання / Нац. стандарт України. Вид. офіц. [Чинний від 2016-07-01]. Київ : ДП «УкрНДНЦ», 2016. 17 с.

34. Комар М.П., Саченко А.О., Васильків Н.М., Гладій Г.М., Коваль В.С., Ліп’яніна-Гончаренко Х.В. Методичні рекомендації до виконання кваліфікаційної роботи з освітньо-професійної програми «Комп’ютерні науки» спеціальності 122 «Комп’ютерні науки» за першим (бакалаврським) рівнем вищої освіти. Тернопіль: ЗУНУ, 2024. 52 с.

35. Островерхов В. М., Біловус Л. І., Возьний К. З., Луцишин О. О., Монастирський Г. Л., Надвиничний С. А., Питель С. В., Шандрук С. К. Загальні методичні рекомендації з підготовки, оформлення, захисту та оцінювання кваліфікаційних робіт здобувачів вищої освіти першого (бакалаврського) і другого (магістерського) рівнів. Тернопіль: ЗУНУ, 2024. 83 с.

Додаток А

Програмні коди проєкту

Лістинг А.1 – config/settings.py – налаштування проєкту

```
from dataclasses import dataclass
from typing import List

@dataclass
class Settings:
    image_size: int = 224
    device: str = "cuda"

    dataset_name: str = "PlantVillage"
    model_type: str = "transfer"  # "base" або "transfer"

    base_cnn_weights: str = "checkpoints/base_cnn.pth"
    transfer_weights: str = "checkpoints/transfer_finetuned.pth"

    output_path: str = "outputs/result.json"

    class_names: List[str] = None

    def __post_init__(self):
        if self.class_names is None:
            self.class_names = [
                "healthy",
                "early_blight",
                "late_blight"
            ]
```

Лістинг А.2 – data/io.py – завантаження зображення

```
from pathlib import Path
from PIL import Image

SUPPORTED_EXTENSIONS = {".jpg", ".jpeg", ".png"}

def load_image(image_path: str) -> Image.Image:
    path = Path(image_path)

    if not path.exists() or not path.is_file():
        raise FileNotFoundError(f"Файл не знайдено: {image_path}")

    if path.suffix.lower() not in SUPPORTED_EXTENSIONS:
        raise ValueError(f"Непідтримуваний формат файлу: {path.suffix}")

    try:
        image = Image.open(path).convert("RGB")
    except Exception as exc:
        raise ValueError(f"Не вдалося відкрити зображення: {image_path}") from
exc

    return image
```

Лістинг А.3 – data/preprocess.py – попередня обробка зображення

```
import torch
from torchvision import transforms
```

```
from PIL import Image
from config.settings import Settings

def build_inference_transform(cfg: Settings) -> transforms.Compose:
    return transforms.Compose([
        transforms.Resize((cfg.image_size, cfg.image_size)),
        transforms.ToTensor(),
        transforms.Normalize(
            mean=[0.485, 0.456, 0.406],
            std=[0.229, 0.224, 0.225]
        )
    ])

def preprocess_image(image: Image.Image, cfg: Settings) -> torch.Tensor:
    transform = build_inference_transform(cfg)
    tensor = transform(image)
    tensor = tensor.unsqueeze(0)
    return tensor
```

Лістинг А.4 – models/base_cnn.py – реалізація базової CNN-моделі

```
import torch
import torch.nn as nn
import torch.nn.functional as F

class SimpleCNN(nn.Module):
    def __init__(self, num_classes: int):
        super().__init__()

        self.conv1 = nn.Conv2d(
            in_channels=3,
            out_channels=32,
            kernel_size=3,
            padding=1
        )

        self.conv2 = nn.Conv2d(
            in_channels=32,
            out_channels=64,
            kernel_size=3,
            padding=1
        )

        self.pool = nn.MaxPool2d(kernel_size=2, stride=2)
        self.bn = nn.BatchNorm2d(64)

        self.fc1 = nn.Linear(64 * 56 * 56, 128)
        self.dropout = nn.Dropout(p=0.5)
        self.fc2 = nn.Linear(128, num_classes)

    def forward(self, x: torch.Tensor) -> torch.Tensor:
        x = self.pool(F.relu(self.conv1(x)))
        x = self.pool(F.relu(self.conv2(x)))

        x = self.bn(x)
        x = x.flatten(1)

        x = F.relu(self.fc1(x))
        x = self.dropout(x)

        return self.fc2(x)
```

Лістинг А.5 – models/transfer.py – реалізація моделі на основі перенесення навчання

```
import torch.nn as nn
from torchvision import models

def build_resnet18_transfer(
    num_classes: int,
    freeze_backbone: bool = True
):
    model = models.resnet18(
        weights=models.ResNet18_Weights.DEFAULT
    )

    if freeze_backbone:
        for parameter in model.parameters():
            parameter.requires_grad = False

    in_features = model.fc.in_features
    model.fc = nn.Linear(in_features, num_classes)

    return model
```

Лістинг А.6 – models/loader.py – завантаження моделі прогнозування

```
import torch
from config.settings import Settings
from models.base_cnn import SimpleCNN
from models.transfer import build_resnet18_transfer

def load_model(cfg: Settings) -> torch.nn.Module:
    num_classes = len(cfg.class_names)

    if cfg.model_type == "transfer":
        model = build_resnet18_transfer(
            num_classes=num_classes,
            freeze_backbone=True
        )
        weights_path = cfg.transfer_weights
    else:
        model = SimpleCNN(num_classes=num_classes)
        weights_path = cfg.base_cnn_weights

    try:
        state = torch.load(weights_path, map_location=cfg.device)
        model.load_state_dict(state)
    except FileNotFoundError:
        print("Файл ваг моделі не знайдено. Використано ініціалізовану модель.")

    model.to(cfg.device)
    model.eval()

    return model
```

Лістинг А.7 – inference/predict.py – виконання прогнозування

```
import torch
import torch.nn.functional as F
from typing import Dict, Any
from config.settings import Settings

@torch.no_grad()
```

```
def predict(
    tensor: torch.Tensor,
    model: torch.nn.Module,
    cfg: Settings
) -> Dict[str, Any]:
    tensor = tensor.to(cfg.device)

    logits = model(tensor)
    probabilities = F.softmax(logits, dim=1).squeeze(0)

    confidence, class_index = torch.max(probabilities, dim=0)
    class_index = int(class_index)

    return {
        "predicted_class": cfg.class_names[class_index],
        "confidence": float(confidence),
        "probabilities": {
            cfg.class_names[i]: float(probabilities[i])
            for i in range(len(cfg.class_names))
        }
    }
```

Лістинг А.8 – utils/report.py – збереження результатів прогнозування

```
import json
from pathlib import Path
from typing import Dict, Any

def save_result_json(result: Dict[str, Any], output_path: str) -> None:
    output_file = Path(output_path)
    output_file.parent.mkdir(parents=True, exist_ok=True)

    with output_file.open("w", encoding="utf-8") as file:
        json.dump(
            result,
            file,
            ensure_ascii=False,
            indent=2
        )
```

Лістинг А.9 – app.py – керуючий модуль запуску програмного рішення

```
from config.settings import Settings
from data.io import load_image
from data.preprocess import preprocess_image
from models.loader import load_model
from inference.predict import predict
from utils.report import save_result_json

def run(
    image_path: str = "samples/tomato_leaf.png"
) -> None:
    cfg = Settings()

    model = load_model(cfg)

    image = load_image(image_path)
    tensor = preprocess_image(image, cfg)

    result = predict(tensor, model, cfg)

    save_result_json(result, cfg.output_path)
```

```
    print("Клас:", result["predicted_class"])
    print("Впевненість:", result["confidence"])
    print("Результат збережено:", cfg.output_path)

if __name__ == "__main__":
    run("samples/tomato_leaf.png")
```

Лістинг А.10 – приклад результату у файлі outputs/result.json

```
{
  "predicted_class": "late_blight",
  "confidence": 0.92,
  "probabilities": {
    "healthy": 0.04,
    "early_blight": 0.04,
    "late_blight": 0.92
  }
}
```

Додаток Б

Скрипти навчання та тестування моделей

Лістинг Б.1 – train_finetune.py – навчання моделі у режимі fine-tuning

```
import os
from dataclasses import dataclass
from typing import Tuple

import torch
import torch.nn as nn
from torch.utils.data import DataLoader
from torchvision import datasets, transforms, models


@dataclass
class TrainConfig:
    dataset_name: str = "PlantVillage"
    data_dir: str = "dataset"
    image_size: int = 224
    batch_size: int = 32
    epochs_head: int = 5
    epochs_finetune: int = 25
    learning_rate_head: float = 0.001
    learning_rate_finetune: float = 0.0001
    num_workers: int = 2
    checkpoint_dir: str = "checkpoints"
    output_model: str = "transfer_finetuned.pth"
    device: str = "cuda" if torch.cuda.is_available() else "cpu"


def build_transforms(cfg: TrainConfig) -> Tuple[transforms.Compose,
transforms.Compose]:
    train_transform = transforms.Compose([
        transforms.Resize((cfg.image_size, cfg.image_size)),
        transforms.RandomHorizontalFlip(p=0.5),
        transforms.RandomRotation(degrees=10),
        transforms.ColorJitter(
            brightness=0.2,
            contrast=0.2,
            saturation=0.2
        ),
        transforms.ToTensor(),
        transforms.Normalize(
            mean=[0.485, 0.456, 0.406],
            std=[0.229, 0.224, 0.225]
        )
    ])

    test_transform = transforms.Compose([
        transforms.Resize((cfg.image_size, cfg.image_size)),
        transforms.ToTensor(),
        transforms.Normalize(
            mean=[0.485, 0.456, 0.406],
            std=[0.229, 0.224, 0.225]
        )
    ])

    return train_transform, test_transform


def build_dataloaders(cfg: TrainConfig):
```

```
    train_transform, test_transform = build_transforms(cfg)

    train_dataset = datasets.ImageFolder(
        os.path.join(cfg.data_dir, "train"),
        transform=train_transform
    )

    val_dataset = datasets.ImageFolder(
        os.path.join(cfg.data_dir, "val"),
        transform=test_transform
    )

    train_loader = DataLoader(
        train_dataset,
        batch_size=cfg.batch_size,
        shuffle=True,
        num_workers=cfg.num_workers
    )

    val_loader = DataLoader(
        val_dataset,
        batch_size=cfg.batch_size,
        shuffle=False,
        num_workers=cfg.num_workers
    )

    return train_dataset, val_dataset, train_loader, val_loader


def build_model(num_classes: int) -> nn.Module:
    model = models.resnet18(weights=models.ResNet18_Weights.DEFAULT)
    in_features = model.fc.in_features
    model.fc = nn.Linear(in_features, num_classes)
    return model


def freeze_backbone(model: nn.Module) -> None:
    for parameter in model.parameters():
        parameter.requires_grad = False

    for parameter in model.fc.parameters():
        parameter.requires_grad = True


def unfreeze_last_blocks(model: nn.Module) -> None:
    for parameter in model.parameters():
        parameter.requires_grad = False

    for parameter in model.layer4.parameters():
        parameter.requires_grad = True

    for parameter in model.fc.parameters():
        parameter.requires_grad = True


def train_one_epoch(
    model: nn.Module,
    loader: DataLoader,
    criterion,
    optimizer,
    device: str
) -> float:
    model.train()
    total_loss = 0.0

    for images, labels in loader:
```

```
        images = images.to(device)
        labels = labels.to(device)

        optimizer.zero_grad()
        outputs = model(images)
        loss = criterion(outputs, labels)

        loss.backward()
        optimizer.step()

        total_loss += loss.item() * images.size(0)

    return total_loss / len(loader.dataset)


@torch.no_grad()
def evaluate_accuracy(
    model: nn.Module,
    loader: DataLoader,
    device: str
) -> float:
    model.eval()
    correct = 0
    total = 0

    for images, labels in loader:
        images = images.to(device)
        labels = labels.to(device)

        outputs = model(images)
        predicted = torch.argmax(outputs, dim=1)

        correct += (predicted == labels).sum().item()
        total += labels.size(0)

    return correct / total


def main():
    cfg = TrainConfig()
    os.makedirs(cfg.checkpoint_dir, exist_ok=True)

    train_dataset, val_dataset, train_loader, val_loader =
build_dataloaders(cfg)
    num_classes = len(train_dataset.classes)

    model = build_model(num_classes)
    model.to(cfg.device)

    criterion = nn.CrossEntropyLoss()

    best_val_accuracy = 0.0
    log_path = os.path.join(cfg.checkpoint_dir, "train_log.csv")

    with open(log_path, "w", encoding="utf-8") as log_file:
        log_file.write("stage,epoch,train_loss,val_accuracy\n")

    freeze_backbone(model)

    optimizer = torch.optim.Adam(
        filter(lambda p: p.requires_grad, model.parameters()),
        lr=cfg.learning_rate_head
    )

    for epoch in range(cfg.epochs_head):
        train_loss = train_one_epoch(
```

```
            model,
            train_loader,
            criterion,
            optimizer,
            cfg.device
        )

        val_accuracy = evaluate_accuracy(model, val_loader, cfg.device)

        with open(log_path, "a", encoding="utf-8") as log_file:
            log_file.write(f"head,{epoch +
1},{train_loss:.4f},{val_accuracy:.4f}\n")

        if val_accuracy > best_val_accuracy:
            best_val_accuracy = val_accuracy
            torch.save(
                model.state_dict(),
                os.path.join(cfg.checkpoint_dir, cfg.output_model)
            )

    unfreeze_last_blocks(model)

    optimizer = torch.optim.Adam(
        filter(lambda p: p.requires_grad, model.parameters()),
        lr=cfg.learning_rate_finetune
    )

    for epoch in range(cfg.epochs_finetune):
        train_loss = train_one_epoch(
            model,
            train_loader,
            criterion,
            optimizer,
            cfg.device
        )

        val_accuracy = evaluate_accuracy(model, val_loader, cfg.device)

        with open(log_path, "a", encoding="utf-8") as log_file:
            log_file.write(f"finetune,{epoch +
1},{train_loss:.4f},{val_accuracy:.4f}\n")

        if val_accuracy > best_val_accuracy:
            best_val_accuracy = val_accuracy
            torch.save(
                model.state_dict(),
                os.path.join(cfg.checkpoint_dir, cfg.output_model)
            )

    print("Навчання завершено")
    print("Найкраща точність на валідації:", best_val_accuracy)


if __name__ == "__main__":
    main()
```

Лістинг Б.2 – evaluate_model.py – тестування моделі та формування метрик

```
import os
import csv
from dataclasses import dataclass

import torch
import torch.nn as nn
```

```
from torch.utils.data import DataLoader
from torchvision import datasets, transforms, models

import numpy as np


@dataclass
class EvalConfig:
    dataset_name: str = "PlantVillage"
    data_dir: str = "dataset"
    image_size: int = 224
    batch_size: int = 32
    num_workers: int = 2
    checkpoint_path: str = "checkpoints/transfer_finetuned.pth"
    output_dir: str = "outputs"
    device: str = "cuda" if torch.cuda.is_available() else "cpu"


def build_transform(cfg: EvalConfig) -> transforms.Compose:
    return transforms.Compose([
        transforms.Resize((cfg.image_size, cfg.image_size)),
        transforms.ToTensor(),
        transforms.Normalize(
            mean=[0.485, 0.456, 0.406],
            std=[0.229, 0.224, 0.225]
        )
    ])


def build_model(num_classes: int) -> nn.Module:
    model = models.resnet18(weights=None)
    in_features = model.fc.in_features
    model.fc = nn.Linear(in_features, num_classes)
    return model


@torch.no_grad()
def predict_all(model, loader, device):
    model.eval()

    true_labels = []
    predicted_labels = []

    for images, labels in loader:
        images = images.to(device)

        outputs = model(images)
        predictions = torch.argmax(outputs, dim=1).cpu().numpy()

        true_labels.extend(labels.numpy())
        predicted_labels.extend(predictions)

    return np.array(true_labels), np.array(predicted_labels)


def build_confusion_matrix(y_true, y_pred, num_classes):
    matrix = np.zeros((num_classes, num_classes), dtype=int)

    for true, pred in zip(y_true, y_pred):
        matrix[true][pred] += 1

    return matrix


def calculate_metrics(confusion_matrix):
    tp = np.diag(confusion_matrix).astype(float)
```

```
    fp = confusion_matrix.sum(axis=0) - tp
    fn = confusion_matrix.sum(axis=1) - tp

    precision = np.divide(
        tp,
        tp + fp,
        out=np.zeros_like(tp),
        where=(tp + fp) != 0
    )

    recall = np.divide(
        tp,
        tp + fn,
        out=np.zeros_like(tp),
        where=(tp + fn) != 0
    )

    f1_score = np.divide(
        2 * precision * recall,
        precision + recall,
        out=np.zeros_like(tp),
        where=(precision + recall) != 0
    )

    return precision, recall, f1_score


def main():
    cfg = EvalConfig()
    os.makedirs(cfg.output_dir, exist_ok=True)

    transform = build_transform(cfg)

    test_dataset = datasets.ImageFolder(
        os.path.join(cfg.data_dir, "test"),
        transform=transform
    )

    test_loader = DataLoader(
        test_dataset,
        batch_size=cfg.batch_size,
        shuffle=False,
        num_workers=cfg.num_workers
    )

    model = build_model(num_classes=len(test_dataset.classes))
    state = torch.load(cfg.checkpoint_path, map_location=cfg.device)
    model.load_state_dict(state)
    model.to(cfg.device)

    y_true, y_pred = predict_all(model, test_loader, cfg.device)

    accuracy = float((y_true == y_pred).mean())

    cm = build_confusion_matrix(
        y_true,
        y_pred,
        num_classes=len(test_dataset.classes)
    )

    precision, recall, f1_score = calculate_metrics(cm)

    summary_path = os.path.join(cfg.output_dir, "metrics_summary.csv")
    class_path = os.path.join(cfg.output_dir, "metrics_per_class.csv")

    with open(summary_path, "w", newline="", encoding="utf-8") as file:
```

```
        writer = csv.writer(file)
        writer.writerow(["metric", "value"])
        writer.writerow(["accuracy", f"{accuracy:.4f}"])
        writer.writerow(["precision_macro", f"{precision.mean():.4f}"])
        writer.writerow(["recall_macro", f"{recall.mean():.4f}"])
        writer.writerow(["f1_macro", f"{f1_score.mean():.4f}"])

    with open(class_path, "w", newline="", encoding="utf-8") as file:
        writer = csv.writer(file)
        writer.writerow(["class", "precision", "recall", "f1_score"])

        for index, class_name in enumerate(test_dataset.classes):
            writer.writerow([
                class_name,
                f"{precision[index]:.4f}",
                f"{recall[index]:.4f}",
                f"{f1_score[index]:.4f}"
            ])

    print("Accuracy:", accuracy)
    print("Звіт збережено:", summary_path)
    print("Метрики за класами збережено:", class_path)


if __name__ == "__main__":
    main()
```

Лістинг Б.3 – plot_results.py – побудова графіків для експериментального аналізу

```
import os
import csv
from dataclasses import dataclass

import matplotlib.pyplot as plt
import numpy as np


@dataclass
class PlotConfig:
    log_path: str = "checkpoints/train_log.csv"
    output_dir: str = "outputs"


def plot_accuracy_curve(cfg: PlotConfig):
    val_accuracy = []

    with open(cfg.log_path, "r", encoding="utf-8") as file:
        reader = csv.DictReader(file)

        for row in reader:
            val_accuracy.append(float(row["val_accuracy"]))

    epochs = range(1, len(val_accuracy) + 1)

    plt.figure()
    plt.plot(epochs, val_accuracy, marker="o")
    plt.xlabel("Епоха")
    plt.ylabel("Точність на валідаційній вибірці")
    plt.title("Зміна точності моделі під час навчання")
    plt.grid(True)

    output_path = os.path.join(cfg.output_dir, "accuracy_curve.png")
    plt.savefig(output_path, dpi=200)
    plt.close()
```

```
def plot_confusion_matrix(matrix, class_names, output_path):
    plt.figure()
    plt.imshow(matrix)

    plt.xticks(
        ticks=np.arange(len(class_names)),
        labels=class_names,
        rotation=45,
        ha="right"
    )

    plt.yticks(
        ticks=np.arange(len(class_names)),
        labels=class_names
    )

    plt.xlabel("Прогнозований клас")
    plt.ylabel("Істинний клас")
    plt.title("Матриця помилок")

    for i in range(matrix.shape[0]):
        for j in range(matrix.shape[1]):
            plt.text(j, i, str(matrix[i, j]), ha="center", va="center")

    plt.tight_layout()
    plt.savefig(output_path, dpi=200)
    plt.close()
```

Лістинг Б.4 – приклад структури каталогу набору PlantVillage

```
dataset/
├── train/
│   ├── healthy/
│   ├── early_blight/
│   └── late_blight/
├── val/
│   ├── healthy/
│   ├── early_blight/
│   └── late_blight/
└── test/
    ├── healthy/
    ├── early_blight/
    └── late_blight/
```

Лістинг Б.5 – приклад вихідного файлу metrics_summary.csv

```
metric,value
accuracy,0.9200
precision_macro,0.9100
recall_macro,0.9000
f1_macro,0.9000
```

Лістинг Б.6 – приклад вихідного файлу metrics_per_class.csv

```
class,precision,recall,f1_score
healthy,0.9500,0.9600,0.9550
early_blight,0.8900,0.8700,0.8799
late_blight,0.8600,0.8400,0.8499
```

Додаток В

Конфігурації та параметри експериментів

Лістинг В.1 – configs/experiment.yaml – конфігурація експерименту

```
experiment:
  name: "plant_disease_prediction"
  description: "Прогнозування хвороб сільськогосподарських культур на основі
CNN"
  seed: 42
  device: "cuda"        # "cuda" або "cpu"

dataset:
  name: "PlantVillage"
  root_dir: "dataset"
  classes:
    - healthy
    - early_blight
    - late_blight
  split:
    train: 0.70
    val: 0.15
    test: 0.15

image:
  size: 224
  channels: 3
  color_space: "RGB"

preprocessing:
  resize: true
  normalize: true
  normalization:
    mean: [0.485, 0.456, 0.406]
    std:  [0.229, 0.224, 0.225]
  to_tensor: true
  shuffle: true

augmentation:
  enabled: true
  horizontal_flip: 0.5
  rotation_degrees: 10
  brightness: 0.2
  contrast: 0.2
  saturation: 0.2

base_cnn:
  enabled: true
  input_shape: [3, 224, 224]
  conv_layers:
    - filters: 32
      kernel_size: 3
      activation: "ReLU"
      pooling: "MaxPooling2D"
    - filters: 64
      kernel_size: 3
      activation: "ReLU"
      pooling: "MaxPooling2D"
  batch_normalization: true
  dropout: 0.5
  dense_units: 128
  output_activation: "Softmax"
```

```
transfer_learning:
  enabled: true
  backbone: "ResNet18"
  pretrained_weights: "ImageNet"
  mode: "fine-tuning"
  freeze_backbone_stage1: true
  unfreeze_last_block_stage2: true
  classifier_replacement: true

training:
  batch_size: 32
  epochs: 30
  optimizer: "Adam"
  learning_rate: 0.001
  fine_tuning_learning_rate: 0.0001
  loss_function: "CrossEntropyLoss"
  checkpoint_dir: "checkpoints"
  save_best_model: true
  save_best_by: "val_accuracy"

evaluation:
  metrics:
    - accuracy
    - precision
    - recall
    - f1_score
  test_subset: "test"
  output_dir: "outputs"
  summary_file: "outputs/metrics_summary.csv"
  per_class_file: "outputs/metrics_per_class.csv"

visualization:
  accuracy_curve: "outputs/accuracy_curve.png"
  confusion_matrix: "outputs/confusion_matrix.png"
  dashboard_screenshot: "screenshot_3_4_results_metrics.png"
```

Лістинг В.2 – приклад файлу outputs/metrics_summary.csv

```
metric,value
accuracy,0.9200
precision_macro,0.9100
recall_macro,0.9000
f1_macro,0.9000
```

Лістинг В.3 – приклад файлу outputs/metrics_per_class.csv

```
class,precision,recall,f1_score
healthy,0.9500,0.9600,0.9550
early_blight,0.8900,0.8700,0.8799
late_blight,0.8600,0.8400,0.8499
```

Лістинг В.4 – приклад файлу outputs/result.json

```
{
  "predicted_class": "late_blight",
  "confidence": 0.92,
  "probabilities": {
    "healthy": 0.04,
    "early_blight": 0.04,
    "late_blight": 0.92
  }
}
```

Додаток Г

Апробація результатів роботи

Почесні співголови конференції:

Десятнюк О.М., ректор Західноукраїнського національного університету, д-р економічних наук, професор;

Дивак М.П., д-р технічних наук, професор, проректор з наукової роботи ЗУНУ;

Голова конференції:

Березький О.М., д-р технічних наук, професор, професор кафедри комп'ютерної інженерії Західноукраїнський національний університет;

Заступники голови конференції:

Мельник Г.М., к.т.н, доцент, Західноукраїнський національний університет;

Піцун О.Й., к.т.н. доцент, Західноукраїнський національний університет;

ПРОГРАМНИЙ КОМІТЕТ КОНФЕРЕНЦІЇ

Склад програмного комітету:

Семанюк В.З., д-р економічних наук, професор, начальник науково-дослідної частини Західноукраїнського національного університету

Антощук С.Г., д.т.н, професор, Національний університет «Одеська політехніка»;

Баловсяк С.В., д.т.н, професор, Чернівецький національний університет

Бармак О.В., д.т.н., професор, Хмельницький національний університет

Батько Ю.М., к.т.н., доцент, Західноукраїнський національний університет

Березька К.М.,к.т.н.,доцент, Західноукраїнський національний університет

Винокурова О.А., д.т.н., професор, Львівський національний університет імені Івана Франка

Возна Н.Я., д.т.н., професор, Західноукраїнський національний університет;

Говорущенко Т.О., д.т.н., професор, Хмельницький національний університет;

Дубчак Л.О., к.т.н., доцент, Західноукраїнський національний університет;

Дунець Р.Б., д.т.н., професор, НУ "Львівська політехніка";

Ізонін І.В., д.т.н., доцент, НУ "Львівська політехніка";

Комар М.П., д.т.н, професор, Західноукраїнський національний університет;

Литвиненко В.І., д.н.т, професор, Національний технічний університет України "Київський політехнічний інститут";

Лупенко С.А., д.т.н., професор, Опольський технологічний університет, Польща;

Лящинський П.Б., доктор філософії з комп'ютерних наук, НУ "Львівська політехніка" ;

Мельникова Н.І., д.т.н., професор, НУ "Львівська політехніка" ;

Мулеса О.Ю., д.т.н., професор, Пряшівський університет, Словаччина

Пелешко Д.Д., д.т.н., професор, Львівський національний університет імені Івана Франка;

Сельський П.Р., д.м.н., професор, Тернопільський національний медичний університет імені І. Я. Горбачевського ;
Субботін С.О., д.т.н., професор, Національний університет «Запорізька політехніка» ;
Теслюк В.М., д.т.н., професор, НУ "Львівська політехніка" ;
Тимченко Л.І., д.т.н., професор, Державний університет інфраструктури та технологій;
Цмоць І.Г., д.т.н., професор, НУ "Львівська політехніка" ;
Якименко І.З., к.т.н., доцент, Західноукраїнський національний університет;
Яровий А.А., д.т.н., професор, Вінницький національний технічний університет ;
Яцків В.В., д.т.н., професор, Західноукраїнський національний університет.

ОРГАНІЗАЦІЙНИЙ КОМІТЕТ КОНФЕРЕНЦІЇ

Склад організаційного комітету:

Мельник Г.М. к.т.н., доцент, Західноукраїнський національний університет
Піцун О.Й. к.т.н., доцент, Західноукраїнський національний університет

Технічна підтримка:

Зінькевич О. В. студентка, Західноукраїнський національний університет
Кіт М. О. студентка, Західноукраїнський національний університет
Лебединський Т.В. студент,Західноукраїнський національний університет;
Кришталь Е.Р., студент, Західноукраїнський національний університет;

Метою конференції є представлення та обговорення наукових і практичних результатів, сприяння активізації творчої і інноваційної діяльності студентів, аспірантів і молодих вчених.
Конференція проводиться із залученням Ради Молодих Вчених та Студентського Наукового Товариства ФКІТ ЗУНУ.

IV Всеукраїнська науково-практична конференція студентів, аспірантів та молодих вчених «Інтелектуальні комп'ютерні системи та мережі» (ІКСМ весна 2026), м. Тернопіль, ЗУНУ, 29 травня 2026 р. Тернопіль, 2026

Адреса:
Вул. О. Теліги, 8, корпус №6, Тернопіль
URL: https://ki.wunu.edu.ua/conference/
e-mail: kistudconference@gmail.com

Тези доповідей подаються за оригіналом рукопису

Березька К.М. , Ілюшин В. В.
Моделювання та програмна реалізація аналізу транспортних потоків м. Тернополя.....111
Дребот С. Р.
Інтелектуальний помічник інструктора автошколи на основі комп'ютерного зору.....114
Ілюшин В. В., Король В.Р.
Програмний модуль кластеризації пасажиропотоків міста Тернополя.....117
Чаус М.В.
Смарт-система для вивчення жестової мови з адаптивним налаштуванням навчальної програми та використанням технологій комп'ютерного зору і машинного навчання ...120
Ткачук К. І.
Розробка системи виявлення аномалій у поведінкових даних студентів на основі автоенкодера.....123
Боднар А.О.
Веб-система автоматизованого створення рекламного відеоконтенту на основі генеративних ШІ-моделей.....125
Керничний В.Р.
HDL-модель низькочастотної фільтрації зображення.....127
Білокур В.В.
Проєктування нейромережевої моделі виявлення вторгнень 129
Вороняк Б.П.
Прогнозування хвороб сільськогосподарських культур на основі згорткових нейронних мереж.....133
Омельчук О.А.
Інтелектуальне відеоспостереження в режимі реального часу на основі згорткових нейронних мереж та туманних обчислень 137
Литвин А. А.
Методи та засоби підвищення швидкодії згорткових нейронних мереж для систем комп'ютерного зору.....141
Лихтей В.В., Андрійчук Д.Р.
Інтелектуальна система розпізнавання загроз у середовищі «розумного будинку».....143

Вороняк Б.П.
студент 4 курсу ФКІТ ЗУНУ
Науковий керівник д.т.н., професор Комар М.П., кафедра ІОСУ ЗУНУ

ПРОГНОЗУВАННЯ ХВОРОБ СІЛЬСЬКОГОСПОДАРСЬКИХ КУЛЬТУР НА ОСНОВІ ЗГОРТКОВИХ НЕЙРОННИХ МЕРЕЖ

Вступ. Сільське господарство є однією з ключових галузей економіки, стабільне функціонування якої безпосередньо впливає на продовольчу безпеку та соціально-економічний розвиток держави. Однією з основних причин зниження врожайності сільськогосподарських культур є поширення різноманітних хвороб рослин, які за відсутності своєчасної діагностики можуть призводити до значних економічних втрат. У зв'язку з цим актуальним завданням є розроблення ефективних методів раннього прогнозування та виявлення захворювань культур.

Традиційні способи діагностики хвороб рослин базуються переважно на візуальному огляді посівів фахівцями або проведенні лабораторних аналізів. Такі підходи, хоча й забезпечують достатню точність, характеризуються високою трудомісткістю, значними часовими витратами та суб'єктивністю оцінювання. Крім того, їх застосування є ускладненим у випадку великих аграрних площ, що обмежує можливість оперативного реагування на появу захворювань.

Розвиток цифрових технологій та методів штучного інтелекту відкрив нові можливості для автоматизації процесів моніторингу стану рослин. Зокрема, методи комп'ютерного зору у поєднанні з глибоким навчанням дозволяють здійснювати аналіз зображень листя та інших частин рослин з метою виявлення характерних ознак захворювань [1-3]. Серед таких методів особливе місце посідають згорткові нейронні мережі, які здатні автоматично формувати інформативні просторові ознаки та демонструють високі показники точності у задачах класифікації зображень [4-6].

Ефективність застосування згорткових нейронних мереж у задачах прогнозування хвороб рослин значною мірою залежить від якості навчальних даних, вибору архітектури моделі та параметрів навчання. Дослідження свідчать, що використання сучасних архітектур глибокого навчання, а також методів аугментації даних і перенесення навчання, дозволяє підвищити узагальнювальну здатність моделей навіть за обмежених обсягів навчальної вибірки [4, 6]. Це робить такі підходи перспективними для практичного використання в аграрних інформаційних системах.

Постановка задачі. Проведений аналіз предметної області та існуючих підходів до прогнозування хвороб сільськогосподарських культур показав, що згорткові нейронні мережі є одним із найбільш ефективних інструментів для автоматизованого аналізу зображень рослин. Сучасні дослідження підтверджують здатність CNN забезпечувати високі показники точності класифікації за рахунок автоматичного виділення просторових ознак, характерних для різних типів захворювань. Водночас більшість наукових робіт зосереджена переважно на вдосконаленні моделей глибокого навчання, тоді як питання практичної реалізації таких підходів у вигляді завершених програмних модулів залишаються недостатньо опрацьованими.

У реальних умовах аграрного виробництва використання методів глибокого навчання потребує не лише наявності ефективної моделі, а й програмного забезпечення, здатного забезпечити повний цикл обробки даних. Такий цикл охоплює завантаження та попередню обробку зображень, виконання прогнозування на основі згорткової нейронної мережі, а також формування результатів у зрозумілому та зручному для користувача вигляді. Недостатня увага до цих аспектів ускладнює впровадження результатів наукових досліджень у практичну діяльність аграрних підприємств.

Об'єктом дослідження є процес автоматизованої діагностики хвороб рослин за цифровими зображеннями.

Предметом дослідження є методи та моделі глибокого навчання, зокрема згорткові нейронні мережі та підходи перенесення навчання, а також програмні засоби їх реалізації.

Метою роботи є проектування програмного модуля прогнозування хвороб сільськогосподарських культур на основі згорткових нейронних мереж, який забезпечує автоматизований аналіз зображень рослин і формування прогнозу щодо їхнього стану з використанням сучасних методів глибокого навчання.

Основний матеріал. Загальна архітектура програмного модуля має багаторівневу структуру та включає кілька логічно відокремлених компонентів, кожен з яких відповідає за виконання окремих функцій. Основними компонентами системи є модуль введення даних, модуль попередньої обробки зображень, модуль прогнозування на основі згорткової нейронної мережі та модуль формування і представлення результатів.

Модуль введення даних забезпечує завантаження зображень рослин у систему та виконує первинну перевірку їх коректності. На цьому етапі здійснюється перевірка формату файлів, розміру зображень та їх доступності для подальшої обробки. Такий підхід дозволяє зменшити ймовірність помилок на наступних етапах роботи програмного модуля.

Модуль попередньої обробки зображень призначений для підготовки вхідних даних до подачі у згорткову нейронну мережу. До його основних функцій належать масштабування зображень до фіксованого розміру, нормалізація значень пікселів, а також виконання операцій, спрямованих на зменшення впливу шумів та варіацій умов зйомки. Реалізація цього модуля є критичною для забезпечення стабільності та узагальнювальної здатності моделі прогнозування [4].

Ключовим компонентом архітектури є модуль прогнозування, який реалізує згорткову нейронну мережу для класифікації стану рослин. Даний модуль відповідає за завантаження попередньо навченої моделі, обробку підготовлених зображень та обчислення ймовірностей належності до кожного з класів. Використання згорткових нейронних мереж у цьому модулі зумовлене їх здатністю ефективно виявляти просторові ознаки, характерні для різних типів захворювань рослин [1-3].

Модуль формування результатів забезпечує інтерпретацію вихідних даних нейронної мережі та їх представлення у зручному для користувача вигляді. Результати прогнозування можуть відображатися у вигляді текстових повідомлень, таблиць або графічних елементів, що містять інформацію про прогнозований клас і відповідні значення ймовірностей. Такий підхід підвищує наочність результатів і полегшує їх подальший аналіз.

Взаємодія між компонентами програмного модуля відбувається послідовно, відповідно до визначеного потоку обробки даних. Зображення, отримані на вході системи, передаються до модуля попередньої обробки, після чого підготовлені дані надходять до модуля прогнозування. Результати обчислень передаються до модуля формування результатів, який завершує цикл роботи системи. Така організація забезпечує прозорість обробки даних та спрощує налагодження й супровід програмного модуля.

Загальна структура програмного модуля прогнозування хвороб сільськогосподарських культур на основі згорткових нейронних мереж може бути представлена у вигляді структурної схеми, що ілюструє основні компоненти системи та напрямки передачі даних між ними.

На рисунку 1 представлено архітектуру програмного модуля прогнозування хвороб сільськогосподарських культур.

Центральним елементом програмного модуля прогнозування хвороб сільськогосподарських культур є модель згорткової нейронної мережі, яка виконує автоматизований аналіз зображень рослин та формує прогноз щодо їхнього стану. Вибір саме згорткової нейронної мережі зумовлений її здатністю ефективно працювати з просторово структурованими даними та виділяти інформативні ознаки, характерні для різних типів захворювань рослин [1-3].

Архітектура розробленої моделі побудована за принципом послідовної обробки даних і складається з кількох функціональних блоків, що включають згорткові шари, шари підвибірки, шар нормалізації, повнозв'язаний шар та вихідний шар класифікації. Загальну структуру згорткової нейронної мережі, використаної у програмному модулі, подано на рисунку 2.

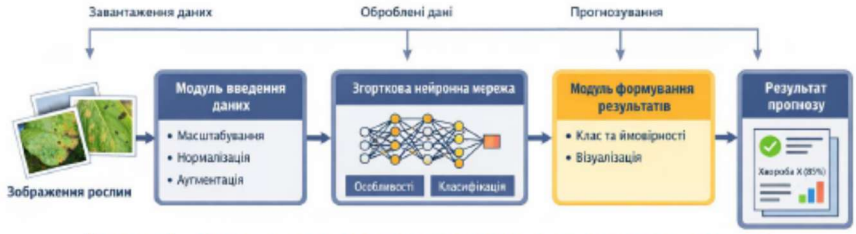

Рисунок 1 – Структурна схема програмного модуля прогнозування хвороб сільськогосподарських культур

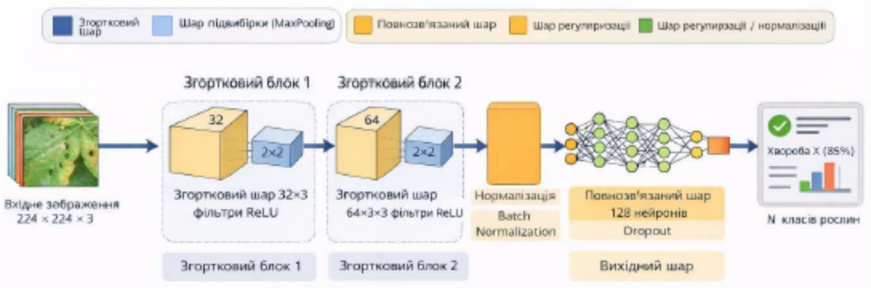

Рисунок 2 – Архітектура згорткової нейронної мережі

На вхід нейронної мережі подаються попередньо оброблені зображення рослин розміром 224×224×3 у форматі RGB. Перший згортковий блок містить згортковий шар із 32 фільтрами розміром 3×3, після якого застосовується функція активації ReLU та шар підвибірки MaxPooling із розміром вікна 2×2. Цей блок призначений для виділення базових просторових ознак, таких як контури та елементарні текстурні елементи.

Другий згортковий блок має аналогічну структуру, проте використовує 64 згорткові фільтри, що дозволяє виділяти складніші та більш абстрактні ознаки, пов'язані з характерними симптомами захворювань рослин. Застосування шару підвибірки на цьому етапі сприяє зменшенню просторової розмірності ознак і зниженню обчислювального навантаження, що є важливим для практичного використання програмного модуля.

Для підвищення стабільності процесу навчання та зменшення внутрішнього зміщення активацій у моделі використовується шар пакетної нормалізації (Batch Normalization). Даний підхід дозволяє пришвидшити збіжність алгоритму навчання та позитивно впливає на узагальнювальну здатність згорткової нейронної мережі [7].

Після завершення згорткових операцій сформовані ознаки передаються до повнозв'язаного шару, який містить 128 нейронів. Цей шар виконує інтеграцію виділених ознак і формує внутрішнє подання, необхідне для виконання класифікації. З метою зменшення ризику перенавчання у повнозв'язаному шарі застосовується регуляризація шляхом випадкового вимкнення нейронів (Dropout) з імовірністю 0,5.

Вихідний шар згорткової нейронної мережі містить N нейронів, де N відповідає кількості класів стану рослин. Для формування кінцевого прогнозу використовується функція активації Softmax, яка дозволяє отримати ймовірнісну оцінку належності зображення до кожного з можливих класів.

Таким чином, розроблена модель згорткової нейронної мережі поєднує відносну простоту архітектури з достатньою виразною здатністю для задач прогнозування хвороб сільськогосподарських культур. Обрана структура забезпечує баланс між точністю

класифікації та обчислювальною ефективністю, що є важливим для подальшої реалізації та експериментального дослідження програмного модуля.

Висновки. Обґрунтовано актуальність розроблення програмного модуля прогнозування хвороб сільськогосподарських культур на основі згорткових нейронних мереж. Визначено, що поширення хвороб рослин є однією з причин зниження врожайності та економічних втрат, а традиційні методи діагностики потребують значних часових витрат, залежать від досвіду фахівців і не завжди придатні для оперативного застосування на великих аграрних площах.

Показано, що використання методів комп'ютерного зору та глибокого навчання дає змогу автоматизувати аналіз зображень рослин і виявляти характерні ознаки захворювань. Особливу увагу приділено згортковим нейронним мережам, які здатні автоматично формувати просторові ознаки зображень і забезпечувати ефективну класифікацію стану рослин.

Сформульовано постановку задачі, визначено об'єкт, предмет і мету дослідження. Обґрунтовано, що для практичного використання методів глибокого навчання необхідно не лише побудувати модель прогнозування, а й спроєктувати програмний модуль, який забезпечує повний цикл обробки даних: завантаження зображення, попередню обробку, прогнозування та подання результату користувачу.

Розглянуто архітектуру програмного модуля, що складається з модуля введення даних, модуля попередньої обробки зображень, модуля прогнозування та модуля формування результатів. Така структура забезпечує послідовну обробку інформації, спрощує налагодження системи та створює умови для подальшого розвитку програмного рішення.

Описано архітектуру згорткової нейронної мережі, яка є центральним компонентом програмного модуля. Визначено основні елементи моделі: згорткові шари, шари підвибірки, шар пакетної нормалізації, повнозв'язаний шар, шар регуляризації та вихідний шар із функцією Softmax. Запропонована структура дозволяє поєднати достатню виразну здатність моделі з прийнятною обчислювальною ефективністю.

Список літератури

1. Ferentinos K. P. Deep learning models for plant disease detection and diagnosis. Computers and Electronics in Agriculture. 2018. Vol. 145. Pp. 311–318.
2. Mohanty S. P., Hughes D. P., Salathé M. Using Deep Learning for Image-Based Plant Disease Detection. Frontiers in Plant Science. 2016. Vol. 7. Article 1419.
3. Ramcharan A., Baranowski K., McCloskey P., Ahmed B., Legg J., Hughes D. P. Deep learning for image-based cassava disease detection. Frontiers in Plant Science. 2017. Vol. 8. Article 1852.
4. Barbedo J. G. A. Impact of dataset size and variety on the effectiveness of deep learning and transfer learning for plant disease classification. Computers and Electronics in Agriculture. 2018. Vol. 153. Pp. 46–53.
5. Fuentes A., Yoon S., Kim S. C., Park D. S. A Robust Deep-Learning-Based Detector for Real-Time Tomato Plant Diseases and Pests Recognition. Sensors. 2017. Vol. 17(9). Article 2022. doi:10.3390/s17092022.
6. Too E. C., Yujian L., Njuki S., Yingchun L. A comparative study of fine-tuning deep learning models for plant disease identification. Computers and Electronics in Agriculture. 2019. Vol. 161. Pp. 272–279.
7. Ioffe S., Szegedy C. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. arXiv. 2015. arXiv:1502.03167. doi:10.48550/arXiv.1502.03167.