Міністерство освіти та науки України
Національний університет «Києво-Могилянська академія»

Застосування згорткових нейронних мереж для розпізнавання образів в обраній предметній області

Use of convolutional neural networks for pattern recognition in a selected domain

Текстова частина до курсової роботи
за спеціальністю „Інженерія програмного забезпечення” 6.050103

Керівник курсової роботи:
к.т.н., доц. Олецький О.В.

Роботу виконав студент 3 курсу
Матійчик Т. Я.

Київ 2025

Міністерство освіти і науки України

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ»

Кафедра мультимедійних систем факультету інформатики

ЗАТВЕРДЖУЮ
Зав.кафедри мультимедійних систем,
доц., к.ф-м.н.
_____О. П. Жежерун
(підпис)
„__10__”_____жовтня_____2025 р.

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ
на курсову роботу

студенту Матійчику Т.Я., факультету інформатики 3 курсу

ТЕМА: **Застосування згорткових нейронних мереж (CNN) для розпізнавання образів/ Using of convolution neural networks for pattern recognition**

Вихідні дані:

- Навчальна вибірка для тренування моделей
- Різні варіації згорткової нейронної мережі для класифікації
-

Зміст ТЧ до курсової роботи:

Індивідуальне завдання
Вступ
1 Опис згорткових нейронних мереж
2 Проблематика обраної теми. Огляд літератури
3 Формування навчальної вибірки. Архітектура досліджених моделей
4 Проведення досліджень та аналіз результатів
Висновки
Список літератури

Дата видачі „_10_” _жовтня_ 2024 р. Керівник _____
(підпис)
Завдання отримав ____
(підпис)

Тема: Застосування згорткових нейронних мереж (CNN) для розпізнавання образів/ Using of convolution neural networks for pattern recognition

Календарний план виконання роботи:

№ п/п	Етап курсової роботи	Термін виконання етапу	Примітка
1.	Отримання завдання на курсову роботу.	10.10.2024	
2.	Ознайомлення з теоретичною літературою згіднотеми	22.11.2024	
3.	Визначення предметної області для дослідження	05.12.2024	
3.	Створення, збирання навчальної вибірки для навчання моделей	26.12.2024	
4.	Дослідження актуальних наукових робіт по темі	20.01.2025	
5.	Налаштування різних конфігурацій моделей, проведення експериментів, запис результатів	24.02.2025	
6.	Оформлення письмової частини курсової роботи	03.03.2025	
7.	Редагування письмової частини згідно зауважень наукового керівника	1.05.2025	
8.	Захист курсової роботи		

Студент **Матійчик Т. Я.**

Керівник **Олецький О.В.**

“___10___”_____жовтня_____

Зміст

Анотація 5
Вступ 5
Розділ 1. Опис згорткових нейронних мереж 7
1.1. Визначення згорткових нейронних мереж 7
1.2. Архітектура ЗНМ 7
1.2.1 Згортковий шар 8
1.2.2. Шар об'єднання 11
1.2.3. Шар згладжування 11
1.2.4. Повнозв'язаний шар 11
1.3. Навчання та оптимізація згорткових мереж. Адаптивна оптимізація 12
1.4 Перенесене навчання 19
Розділ 2. Проблематика обраної теми. Огляд літератури 21
2.1. Актуальні питання впровадження згорткових мереж у агропромисловість 21
2.2. Сучасні рішення із застосуванням ЗНМ 24
Розділ 3. Проведення досліджень 27
3.1. Використане програмне забезпечення та технології у дослідженні 27
3.2. Формування та препроцесинг вибірки даних 27
3.3. Обрана архітектура ЗНМ. Будова мереж EfficientNet. 31
3.3.1. Архітектура та особливості EfficientNet 31
3.3.2. Вибір моделі для експериментів 34
3.4. Експерименти 35
3.4.1. Повне та часткове тонке налаштування. Порівняння із ЗНМ, побудованої з нуля 39
3.4.2. Встановлення різних гіперпараметрів 43
3.4.3. Робота моделей EfficientNet B3 та побудованої з нуля без аугментації даних 46
Розділ 4. Результати. Подальші покращення 53
Висновки 55
Список використаних джерел 56

Анотація

Курсова робота присвячена застосуванню згорткових нейронних мереж у завданні розпізнавання образів у класифікація хвороб пшениці та соняшнику. Було досліджено ефективність претренованої мережі EfficientNet B3 на поставленій предметній області, порівняно з роботою ЗНМ, побудованої з нуля (from scratch англ.). Досліджено модифікації EfficientNet B3 у контексті тонкого налаштування (fine-tuning англ.) та застосування різних гіперпараметрів до повнозв'язаного шару та опорної частини (backbone англ.), вплив аугментації даних на успішність EfficientNet B3 та такої, що побудована з нуля. Проаналізовано ефективність мереж на метриках, як валідаційні точність та похибка, влучність-повнота, F1-бал, матриця похибок (confusion matrix англ.). Проведено огляд сумісних робіт, де порівняно результати досліджень із даним; розглядано успішність згорткових мереж на класифікації хвороб рослин. Досліджені актуальні досягнення та виклики згорткових мереж у вибраній галузі.

Ключові слова: згорткові нейронні мережі, класифікації хвороб рослин, EfficientNet B3, перенесене навчання, тонке налаштування, мультикласове розпізнавання зображень, аугментація даних

Вступ

Технології згорткових нейронних мереж досягли високого рівня успішності в задачах комп'ютерного зору. Завдяки високій ефективності в цих задачах їх інтегрують у велику кількість галузей. Зараз вони широко застосовуються в розпізнаванні медичних зображень, як знімки МРТ, рентгенівські та мікроскопічні знімки; у розпізнаванні облич, моніторингу дорожнього руху, задачах із детекції смуг руху тощо.

Це дослідження показує, що технологія має потенціал в агрокультурі, а саме в класифікації хвороб рослин. У сфері спостерігається декілька проблем, які здатні усунути згорткові мережі:

- Великі затрати часу й фінансів при детекції людьми
- Людський фактор, або необізнаність працівників у визначенні стану рослини
- Погодні умови, що унеможливлюють точне встановлення хвороби
- Фінансові збитки, пов’язані з хибною класифікацією

У роботі було взято 2 рослини для дослідження – пшениця та соняшник, які є одними з найпоширеніших у світі, зокрема в Україні. За даними Державної митної служби [1], в Україні за 2024 рік обсяги експорту пшениці складали 20 664 815 тон, в той час як соняшникової олії – 6 008 495. Це є два найбільш експортованих українських товари протягом 2024 року. Тому обидві культури відіграють важливу роль в економіці країни, і це є основним аргументом у виборі саме такого напрямку дослідження.

Отже, розробка ЗНМ, призначеної для класифікації хвороб пшениці та соняшнику, є актуальною. Щодо цього виділено декілька чинників: зменшення часу виявлення хвороб, оптимізація грошових витрат, розпізнавання за умов дощу, сильного вітру, туману – умов, за яких робота людей не є ефективною – та усунення потреби в ручній перевірці.

Розділ 1. Опис згорткових нейронних мереж

1.1. Визначення згорткових нейронних мереж

Протягом останнього десятиліття глибинне навчання виділилося в окремий напрям від машинного навчання. Його фокус охоплює побудову глибоких нейронних мереж (deep neural networks англ.), що імітують роботу нейронних з'єднань в людському мозку та складаються з багатьох видів шарів. Основна мета напряму – створення нейронних мереж, здатних до аналізу комплексних даних та наданні точного результату.

Згорткові нейронні мережі (Convolutional neural network англ.) є одним з видів глибоких нейронних мереж, що спроектовані для задач з обробки сітчастих даних, зокрема зображень або таблиць; хоча тип мережі використовують також і для аналізу текстових, звукових, відео-даних тощо. У задачах з класифікації зображень вони здатні оцінювати його належність до визначеного класу або класів.

Основою ЗНМ є згортки (convolution англ.), що проходять скрізь усі карти ознак (feature maps англ.) – багатовимірні масиви, або тензори скалярних значень – та видають оновлені карти ознак для наступних конволюцій чи інших операцій. Між згортками можливі операції об'єднання задля зменшення розмірностей карт ознак, часу на подальші обчислення рештою мережі. Після об'єднання відбувається згладжування, що перетворює карти ознак на суцільний вектор. Цей вектор передається на вхід повнозв'язаному шару, у якому кожен нейрон має зв'язки з усіма попередніми. У шарі кожен нейрон обчислює зважену суму відповідно до вхідних значень, застосовує активаційну функцію та передає результат наступним нейронам. У кінці ми отримуємо вектор з імовірностями належностей до визначених класів зображень.

1.2. Архітектура ЗНМ

ЗНМ містять наступні ключові складові: згорткові шари, шари об’єднання (Pooling), шар згладжування (Flatten) та повнозв’язаний шар (Fully-connected – FC).

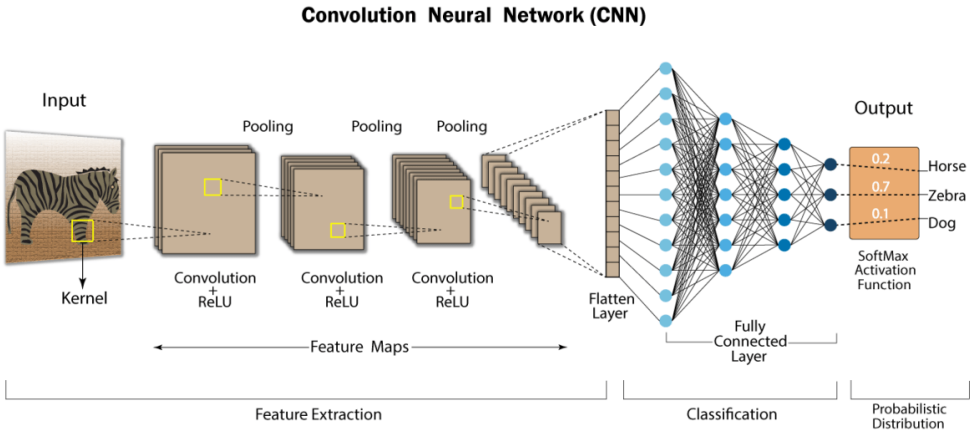


Рисунок 1.1 – Приклад архітектури ЗНМ для класифікації зображень. Посилання.

1.2.1 Згортковий шар

Суть згорткових шарів (convolutional layer англ.) полягає в тому, що вони здійснюють згортки над вхідними картами ознак, застосовуючи матричне множення між ними й так званими фільтрами, або ядрами (kernels англ.). Карти ознак – тензори, що містять матриці зі скалярами, зазвичай, пікселів. Розмірність карт дорівнює W×H×C, де W – довжина, H – ширина, а C – глибина карти, що позначає кількість їх каналів. На початку глибина може дорівнювати 3 або 1 в залежності, чи знімок є кольоровим (у форматі RGB) або чорно-білим. Фільтр – це квадратна матриця, розмір якої менший за карту ознак та яка призначена для виявлення таких ознак на карті, що відповідають значенням у фільтрі. Ознаками можуть бути лінії або кути на карті – для виявлення, чи карта ознак має діагональну лінію, фільтр матиме вигляд діагональної матриці. Фільтр сканує слайд – ділянку карти ознак, розмір якої дорівнює його розмірності – далі відбувається їх матричне множення, після чого фільтр зміщується на визначений крок (stride англ.) рухаючись до останнього слайду. На рисунку 1.2 описано процес згортки карти через фільтр з джерела [2].

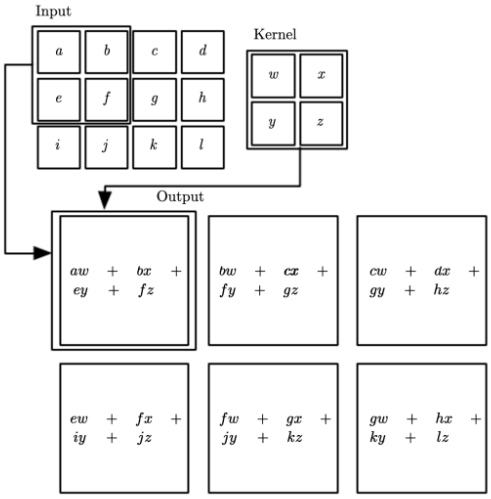


Рисунок 1.2 – Процес згортки вхідної карти 3×4 при застосуванні фільтра 2×2 з кроком 1.

Якщо фільтр при зсуві «виходить» за межі карти, до неї застосовується нульове заповнення (zero-padding англ.), щоб розмірності слайду та фільтру співпадали. Фактично це розширення розмірності вхідної карти нулями для успішної згортки.

Зазвичай, у таких шарах застосовуються багато фільтрів для видобутку багатьох характерних рис вхідних карт, що є на зображенні. Тому на виході згорткового шару ми отримуємо не 1 карту, а N вхідних карт розмірності D×D, де N дорівнює кількості використаних ядр. До того, оновлена розмірність карт буде наступною:

$$D = \frac{(I-F) + 2P}{S} + 1, \qquad (1.1)$$

де I – ширина або довжина вхідної карти;

F – розмір ядра;

P – розмір заповнення;

S – розмір кроку;

Застосування функції активації є останньою дією над картами ознак в згортковому шарі. Її задача полягає у внесенні нелінійних закономірностей між

нелінійними даними, для аналізу яких і призначені ЗНМ. Існують наступні поширені активаційні функції в конволюціях:

- випрямлений лінійний вузол ReLU (rectified linear unit англ.)

$$ReLU(x) = \begin{cases} x, & x > 0 \\ 0, & x \leq 0 \end{cases} \quad (1.2)$$

- сигмоїд (sigmoid англ.)

$$sigmoid(x) = \frac{1}{1+e^{-x}} \quad (1.3)$$

- гіперболічний тангенс tanh (hyperbolic tangent).

$$tanh(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}} \quad (1.4)$$

Здебільшого зараз використовується ReLU в побудові ЗНМ. Це пояснюється тим, що ReLU краще справляється з тим, що під час зворотнього поширення помилки (backpropagation англ.) градієнти – часткові похідні від ваги нейрона та обчисленої похибки після епохи тренування – майже не змінюються, як і ваги нейронів. Темп навчання мережі стрімко сповільнюється й спостерігається проблема затухання градієнтів (gradients vanishing problem англ.). ReLU завжди повертає додатні значення, таким чином значення градієнту від функції завжди дорівнює 1. Таким чином при значенні 1 градієнти передаються наступним нейронам, що розташовані за поточним нейроном, а при 0 не передаються. Завдяки цьому градієнти зменшуються не так стрімко протягом навчальних епох як у сигмоїді або tanh. Застосування ReLU продемонстровано на рисунку 1.3.

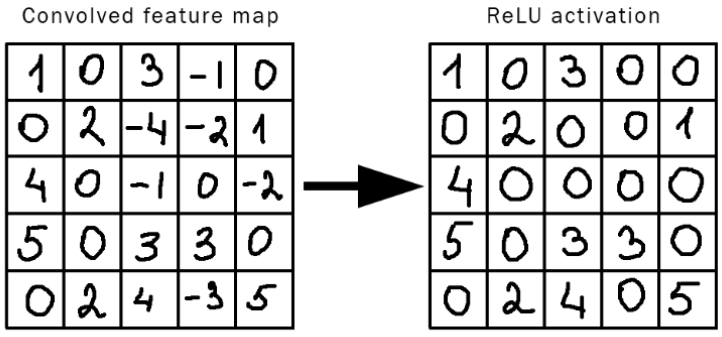


Рисунок 1.3 – ілюстрація застосування функції активації ReLU під час згортки

1.2.2. Шар об’єднання

Після згортки нові карти передаються на вхід шару об’єднання (pooling layer англ.). Шари об’єднання проходять по кожній карті ознак, виділяючи на них певну зону та усунення несуттєвих, невеликих значень. У кожній карті алгоритм виділяє зону N×N, лишає в ній максимальне значення або рахує середнє на основі чисел в ділянці – відповідно до способу вирізняють максимальне та середнє об’єднання (max pooling, average pooling). Опісля алгоритм зсовує зону на крок – такий же як і в згортці – та повторює дії до останньої ділянки. Його призначення – виділення найважливіших ознак із карти та зменшення розмірності задля оптимізація оброблених даних для наступних згорток. Результатом об’єднання є N карт ознак з розмірністю, що обчислюється за формулою (1.1) та де N дорівнює кількості карт на вході в шар.

1.2.3. Шар згладжування

Після всіх послідовностей шарів згортки та об’єднання кінцеві карти ознак передаються в спеціальний шар згладжування. У ньому карти ознак конвертуються у вектор, розмірність якого обчислюється за формулою:

$$S = WHC \qquad (1.5)$$

Де W – к-сть стовпців карти, H – к-сть рядків карти, C – глибина, або к-сть каналів. У результаті повертає суцільний вектор із вагами з вхідних карт ознак.

1.2.4. Повнозв’язаний шар

Повнозв’язаний шар (Fully-connected layer англ.) є кінцевим шаром у ЗНМ, у якому обчислюються імовірності належності до заданих класів. Він складається з послідовності шарів нейронів, що є з’єднані з усіма нейронами з попередніх шарів. Перший шар нейронів, кількість нейронів у якому дорівнює кількості вхідних ваг, приймає згладжений вектор ознак. Далі відбувається пряме

розповсюдження (forward propagation англ.) – нейрони у кожному прихованому шарі обчислюють зважену суму добутків ваг та вхідних значень (формула надана в 1.6) із доданим зсувом, застосовують функцію активації ReLU й передають результат наступним нейронам. Кінцевий нейронний шар у повнозв'язаному шарі складається з нейронів, кількість яких дорівнює кількості класів, приймає вектор із N значень – N дорівнює кількості класів класифікації даних – застосовує до них функцію softmax (формула надана в 1.7) для конвертації значень у вірогідності, сума яких обов'язково дорівнює 1. Фінальний результат – це вектор вірогідностей належності до певного класу, де найбільше значення позначає найбільшу впевненість у належності.

$$y = f(b + \sum_{n}^{i=1} w_i x_i) \quad (1.6)$$

y – зважена сума, що передається наступним нейронам;

n – кількість зв'язків із попередніми нейронами;

b – значення зсуву;

w_i – вага зв'язку із попереднім і-тим нейроном;

x_i – вихідне значення з і-того нейрону;

$$\sigma(x_i) = \frac{e^{x_i}}{\sum_{k=1}^{n} e^{x_k}} \quad (1.7)$$

де x_i – і-те значення кінцевого вектора;

n – кількість значень у векторі вірогідностей;

1.3. Навчання та оптимізація згорткових мереж. Адаптивна оптимізація

Повний цикл навчання складається з прямого (forward propagation англ.) та зворотного поширення похибки (backpropagation англ.). Після проходження 1 навчальної епохи – 1 циклу прямого поширення – та визначення вірогідностей належності зображення до класів, обов'язково відбувається обчислення похибки

передбачених вихідних балів відносно істинних. Для кожного зображення зберігається вектор фундаментальної істини (Ground truth vector англ.), який зберігає правдиві бали належності до класів. У багатокласовому розпізнаванні використовується формула категоріальної перехресної ентропії (Categorical cross entropy англ.):

$$CCE(x_i) = -\sum_{I=1}^{N} t_i \log(\sigma(x_i)) \quad (1.8)$$

Після обчислення похибок відбувається процес зворотного поширення помилки, що створений максимально зменшити значення функції похибки. Він оновлює, уточнює ваги нейронів у попередніх шарах шляхом градієнтного спуску – механізму знаходження мінімальної похибки через обчислення часткових похідних похибки від значень ваг. Такий вид оптимізатора називається ще стохастичний ґрадієнтний спуск SGD (Stochastic Gradient Descent). Для визначення сили зміни ваги використовується рейтинг навчання, який відображає “крок” ґрадієнтів на шляху до глобального мінімуму. Базова формула обчислення ґрадієнтів та оновлення ваг є наступною:

$$w = w_0 - \alpha \frac{\partial(Loss)}{\partial(w_0)} \quad (1.9)$$

де w_0 – поточне значення нейрона;

w – нове значення нейрона;

α – рейтинг навчання;

Loss – значення функції похибки;

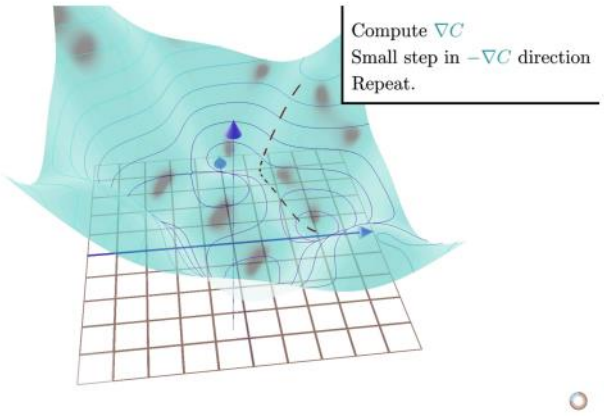


Рисунок 1.4 – ілюстрація ґрадієнтного спуску з джерела [2]. Сіра пунктирна лінія позначає шлях пошуку глобального мінімуму.

Тим не менше, оптимізація за формулою 1.9 призводить до фіксації ґрадієнтів у локальних мінімумах, коли їхнє значення коло локальних мінімумів є близьким до нуля. Рішенням проблеми стало впровадження до SGD моментуму (імпульсу), який накопичується протягом зміни ваги в одному напрямку й сприяє виходу ґрадієнтів з локальних мінімумів:

$$w = \nu + w_0 \tag{1.10}$$

де w – нове значення ваги;

ν – значення моментуму;

w_0 – поточне значення ваги.

$$\nu = \eta \nu_0 - \alpha \frac{\partial(Loss)}{\partial(w_0)} \tag{1.11}$$

де ν, ν_0 – відповідно нове та поточне значення моментуму;

η – коефіцієнт моментуму, який визначає скільки імпульсу попередньої ваги передається наступній;

α – рейтинг навчання;

Loss – значення функції похибки;

Прискорені ґрадієнти Нестерова (Nesterov Accelerated Gradients) є удосконаленням базового принципу з імпульсом, який робить його «керованим» та усуває проблему «перескоку» градієнтів через глобальні мінімуми [3]. Тепер оптимізатор перед обрахунком ґрадієнтів знаходить приблизне майбутнє значення ваги *w*, обчислюючи різницю $w - \eta v_0$, де ηv_0 є попереднім моментумом. Повна формула NAG виглядає наступним чином:

$$w_1 = w_0 - \eta v_0 \tag{1.12}$$

де w_1 – майбутнє прогнозоване значення ваги;

w_0 – поточна вага;

η – коефіцієнт моментуму;

$$v = \eta v_0 + \alpha \frac{\partial(Loss)}{\partial(w)}, w = w - v \tag{1.13}$$

де v – імпульс ваги прогнозованої ваги w;

v_0 – поточний імпульс;

α – рейтинг навчання.

Передовою є ідея адаптивної оптимізації, в якій рейтинг навчання α змінюється протягом зворотного поширення помилки. Основний принцип ідеї полягає в тому, що при надто малій зміні ґрадієнтів, рейтинг навчання збільшується, а при надто великій зміні зменшується. Існує декілька найпоширеніших адаптивних оптимізаторів, що використовуються в навчанні ЗНМ:

- Adagrad [3]: розширений алгоритм SGD, у якому рейтинг навчання уточнюється через кожну епоху із застосуванням попередніх ґрадієнтів. Ґрадієнти зберігаються в діагональній квадратній матриці, де елемент g_i є сумою квадратів всіх попередніх градієнтів:

$$G_{i,i} = G_{i-1,i-1} \cdot \frac{\partial(Loss)}{\partial(w_0)}, w_i = w_{i-1} - \frac{\alpha}{\sqrt{G_{ii}+\varepsilon}} \cdot \frac{\partial(Loss)}{\partial(w_{i-1})} \tag{1.14}$$

де $G_{i,i}$, $G_{i-1,i-1}$ – відповідно елементи матриці сум квадратів ґрадієнтів в і-ій та і-1 позиціях;

w_i, w_{i-1} –відповідно нове та поточне значення ваги;

α – рейтинг навчання;

ε – мізерне значення для запобігання діленню на нуль (здебільшого $\varepsilon = 1e^{-8}$).

Основний недолік цього оптимізатора полягає в неперервному зростанні попередніх ґрадієнтів, через що рейтинг навчання α неперервно зменшується. Відбувається стагнація ваг, і пошук глобального мінімуму функції похибки стає неефективним.

- Adadelta [3]: адаптивний алгоритм, що вирішує проблему монотонну зростаючих сум ґрадієнтів Adagrad. Натомість Adadelta використовує експоненційні спадаючі середні квадратів ґрадієнтів (exponentially decaying averages англ.), які з більшими епохами зменшують вплив давніших значень на противагу нещодавнім. Фактично тепер рейтинг навчання ділиться на середнє квадратичне попередніх ґрадієнтів у період t:

$$E[(\frac{\partial(Loss)}{\partial(w_t)})^2] = \rho \cdot E[(\frac{\partial(Loss)}{\partial(w_{t-1})})^2] + (1-\rho)E[(\frac{\partial(Loss)}{\partial(w_t)})^2] \quad (1.15)$$

де E – експоненційне спадаюче середнє ґрадієнтів;

ρ – спадаюча стала, що визначає силу зменшення впливу давніх ґрадієнтів (принцип схожий до роботи моментуму).

$$RMS[g]_t = \sqrt{E\left[\left(\frac{\partial(Loss)}{\partial(w_t)}\right)^2\right] + \varepsilon} \quad (1.16)$$

$$w_t = w_{t-1} - \frac{RMS[g]_{t-1}}{RMS[g]_t} \cdot \frac{\partial(Loss)}{\partial(t)} \quad (1.17)$$

де RMS – середньоквадратичне ґрадієнту g у періоді t;

ε – значення для уникнення ділення на нуль.

- Adam [4]: Алгоритм поєднує сильні сторони попередніх оптимізаторів, застосовуючи водночас ідею імпульсу (моментуму) та RMS. Для цього в Adam вводяться обидві величини для збереження експоненційних спадаючих середніх ґрадієнтів m_t, як імпульс, та квадратів ґрадієнтів v_t.

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t$$
$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2$$

Рисунок 1.5 – обчислення параметрів в Adam [3]. β_1 та β_2 є ідентичними до коефіцієнту моментуму η (див. формулу 1.12) та сталій ρ (див. формулу 1.15). g_t є ґрадієнтом відносно ваги w_t.

За авторами [4], протягом початкових періодів оптимізатор може бути упередженим відносно нуля. Для цього вони обраховують ті ж параметри, зважені на зсув:

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t}$$
$$\hat{v}_t = \frac{v_t}{1 - \beta_2^t}$$

Рисунок 1.6 – обчислення скорегованих m_t та v_t [3].

Останньою дією в оптимізації є власне оновлення ваги, ідентично до підходів RMSProp і Adadelta:

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\hat{v}_t} + \epsilon} \hat{m}_t$$

Рисунок 1.7 – формула оновлення параметрів через Adam [3]. θ_t, θ_{t+1} позначає поточну та нову вагу відповідно.

Algorithm 1: *Adam*, our proposed algorithm for stochastic optimization. See section 2 for details, and for a slightly more efficient (but less clear) order of computation. g_t^2 indicates the elementwise square $g_t \odot g_t$. Good default settings for the tested machine learning problems are $\alpha = 0.001$, $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 10^{-8}$. All operations on vectors are element-wise. With β_1^t and β_2^t we denote β_1 and β_2 to the power t.

Require: α: Stepsize
Require: $\beta_1, \beta_2 \in [0, 1)$: Exponential decay rates for the moment estimates
Require: $f(\theta)$: Stochastic objective function with parameters θ
Require: θ_0: Initial parameter vector
  $m_0 \leftarrow 0$ (Initialize 1st moment vector)
  $v_0 \leftarrow 0$ (Initialize 2nd moment vector)
  $t \leftarrow 0$ (Initialize timestep)
  while θ_t not converged **do**
    $t \leftarrow t + 1$
    $g_t \leftarrow \nabla_\theta f_t(\theta_{t-1})$ (Get gradients w.r.t. stochastic objective at timestep t)
    $m_t \leftarrow \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t$ (Update biased first moment estimate)
    $v_t \leftarrow \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot g_t^2$ (Update biased second raw moment estimate)
    $\widehat{m}_t \leftarrow m_t / (1 - \beta_1^t)$ (Compute bias-corrected first moment estimate)
    $\widehat{v}_t \leftarrow v_t / (1 - \beta_2^t)$ (Compute bias-corrected second raw moment estimate)
    $\theta_t \leftarrow \theta_{t-1} - \alpha \cdot \widehat{m}_t / (\sqrt{\widehat{v}_t} + \epsilon)$ (Update parameters)
  end while
  return θ_t (Resulting parameters)

Рисунок 1.8 – повний алгоритм роботи оптимізатора Adam [4].

1.4 Перенесене навчання

Перенесене навчання (transfer learning англ.) – техніка машинного навчання, в якому використовуються набуті знання з одної предметної області для вдосконалення ефективності моделі в іншій сумісній області [5]. На практиці це означає застосування нейронних мереж, тренованих та адаптованих до початкових даних, для задач у сумісному домені. Техніка перенесеного навчання значно оптимізовує обчислювальні ресурси та час для тренування моделі; також вона є корисною у випадку малої вибірки даних із нового домену, сприяє пристосуванню моделі до все більш різноманітної інформації – генералізації.

Виділяють 3 види перенесеного навчання: індуктивне (inductive англ.), навчання без вчителя (unsupervised англ.) та трансдуктивне (transductive англ.).

Суть індуктивного навчання полягає у застосуванні знань із оригінального домену в адаптації до подібного. Фактично, це адаптація претренованої мережі, що навчалася на певній оригінальній вибірці і мала вирішувати одну задачу, до вибірки, що володіє схожими ознаками початкової, але задача перед якої поставлена інша. Для такого підвиду перенесеного навчання важливим є використання переднавчених ваг на початковому датасеті, яке переміщує базові знання до нової задачі.

Трансдуктивне навчання, або адаптація домену, зазвичай має призначення до однакових завдань, але різних доменів [6]. Застосовується у випадках протилежних до індуктивних – завдання для мереж є ідентичними, проте для другого обирається навчальна вибірка, домен, чиї дані мають частково відмінну природу. Здебільшого у трансдуктивному навчанні нова вибірка не є розміченою (unlabeled англ.), тобто не класифіковані.

Навчання без учителя, як індуктивне, містить однакові початкову та поточну задачі [7]. Проте у поточній задачі використовуються нерозмічені дані на противагу початковій. Сферами застосування такого підвиду перенесеного навчання переважно є кластеринг (clustering англ.) – обробка нерозмічених

даних, їхня категоризація за певними закономірностями –, також зменшення розмірностей у датасетах (dimension reduction англ.) та асоціативні правила (association rules), де мережа досліджує потенційні кореляції між одиницями даних у вибірці.

У цій роботі застосовано індуктивний підхід перенесеного навчання. У досліджені було взято переднавчену конволюційну мережу, яка тренувалася на даних, зібраних із платформи ImageNet [8]. Цей датасет містить близько 14 197 122 зображень, розбитих по 1000 класах, які містять зразки різних об’єктів із реального світу, зокрема, рослин. Це ж дослідження покликане адаптувати цю мережу до звуженого домену – хворобливих та здорових станів двох агрокультур пшениці та соняшнику.

Розділ 2. Проблематика обраної теми. Огляд літератури

2.1. Актуальні питання впровадження згорткових мереж у агропромисловість

Одною з найпоширеніших проблем у сільському господарстві є боротьба з хворобами. На захист урожаю аграрії змушені виділяти великий як людський, так і фінансовий ресурси. Зокрема, завдання виявлення хвороб вимагає багато часу через великі площі земельних ділянок та залучення ручної праці. У цьому напрямку згорткові мережі стали одними з передових рішень, що здатні точно розпізнавати хвороби завдяки їхній глибокій архітектурі, методикам глибинного навчання. Ба більше, завдяки можливості масштабування та автоматизації вони здатні значно швидше та своєчасно визначати зони інфекції на полях.

Однак впровадження технології все ще потребує суттєвих удосконалень. Автори [9] провели аналіз 100 наукових статей, що стосуються класифікації хвороб рослин за допомогою CNN. Вони виділили основні специфічні для галузі проблеми, серед яких – обмежені датасети знімків інфікованих рослин, проблема вибору фону зображення. Дослідники визначають чотири шляхи подолання першої проблеми: ауґментація даних, перенесене навчання, залучення непрофесійних учасників до збору даних та обмін даними між науковими спільнотами. Рішенням для другої проблеми автори називають сегментацію. Завдяки сегментації зменшується можлива похибка, пов’язана зі спільними ознаками на рослині та оточенні, що може призвести до розбіжності між справжнім та передбаченим класами. Тим не менш, вони вказують на те, що здебільшого фон зображення зберігається задля їх однорідності. До того ж, Моганті та ін. [10] провели експеримент, в якому перевірялася ефективність роботи двох моделей AlexNet та GoogleNet на кольоровому, сегментованому та чорно-білому датасеті. У підсумку з’ясувалося, що різниця точностей між кольоровими та сегментованими зображеннями була мінімальною. Проте перевага в класифікації моделей була саме на кольоровому датасеті.

Також у статті [11] розглянуто 16 досліджень, пов’язаних із застосуванням ЗНМ до класифікації хвороб рослин, та визначено переваги й недоліки описаних у них методів. Також ця робота презентує актуальні питання та виклики перед сучасними ЗНМ-розробками у розпізнаванні хвороб рослин, освітлено подальші напрямки розвитку даних технологій, запропоновані рішення розглянутих проблем. Розглянувши роботи дослідники виділили наступні проблеми, пов’язані з датасетами: їхня незбалансованість та невідповідність реальним умовам класифікації хвороб рослин. Незбалансованість визначається тим, що обсяг зображень одних класів або шаблонів (labels англ.) значно перевищує інших. І справді, це призводить до збільшення упередження до більших класів на противагу меншим – при оцінці зображення модель надаватиме перевагу більшим класам; погіршеного узагальнення (generalization англ.) – здатність точно класифікувати зображення поза тренувальною вибіркою – та малої конвергенції між значенням точності та похибки. Під невідповідністю автори розуміють те, що багато датасетів сформовані не в природніх умовах, зокрема під час дощу, сильного вітру, туману, недостатнього освітлення, а в окремих штучних умовах, наприклад в лабораторії на столі, підставках, за спеціального освітлення тощо. Це є суттєвий недолік для впровадження ЗНМ для виявлення хвороб на реальних умовах сільськогосподарської промисловості.

У роботі [12] дослідники проаналізували близько ста джерел, релевантних до теми імплементації ЗНМ у розпізнаванні хвороб рослин, визначили чотири головні проблеми та запропонували їх рішення. Пріоритетною проблемою автори назвали неповноцінність датасетів у розмірі та різноманітності. Видобування даних вимагає великих обсягів часу та людського ресурсу, що робить надзвичайною складною задачу побудови повноцінного датасету у цій галузі. За словами вчених на сьогодні існують п’ять методів боротьби з нерепрезентативністю даних, якими є перенесене навчання, ауґментація даних, навчання з декількома зразками (few-shot learning англ.), залучення не спеціалістів до збору даних (citizen science англ.) та дата-шеринг (data sharing англ.). Другою проблемою є недостатня адаптивність (robustness англ.) мереж

до складних випадків. Насправді, ця проблема близько перетинається з першою, адже нерепрезентативністю даних сприяє навченості моделі до конкретних зразків – перенавчанню (overfitting англ.). Таким чином, моделі буде складно точно класифікувати знімки хвороб, зроблені за поганих погодних умов дощу, туману, вітру або за вечірньої, ранньої пори. Серед запропонованих рішень вони виділяють спрощення архітектури моделі, що водночас зменшує рівень перенавчання, проте обмежує її здатність у розпізнаванні складних ознак на знімках. Також долають цю проблему урізноманітнення датасетів, тобто пряме додавання нових зразків, навчання без учителя, вкраплення спотворень зображень як розмиття та ротації.

Третьою проблемою є варіативність та поєднання симптомів (ознак) хвороб на зображеннях, що призводить до хибної класифікації. У статті описано три сценарії, за яких з'являється ця проблема:

- хвороби рослин можуть мати різні стадії хвороби протягом її розвитку – наприклад легка, середня, важка форми;
- на одному знімку можуть бути виявлені декілька хвороб;
- схожі симптоми однієї хвороби проявляються в декількох інших.

Основним рішенням проблеми автори виділили формування класів хвороби, що містять повний обсяг і опис усіх можливих симптомів за різних умов. Це необхідно для побудови адаптованої та точної ЗНМ для застосування у даній галузі. Тим не менш, процес формування такого датасету є високозатратним з точки зору часу, фінансів та людей, тому пропонується поступове збагачення датасету різними зразками з практичного застосування.

Останньою проблемою є вплив фону, або бекграунду, зображення на класифікацію. Існують зображення двох типів: такі, що мають уніфікований фон та такі, що взяті з реальних умов. Для обидвох сценаріїв рішенням є техніки сегментації зображень такі, як порогування (threshold segmentation), Кластеризація методом k-середніх, повні згорткові мережі (Fully convolutional networks англ.) та watershed segmentation.

Отже, найбільш гостро поставлені проблеми, що стосуються навчальної вибірки для мереж. Кількість знімків у датасетах не відповідають адекватній кількості. Також вони не містять або недостатньо містять тих зображень, що зроблені у реальних умовах підприємств або за складних погодних умов. Головними рішеннями є ауґментативні та сегментативні техніки, доповнення датасетів зображеннями, зробленими у реальних умовах та перенесене навчання.

2.2. Сучасні рішення із застосуванням ЗНМ

Тим не менш, актуальні дослідження показують потенціал у застосуванні ЗНМ до вибраної галузі – сучасні розробки досягають високих показників точності класифікації хвороб за низького значення похибки. Протягом 2024 року були опубліковані дослідження із застосування ЗНМ до класифікації хвороб рослин, в тому числі пшениці та соняшника, у яких було досягнуто передових результатів.

Так Фанг та ін. [13] розробили модель, яка поєднує в собі ідеї згорткових мереж сімейства Inception – збільшення ширини шарів та одночасний видобуток ознак під різними розмірами – та ResNet – перенесення ознак із входу до шарів на вихід для запобігання деградації мережі та градієнтів – а також механізмів уваги CBAM та ECA (англ. attention mechanisms) задля виокремлення найважливіших ділянок зображення. CBAM (Convolutional Block Attention Mechanism) є механізмом уваги, що поєднує просторове та міжканальне виділення пріоритетних ділянок карт ознак. ECA (Efficient Channel Attention) є покращеною версією механізму SE (Squeeze-And-Excitation), що спрощує обчислення завдяки застосуванню 1×1 згортки замість повнозв’язаних шарів, як у SE. Варто зазначити, що датасет дослідників складався лишень із двох частин – тренувальної та тестової (валідаційної). Оптимізатор Adam був використаний як такий, що дає найбільший приріст точності. Запропонована мережа досягла

на тестовій підвибірці 98,76% точності, 98,77% влучності, 98,81% повноти – F1-значення склало 98,79%.

У дослідженні [14] створили конволюційну мережу CropNet для розпізнавання хвороб пшениці. Для цього вони розробили нову архітектуру, у якій пре-тренована мережа EfficientNet B0 – частина структури до повнозв'язаного шару – була встановлена як нетренована (freezed англ.) – застосовується для видобування ознак (feature extraction англ.). Опісля вихідні ознаки опрацьовуються створеною згортковою мережею, що складається з 3 згорткових шарів із 32, 64 та 128 фільтрів 3x3, нормалізацій батчу, шару пулінгу та відсіву (dropout англ.) зі значенням 0.5; далі – вбудований повнозв'язаний із використанням softmax функції. CropNet досягла передових результатів на тестовій частині, показавши 99,80% точності, 100% повноти (recall англ.), 99% влучності (precision англ.) та 99,70% F1-балу. Це дослідження доводить, що EfficientNet B0 слугує якісною моделлю для використання перенесеного навчання в задачах класифікації хвороб рослин.

Ґульзар та ін. [15] розглянули ефективність застосування різних моделей згорткових мереж на задачі класифікації 4 класів хвороб соняшнику: пероноспороз, сіра гниль, здоровий стан та пошкодження листя (downy mildew, gray mold, fresh leaf, leaf scars англ. відповідно). У роботі були обрані моделі AlexNet, VGG16, InceptionV3, MobileNetV3 та EfficientNet B3 як такі, що є провідними розробками на даний час, та проведені аналізи результатів на сучасних метриках precision, recall та F1-score. Результати експериментів показали, що EfficientNet B3 досягла найвищої точності класифікації серед усіх розглянутих моделей. Для класів хвороб, таких як пероноспороз, здоровий стан, сіра гниль та пошкодження листя, досягнуті точності 94.4%, 100%, 100%, 96.1%; досягнуті влучності 0.957, 1.0, 1.0, 0,949; значення повноти 0.949, 1.0, 1.0, 0.961; значення F1-балу 0.950, 1.0, 1.0, 0.955 відповідно.

Рамадан та ін. [16] впровадили техніки ауґментації даних CycleGAN, SMOTE, STOTETomek та ADASYN в архітектуру таких ЗНМ-моделей, як DenseNet121, ResNet50V2, DenseNet169, Xception, ResNet152V2 та MobileNetV2. Датасет складався із 407 зображень здорової пшениці та хворої на смугасту іржу, септоріоз. CycleGAN є розширенням генеративних змагальних мереж, що створюють нові зразки даних на основі вхідних даних. ADASYN (Adaptive Synthetic Sampling) є методикою ауґментації даних генерації даних для значно менших класів у датасеті, що усуває проблему дисбалансу між обсягом зображень різних класів. SMOTE (Synthetic Minority Over-sampling Techinque) також бореться з проблемою дисбалансу класів у вибірці як і ADASYN, але використовує інший підхід, обираючи випадкове зображення з малого класу із застосуванням k-найближчих сусідів (k-nearest neighbour англ.). SMOTETomek виконує ідентичні дії, що й SMOTE, проте використовує зв’язки Томека (Tomek links англ.), що знаходять найближчих сусідів двох сутностей з більшого та меншого класу, усувають ці зв’язки для чіткішого відокремлення двох класів. Результати експериментів показали, що без вкраплення ауґментативних технік найефективнішою мережею стала MobileNetV2 із точністю класифікації 98.36%. Із застосуванням SMOTE найкращий показник був 85.25% у ResNet152V2 , із застосуванням SMOTETomek – 84.52% у ResNet152V2, із застосуванням ADASYN, CycleGAN – 100% у MobileNetV2. Застосовуючи CycleGAN разом із MobileNetV2 найкращі точності були досягнути й мережами DenseNet121, DenseNetV169, Xception, ResNet152V2 на противагу іншим методикам.

Отже, на даний час в агропромисловості спостерігається стрімкий ріст досліджень та впровадження згорткових нейронних мереж, комбінуючи різні сучасні технології та практики. Результати досліджень є передовими та доводять, що ЗНМ мають великий потенціал для комерційного впровадження у сільському господарстві. Для цього потребується більше досліджень стосовно багатошаблонного розпізнавання хвороб рослин та розпізнавання хвороб у реальному часі.

Розділ 3. Проведення досліджень

3.1. Використане програмне забезпечення та технології у дослідженні

Для розгортання згорткових мереж була використана одна з найпоширеніших бібліотек у машинному навчанні PyTorch. Основними перевагами фреймворку називають зручність, простоту використання через його широку інтеграцією з Python, підтримку гнучких розробок глибоких мереж та швидкого прототипування; орієнтацію бібліотеки на дослідницьку, наукову роботу. Додатково були використані інші фреймворки: для візуалізації процесу навчання та тестування моделі були застосовані MatPlotLib, Seaborn та Sklearn для побудови матриці похибок, обчислення похідних від неї значень.

3.2. Формування та препроцесинг вибірки даних

Під час формування датасету було використано два посібники з хвороб пшениці [17] та соняшнику [18]. Для дослідження були використані вже доступні датасети з зображеннями хвороб пшениці й соняшнику із платформ Kaggle [19, 20] та Mendeley [21], а також були зібрані зображення за запитами у пошукові системі Google. Фінальна вибірка складається з 7162 зображень у форматах .jpg, .jpeg, .png, розподілених в тренувальну, валідаційну та тестову підкатегорії – train, valid, test – зі співвідношенням 8:1:1.

(а) (б) (в)

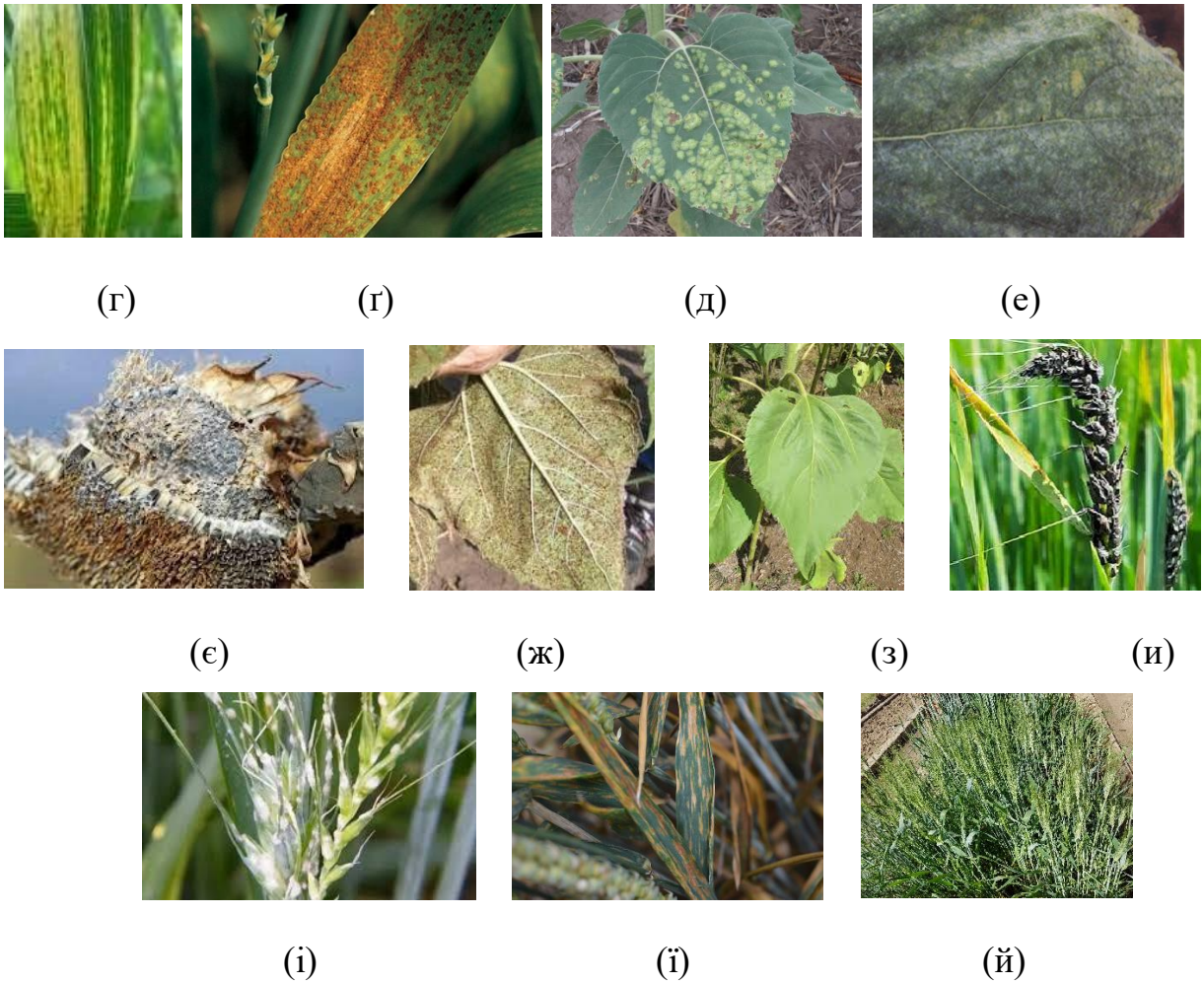


Рисунок 3.1 – ілюстрація зображень з датасету соняшнику – (а) альтернаріоз, (б) здоровий стан (кошик), (в) ризопусна гниль, (д) пероноспороз, (е) борошниста роса, (є) склеротиніоз, (ж) іржа листя, (з) здоровий стан (листя) – та пшениці – (г) мозаїка, (ґ) іржа листя, (и) сажки, (і) борошниста роса, (ї) септоріоз, (й) здоровий стан.

Датасет містить 13 класів зображень, що позначають хвороби:

- Для соняшника

Захворювання	train	valid	test
Альтернаріоз листя	151	18	18
Пероноспороз	171	21	22
Борошниста роса	117	15	14
Ризопусна гниль кошика	90	11	12
Склеротиніоз кошика	95	11	12
Іржа листя	120	15	14
Здоровий стан	362	45	45
Загалом	1106	136	137

Таблиця 3.1 – розподіл зображень по хворобах соняшнику та підкаталогах train, valid, test

- Для пшениці

Захворювання	train	valid	test
Борошниста роса	556	70	69
Мозаїка листя	289	36	37
Іржа листя	1172	146	147
Септоріоз	1050	131	131
Сажки	817	102	102
Здоровий стан	743	93	93
Загалом	4627	578	579

Таблиця 3.2 – розподіл зображень по хворобах пшениці та підкаталогах train, valid, test

Із таблиць 3.1 та 3.2 видно, що деякі класи хвороб цих рослин містять малу кількість зображень, що призведе до значного падіння точності віднесення тої чи іншої хвороби до відповідного класу. Саме у такому випадку було інтегровано перенесене навчання як ефективний метод побудови рішення для класифікації хвороб рослин із застосуванням ЗНМ. Також тренувальна, валідаційна та тестова вибірки були розподілені на батчі по 32, 16 та 16 передоброблених зображень відповідно.

Додатково до навчальної вибірки була застосована аугментація даних (data augmentation англ.). При аугментація даних створюються нові зразки зображень на основі оригінальних зразків [22]. Таким чином датасет розширюється модифікованими зображеннями для досягнення більшої генералізації – кращої адаптації моделі до різних даних. Також аугментація є ефективною для боротьби з перенавчанням. Тож задля збільшення конвергенції між тренувальною та валідаційною точностями було застосовано дата аугментацію тренувальної вибірки, а саме трансформації повороту на 30 градусів RandomRotation(), та горизонтальне віддзеркалення RandomHorizontalFlip().

Важливим елементом для покращення навчання мережі є застосування батч-нормалізації [23], або пакетної нормалізації вхідного датасету (batch normalization англ.). Вона застосовується для унормування та прискорення навчання глибинних нейронних мереж шляхом певних операцій над вхідними даними як у внутрішніх шарах, так і в першому шарі, який приймає незмінні

дані. Нормалізація змінює дані таким чином, щоб вони були «відцентровані» відносно нуля та мали стандартне відхилення зі значенням 1. Реалізується це із застосуванням наступних формул із [23]:

$$\mu_B = \frac{1}{m}\sum_{i=1}^{m} x_i \, , B = \{x_1, x_2 \dots m\} \qquad (3.1)$$

де m – останнє значення батчу-пакету B;

x_i – і-ий елемент пакету B.

μ_B – середнє всіх елементів з B.

$$\sigma_B{}^2 = \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_B)^2 \, , B = \{x_1, x_2 \dots m\} \qquad (3.2)$$

де m – останнє значення батчу-пакету B;

σ_B – стандартне відхилення елементів з B;

x_i – і-ий елемент пакету B.

μ_B – середнє всіх елементів з B.

Далі кожен елемент із батчу проходить нормалізацію із застосуванням формули:

$$\hat{x}_B = \frac{x_i - \mu_B}{\sqrt{\sigma_B{}^2 + \varepsilon}} \qquad (3.3)$$

де ε – дуже мале значення для запобіганню діленню на нуль.

Варто звернути увагу, що лишень нормалізація може деформувати значення в елементах, спотворюючи важливі ознаки для класифікації. Щоб дотримати оригінальне розташування ознак значення зсуваються на значення β та масштабують в значення γ:

$$y_i = \gamma \hat{x}_i + \beta \qquad (3.4)$$

де γ, β - треновані параметри, так як інші ваги в моделі;

Мережі EfficientNet треновані на даних з датасету ImageNet, що містить близько 15 мільйонів зображення, тому для нормалізації були використані значення

середнього та стандартного відхилення за замовчуванням [24] – (0.485, 0.456, 0.406), (0.229, 0.224, 0.225) відповідно.

3.3. Обрана архітектура ЗНМ. Будова мереж EfficientNet.

3.3.1. Архітектура та особливості EfficientNet

Для роботи було обрано сімейство згорткових нейронних мереж EfficientNet [25]. Розроблені у 2019 році, вони змогли “обігнати” за точністю такі відомі мережі, як ResNet, Inception, Xception та DenseNet. Ба більше, авторам вдалося розробити легковажні (lightweight англ.) мережі, що містять у декілька разів менше тренованих параметрів, ніж зазначені вище моделі.

Model	Top-1 Acc.	Top-5 Acc.	#Params	Ratio-to-EfficientNet	#FLOPs	Ratio-to-EfficientNet
EfficientNet-B0	**77.1%**	**93.3%**	**5.3M**	**1x**	**0.39B**	**1x**
ResNet-50 (He et al., 2016)	76.0%	93.0%	26M	4.9x	4.1B	11x
DenseNet-169 (Huang et al., 2017)	76.2%	93.2%	14M	2.6x	3.5B	8.9x
EfficientNet-B1	**79.1%**	**94.4%**	**7.8M**	**1x**	**0.70B**	**1x**
ResNet-152 (He et al., 2016)	77.8%	93.8%	60M	7.6x	11B	16x
DenseNet-264 (Huang et al., 2017)	77.9%	93.9%	34M	4.3x	6.0B	8.6x
Inception-v3 (Szegedy et al., 2016)	78.8%	94.4%	24M	3.0x	5.7B	8.1x
Xception (Chollet, 2017)	79.0%	94.5%	23M	3.0x	8.4B	12x
EfficientNet-B2	**80.1%**	**94.9%**	**9.2M**	**1x**	**1.0B**	**1x**
Inception-v4 (Szegedy et al., 2017)	80.0%	95.0%	48M	5.2x	13B	13x
Inception-resnet-v2 (Szegedy et al., 2017)	80.1%	95.1%	56M	6.1x	13B	13x
EfficientNet-B3	**81.6%**	**95.7%**	**12M**	**1x**	**1.8B**	**1x**
ResNeXt-101 (Xie et al., 2017)	80.9%	95.6%	84M	7.0x	32B	18x
PolyNet (Zhang et al., 2017)	81.3%	95.8%	92M	7.7x	35B	19x
EfficientNet-B4	**82.9%**	**96.4%**	**19M**	**1x**	**4.2B**	**1x**
SENet (Hu et al., 2018)	82.7%	96.2%	146M	7.7x	42B	10x
NASNet-A (Zoph et al., 2018)	82.7%	96.2%	89M	4.7x	24B	5.7x
AmoebaNet-A (Real et al., 2019)	82.8%	96.1%	87M	4.6x	23B	5.5x
PNASNet (Liu et al., 2018)	82.9%	96.2%	86M	4.5x	23B	6.0x
EfficientNet-B5	**83.6%**	**96.7%**	**30M**	**1x**	**9.9B**	**1x**
AmoebaNet-C (Cubuk et al., 2019)	83.5%	96.5%	155M	5.2x	41B	4.1x
EfficientNet-B6	**84.0%**	**96.8%**	**43M**	**1x**	**19B**	**1x**
EfficientNet-B7	**84.3%**	**97.0%**	**66M**	**1x**	**37B**	**1x**
GPipe (Huang et al., 2018)	84.3%	97.0%	557M	8.4x	-	-

Рисунок 3.2 – ілюстрація результатів мереж EfficientNet на класифікації зображень з датасету ImageNet. Порівняння точностей, кількості тренованих параметрів та операцій з плаваючою комою FLOPs з іншими мережами. Взято з таблиці 2 [25].

	Comparison to best public-available results						Comparison to best reported results					
	Model	Acc.	#Param	Our Model	Acc.	#Param(ratio)	Model	Acc.	#Param	Our Model	Acc.	#Param(ratio)
CIFAR-10	NASNet-A	98.0%	85M	EfficientNet-B0	98.1%	4M (21x)	†Gpipe	**99.0%**	556M	EfficientNet-B7	98.9%	64M (8.7x)
CIFAR-100	NASNet-A	87.5%	85M	EfficientNet-B0	88.1%	4M (21x)	Gpipe	91.3%	556M	EfficientNet-B7	**91.7%**	64M (8.7x)
Birdsnap	Inception-v4	81.8%	41M	EfficientNet-B5	82.0%	28M (1.5x)	GPipe	83.6%	556M	EfficientNet-B7	**84.3%**	64M (8.7x)
Stanford Cars	Inception-v4	93.4%	41M	EfficientNet-B3	93.6%	10M (4.1x)	‡DAT	**94.8%**	-	EfficientNet-B7	94.7%	-
Flowers	Inception-v4	98.5%	41M	EfficientNet-B5	98.5%	28M (1.5x)	DAT	97.7%	-	EfficientNet-B7	**98.8%**	-
FGVC Aircraft	Inception-v4	90.9%	41M	EfficientNet-B3	90.7%	10M (4.1x)	DAT	92.9%	-	EfficientNet-B7	**92.9%**	-
Oxford-IIIT Pets	ResNet-152	94.5%	58M	EfficientNet-B4	94.8%	17M (5.6x)	GPipe	**95.9%**	556M	EfficientNet-B6	95.4%	41M (14x)
Food-101	Inception-v4	90.8%	41M	EfficientNet-B4	91.5%	17M (2.4x)	GPipe	93.0%	556M	EfficientNet-B7	**93.0%**	64M (8.7x)
Geo-Mean						**(4.7x)**						**(9.6x)**

†GPipe (Huang et al., 2018) trains giant models with specialized pipeline parallelism library.
‡DAT denotes domain adaptive transfer learning (Ngiam et al., 2018). Here we only compare ImageNet-based transfer learning results.
Transfer accuracy and #params for NASNet (Zoph et al., 2018), Inception-v4 (Szegedy et al., 2017), ResNet-152 (He et al., 2016) are from (Kornblith et al., 2019).

Рисунок 3.3 – порівняння мереж сімейства EfficientNet з моделями Inception, ResNet, NasNet, DAT та GPipe в роботі з різними датасетами. Взято з таблиці 5 [25].

Основним принципом, що допоміг моделям досягти настільки значущих результатів, є складене масштабування (compound scaling англ.). Це – принцип автоматичної адаптації трьох головних параметрів ЗНМ – глибини (depth), ширини (width) та розширення (resolution). Глибина позначає кількість шарів із вагами у моделі, ширина – кількість каналів карт ознак у шарах згортки, розширення – розмір вхідних зображень. Як пояснюють дослідники [25], суттєвою проблемою тогочасних ЗНМ було те, що вони масштабують тільки одну з даних метрик. Такі мережі або сягають великих обсягів параметрів за рахунок великої глибини, або втрачають у точності за рахунок великої ширини та малої глибини.

Складене масштабування оперує одразу трьома метриками, шо дозволяє EfficientNet B0-7 досягати якісної точності при малій кількості параметрів та значно менших обчислювальних потужностях. Це досягається через наступну формулу [25]:

$$\begin{aligned} \text{depth: } & d = \alpha^\phi \\ \text{width: } & w = \beta^\phi \\ \text{resolution: } & r = \gamma^\phi \\ \text{s.t. } & \alpha \cdot \beta^2 \cdot \gamma^2 \approx 2 \\ & \alpha \geq 1, \beta \geq 1, \gamma \geq 1 \end{aligned}$$

Рисунок 3.4 – демонстрація методу складеного масштабування. Формула 3 [25].

У формулі α – коефіцієнт масштабування глибини, □ – ширини, γ –розширення. Такі величини визначають частку ресурсів, розподілених на масштабування

відповідної метрики. Складений коефіцієнту φ уніфіковує розширення цих метрик мережі, який позначає, скільки ресурсів для масштабування доступно. Розробники EfficientNet обмежили добутку коефіцієнтів α, β, γ ≈ 2 через емпіричне спостереження: розширення метрик мережі збільшує значення FLOPs у $(\alpha \cdot \beta \cdot \gamma)^{\varphi}$ разів [25]. Таким чином для будь-якого φ FLOPs збільшуться близько в 2^{φ} рази.

Головними «архітектурними блоками» ЗНМ EfficientNet є мобільний вузький згортковий шар MBConv (Mobile Inverted Bottleneck Convolution) [26] та механізм уваги Squeeze-and-Excitation (SE) [27].

MBConv є шаром, створеним на базі вузької залишкового блоку (Inverted Residual block англ.) із інтеграцією інвертованих залишків (Inverted residuals англ.).

Input	Operator	Output
$h \times w \times k$	1x1 conv2d , ReLU6	$h \times w \times (tk)$
$h \times w \times tk$	3x3 dwise s=s, ReLU6	$\frac{h}{s} \times \frac{w}{s} \times (tk)$
$\frac{h}{s} \times \frac{w}{s} \times tk$	linear 1x1 conv2d	$\frac{h}{s} \times \frac{w}{s} \times k'$

Рисунок 3.5 – будова шару MBConv, де s є кроком (stride) величина t є фактором розширення k каналів до k'. Взято з таблиці 1 [26].

З рисунку 3.5 вбачається, що перший шар застосовує 1×1 конволюції розширює глибину карт ознак, збільшуючи їх кількість каналів в t разів. ReLU6 – це активаційна функція, що додає стабільності при низькоточних обчисленнях. Другим шаром є глибинна згортка 3×3 (Depthwise convolution англ.), де кожна карта ознак обробляється окремим ядром на противагу звичайній згортці, коли кожне ядро проходить усі карти ознак – ширина й висота карт ознак ділиться на s. Третій шар стискає карти ознак назад до розміру k'. Цей шар є лінійним ботлнеком, бо активація не застосовується на цьому етапі задля збереження важливих ознак.

Squeeze-and-Excitation [27] є уважним механізмом, що вдосконалює вилучення ознак, застосовуючи поканальну (channel-wise англ.) оптимізацію через

проектування взаємозв'язків між каналами. Механізм виконує дві операції: стиснення та уточнення. Стиснення стискає вхідну карту ознак до дескриптора каналів, що ловить просторову інформацію з карти. Етап уточнення калібрує дескриптори та вивчає найбільш важливі значення ваг у каналах. Це досягається передачею дескрипторів повнозв'язаному шару ЗНМ, що дає на виході ваги, що застосовуються до калібрування каналів у картах.

3.3.2. Вибір моделі для експериментів

Оптимальна модель EfficientNet була обрана емпіричним шляхом. Для вибору конкретної версії було проведено тестове розгортання усіх моделей B0-7 на власному датасеті з такими налаштуваннями: 13 класів, 5 навчальних епох, всі шари є тренованими, навчальний рейтинг у 0.001, бал затримки ваг (weight decay) в 0.0001:

Модель	Обсяг моделі (MB)	GFLOPs	К-сть ваг	Точність	Час тренування (с)
B0	188.99	0.41	4 024 201	92.02%	490.6
B1	310.52	0.66	6 529 837	92.44%	504.6
B2	392.78	0.7	7 719 311	92.99%	510.9
B3	**666.56**	**1.02**	**10 716 213**	**93.84%**	**548.9**
B4	1 373.61	1.58	17 571 925	95.66%	747.1
B5	2 739.47	2.46	28 367 421	83.05%	1105.6
B6	4 616.92	3.49	40 765 669	89.91%	1574.5
B7	8 009.18	5.34	63 820 253	83.33%	2840.6

Таблиця 3.3 – порівняння моделей EfficientNet. Тренування B4, B5 відбувалося на 16 батчах (batch англ.), B6 на 8, B7 на 4 через великі затрати часу. Точність взято з останньої п'ятої епохи.

Тестування відбувалося на персональному комп'ютері з наступними характеристиками:

- процесор: Intel Core I7-10700F 2.90GHz
- відеокарта: GeForce RTX 3060, 3584 ядра CUDA
- 16 Гб ОП ОС
- Win10 x64 22H2

При звичних налаштуваннях розмірів пакетів (batch англ.) 32-16-16 (для тренувальної, валідаційної та тестової підгруп) тренування моделей B5-B7 займало неприпустимо багато часу. Внаслідок зменшення розмірів пакетів в останніх моделях точність упала приблизно на 2-12%. Моделі B4-B6 вимагають великих ресурсів для роботи. Тому було прийнято рішення обрати EfficientNet B3 як оптимальну ЗНМ, що балансує високу точність, малий обсяг пам'яті та часові затрати.

3.4. Експерименти

Усі проведені експерименти є фактично аналізом роботи різних версій EfficientNet B3, їхнього порівняння з мережею, побудованою з нуля. Для порівняння виокремлено два основних напрямки досліджень: уточнена EfficientNet B3 (fine-tuned англ.) проти ЗНМ, побудованої з нуля; встановлення різних рейтингів навчання EfficientNet B3 для повнозв'язаного шару й бекбон-частини – частини ЗНМ, що розташована перед повнозв'язаним шаром (backbone англ.). Також окремо було приділено увагу дослідам щодо впливу ауґментації даних на успішність класифікації указаних версій ЗНМ.

Будівними блоками архітектури EfficientNet B3, як і для інших мереж сімейства, є мобільні інвертовані вузькі згортки MBConv та SE-блоки уваги. Зокрема, в PyTorch реалізації EfficientNet B3 доповнюється блоками Conv2dNormActivation, що складається з трьох шарів – глибинної згортки, пакет-нормалізації та активації SiLU, або Swish. Іншим важливим елементом MBConv шару є стохастична глибина [28], що є регуляризаційною технікою й зменшує глибину мережі шляхом пропуску залишкових (residual) блоків під час прямого поширення з визначеною наперед вірогідністю. Для B3-версії вхідний розмір зображень встановлений як 300×300 [29].

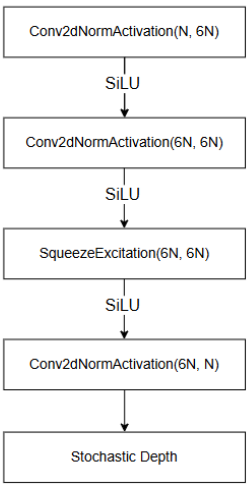


Рисунок 3.6 – структура блоку MBConv у EffNet B3. Значення в дужках є вхідним та вихідним обсягом каналів відповідно.

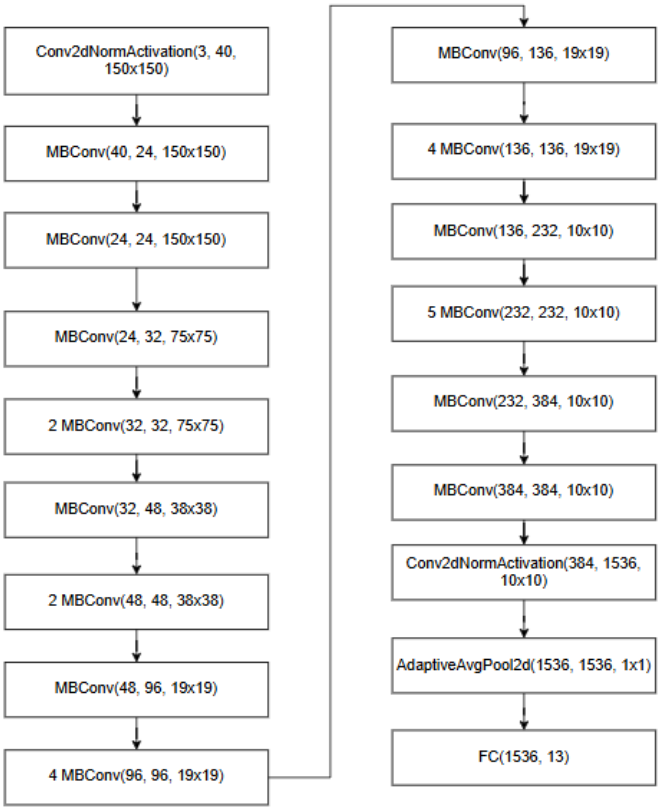


Рисунок 3.7 – архітектура EfficientNet B3 з модифікованим FC-шаром. У дужках позначено вхідну, вихідну кількості каналів та вихідне розширення карт ознак.

Також для порівнянь була спроектована довільна згорткова нейронна мережа. Вона має на меті показати результати тренувань на датасеті «з нуля» без використання жодної попередньо здобутої інформації. Модель спроектована за

аналогією мереж EfficientNet задля малої ємкості мережі та кількості обчислень. Мережа складається з 7 MBConv-шарів, ствола (stem англ.) та повнозв'язаного шару. Ствол є початковим шаром у мережі й призначений для видобутку загальних ознак й формуванню відповідних карт ознак; він складається з конволюції 3×3, пакет-нормалізації, активації SiLU. Наступні MBConv-шари розширюють канали карт ознак через фактор розширення, проводять глибинну та поточкову згортку. У разі однакової кількості вхідних та вихідних каналів, до вихідних карт ознак додаються вхідні для покращення градієнтного спуску. У кінці розташовані ще один ствол, згладжування, далі проводиться вимикання випадкових нейронів, або регуляризація, і повнозв'язаний шар.

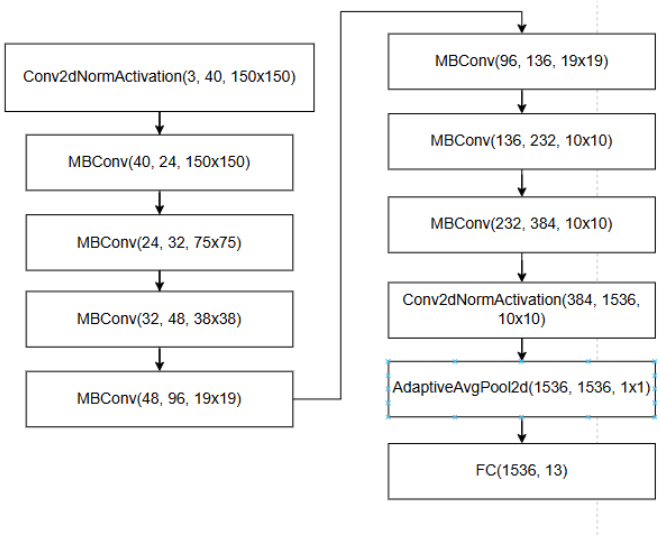


Рисунок 3.8 – архітектура ЗНМ, тренованої з нуля.

Усі результати отримані під час запуску всіх модифікації мереж з використанням наступних характеристик:

- оптимізатор: Adam
- рейтинг навчання (learning rate): $1e^{-3}$
- затримка ваг (weight decay): $1e^{-4}$
- період тренування: 10 епох
- аугментація: випадкові оберти на 30 градусів та горизонтальні оберти – RandomRotation(30), RandomHorizontalFlip()

Також для оцінки ефективності на тестовій вибірці було застосовано інші підходи оцінювання, такі як precision, recall, F1-значення, матриця похибок. Матриця похибок – таблиця істинної та хибної класифікації зображень відносно заданих класів. При такій оцінці оперують поняттями як істинно позитивна TP, істинно негативна TN, хибна позитивна FP та хибна негативна FN класифікації зразків. Позитивний та негативний класи в цьому випадку відображають 2 довільних класи з одного домену, які є вірно (true) або хибно (false) розпізнаними.

Відповідно існують такі метрики оцінки, які використовують на тестових даних:

- влучність (precision) – міра здатності моделі вірно визначати зразки позитивного класу серед усіх позитивних:

$$Precision = \frac{TP}{TP+FP} \qquad (3.5)$$

де TP – значення істинно позитивних зразків;
FP – значення хибно позитивних зразків;

- повнота (recall) – здатність моделі вірно визначити позитивні зразки серед усіх вірно визначених зразків:

$$Recall = \frac{TP}{TP+FN} \qquad (3.6)$$

де FN – хибно негативні зразки;

- F1-значення – гармонійне середнє влучності та повноти. Об’єктивна метрика оцінки успішності моделі, особливо у випадку незбалансованої вибірки даних:

$$F1 = 2\frac{Precision \cdot Recall}{Precision + Recall} \qquad (3.7)$$

3.4.1. Повне та часткове тонке налаштування. Порівняння із ЗНМ, побудованої з нуля

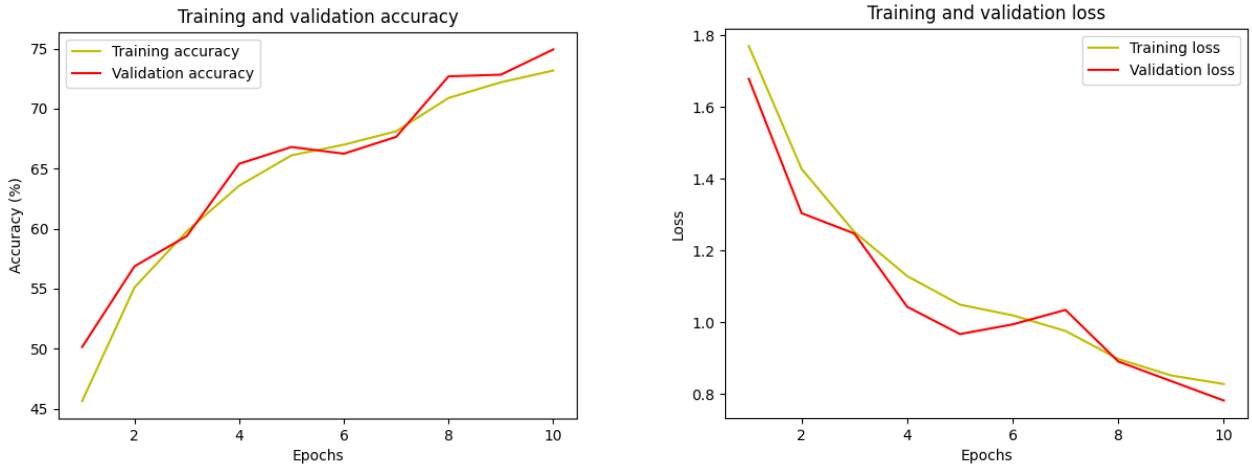
Тонке налаштування (fine-tuning англ.) [30] є розгалуженням у техніці перенесеного навчання, яке призначене адаптувати передналаштовану мережу на якісну роботу у визначеному домені. Цей метод розподіляється на повне тонке налаштування та часткове (full та partial fine-tuning англ.). В цій частині експериментів представлено 3 моделі для тренувань. Перша це повністю уточнена, або розморожена модель EfficientNet B3, в якій всі ваги є тренованими. У другій версії більшість backbone-частини – шарів ЗНМ, що знаходяться перед повнозв'язаним шаром – є замороженою, тобто ваги цих шарів становляться не тренованими й зберігають оригінальні значення. Третя модель це власноруч побудована ЗНМ без використання перенесеного навчання взагалі. Часткове уточнення полягає у відключенні малої кількості шарів від тренування. У частково уточненій версії лишилися тренованими повнозв'язаний та три останніх шари – шари об'єднання AdaptiveAvgPool2d() та 2 MBConv; решта верхніх шарів були «заморожені». У двох модифікаціях EfficientNet B3 було замінено значення вихідного вектору вірогідностей на 13, відповідно до кількості класів у датасеті.

Тип моделі	GFLOPs	Обсяг моделі (MB)	Треновані параметри
Full fine-tuned EffNet B3	1.93	666.56	10 716 213
Partial fine-tuned EffNet B3	1.93	666.56	3 897 095
From scratch CNN	0.59	254.67	2 035 741

Таблиця 3.4 – Порівняння обсягу та складності двох уточнених версій EfficientNet B3 та тренованої з нуля.

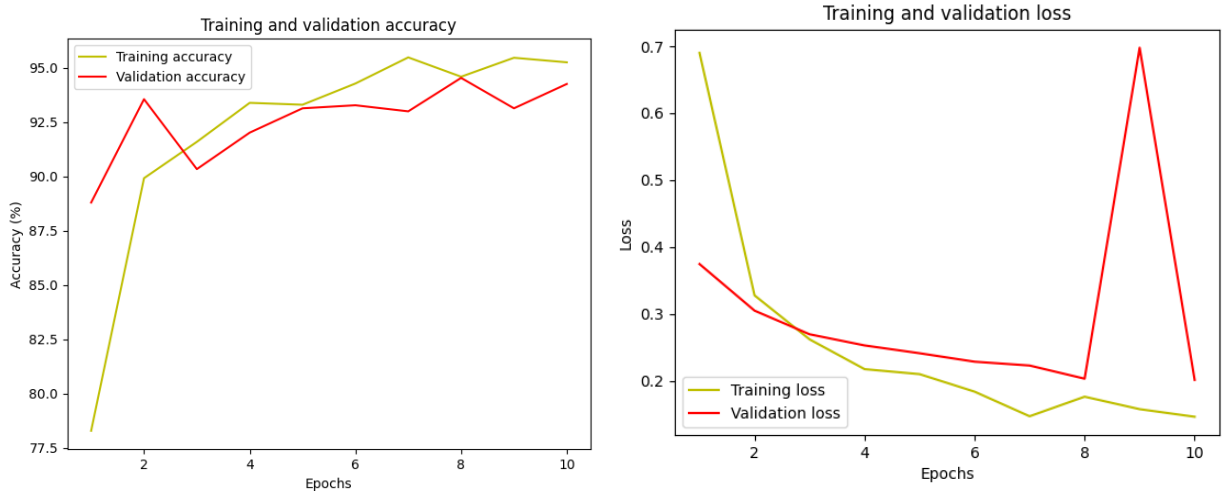
Тип	Тренувальна точність (%)	Валідаційна точність (%)	Трен. похибка	Валід. похибка	Час виконання (с)
Full fine-tuned	95.26	**94.26**	0.14	0.20	1340.01
Partial fine-tuned	**95.55**	93.97	**0.13**	**0.18**	**1030.6**
From scratch	73.17	74.93	0.82	0.78	1079.74

Таблиця 3.5 – показники успішності повністю розмороженої та частково замороженої EfficientNet B3, навченої з нуля ЗНМ. Значення взяті з останньої 10-ї епохи.



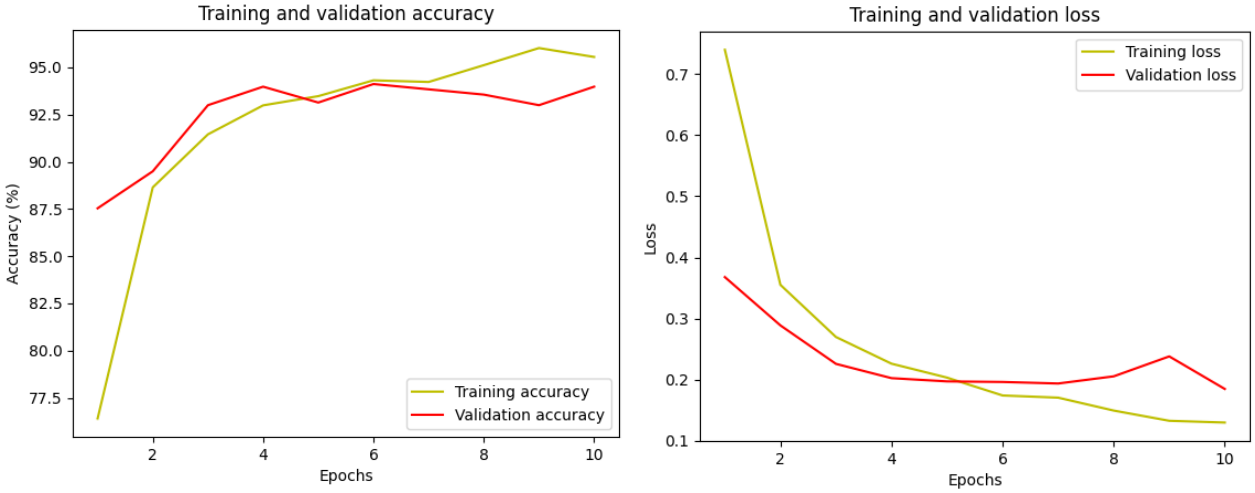
(a) (b)

Рисунок 3.9 – графік зміни точності (а) та похибки (б) на тренувальній, валідаційній вибірці для моделі, тренованої з нуля



(a) (b)

Рисунок 3.10 – графік зміни точності (а) та похибки (б) на тренувальній, валідаційній вибірці для повністю розмороженої EfficientNet B3.

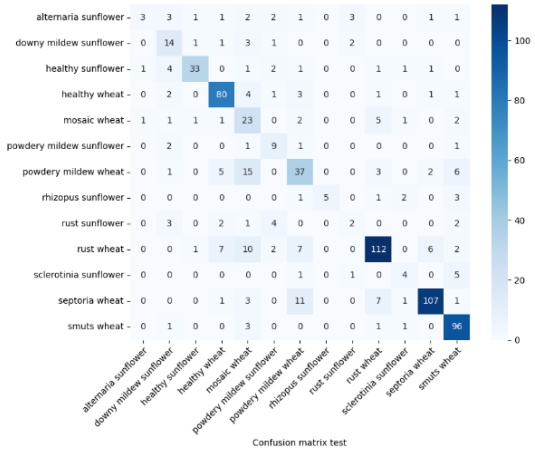


(а) (б)

Рисунок 3.11 – графік зміни точності (а) та похибки (б) на тренувальній, валідаційній вибірці для частково замороженої EfficientNet B3.

Застосування двох підходів тонкого налаштування на тестовій вибірці продемонстровано на побудові матриці похибок – таблиці істинної та хибної класифікації зображень відносно заданих класів:

(а) (б)



(c)

Рисунок 3.12 – матриці похибок для повністю розмороженої EffNet B3 (a), частково замороженої EffNet B3 (b) та тренованої з нуля моделі (c)

Тип	Влучність	Повнота	F1-бал
Full fine-tuned	0.93	0.93	0.93
Partial fine-tuned	**0.94**	**0.94**	**0.94**
From scratch	0.76	0.76	0.75

Таблиця 3.6 – Таблиця показників моделі, обчислених з використанням матриці похибок.

Фінальні розрахунки тестових метрик є зваженими відносно всіх класів та проведені за наступною формулою:

$$X = \frac{\sum_{i=1}^{C}(X_i \cdot N_i)}{N} \qquad (3.6)$$

де C – кількість класів у датасеті;

X, X_i – обчислюваний загальний параметр влучності, повноти або F1-балу та параметр для i-го класу;

N, N_i – обсяг зразків у всій вибірці та для i-го класу відповідно.

У підсумку частково заморожена мережа демонструє кращу успішність ніж попередня за рахунок використання вивчених ваг у верхніх заморожених шарах. З таблиці 3.5 обидві конфігурації досягли валідаційної точності рівня 94%, проте частково заморожена досягла такого результату приблизно на 23%

швидше. Неодноразові запуски двох моделей можуть демонструвати результати навіть більші в частково замороженої, ніж у першої. Зокрема, в частково уточненій версії спостерігається краща конвергенція тренувальної та валідаційної точностей. Показовим є те, що в другій конфігурації F1-бал є незначно, але вищим, ніж у першій, що демонструє кращу пристосованість до нових даних, незважаючи на нерівномірний розподіл зображень по класах у датасеті. Що стосується третьої моделі, то вона демонструє достатньо хорошу конвергенцію тренувальної та валідаційної точностей, проте самі точності є значно нижчими, ніж у претренованих мережах. Точність змогла би вирости ще при довшому навчанні та збільшенні епох, але очевидно те, що вона не може досягати високих показників за відносно малий час, так як на це здатна частково уточнена EfficientNet B3. Отже, частково уточнена модель досягає однаково високих показників, що й повністю уточнена, і перевершує ту, що навчена з початку, за найменший час тренування.

3.4.2. Встановлення різних гіперпараметрів

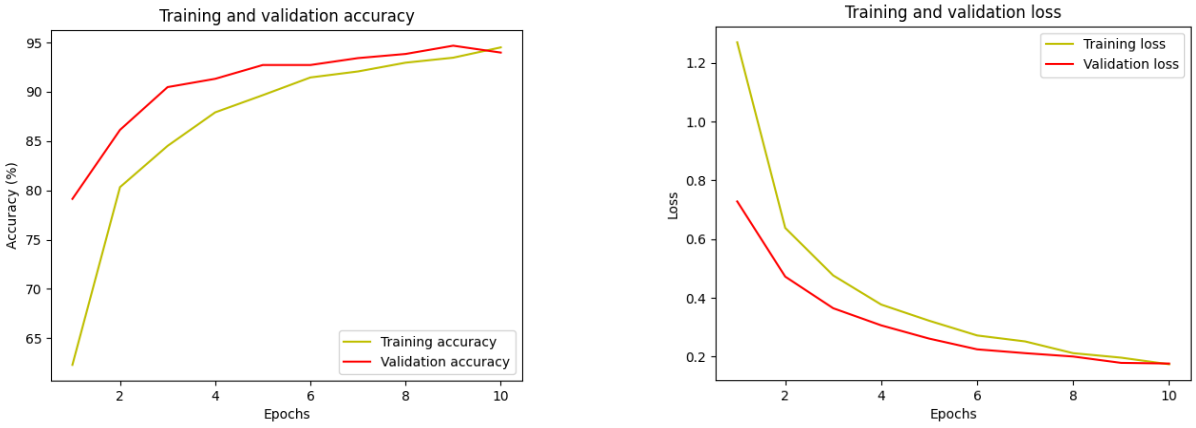
Другий напрямок дослідів розкриває застосування диференційних рейтингів навчання та затримки ваг у мережі. Підхід полягає у встановленні менших показників для бекбон-частини та більших для повнозв’язаного шару. Згортки в ранніх шарах, зазвичай, видобувають загальні риси з карт ознак такі як лінії або кути. Тому ваги в таких шарах краще відкориговані, відповідно, вимагають меншої оптимізації. У цьому розділі порівняна ефективність претренованої мережі EfficientNet B3 з використанням наступних характеристик:

Тип оптимізатора	Рейтинг навчання	Затримка ваг
Single Adam	$1e^{-3}$	$1e^{-4}$
FC Adam	$1e^{-3}$	$1e^{-4}$
Backbone Adam (1)	$1e^{-5}$	$1e^{-6}$
Backbone Adam (2)	$1e^{-4}$	$1e^{-5}$

Таблиця 3.7 – показники рейтингу навчання та затримки ваг для різних налаштувань оптимізатора Adam.

Перша версія є повністю уточненою моделлю, що використовує єдині гіперпараметри для всіх шарів. Друга версія застосовує стандартні значення для

повнозв’язаного шару, для опорної (backbone) частини ЗНМ встановлений оптимізатор Backbone Adam (1). У третій конфігурації для опорної частини встановлено оптимізатор Backbone Adam (2); дана версія має на меті показати реакцію мережі на більші значення гіперпараметрів для бекбону.



(a) (b)

Рисунок 3.13 – графік точності (a) та похибки (b) тренувальної, валідаційної вибірок для мережі з Backbone Adam (1).

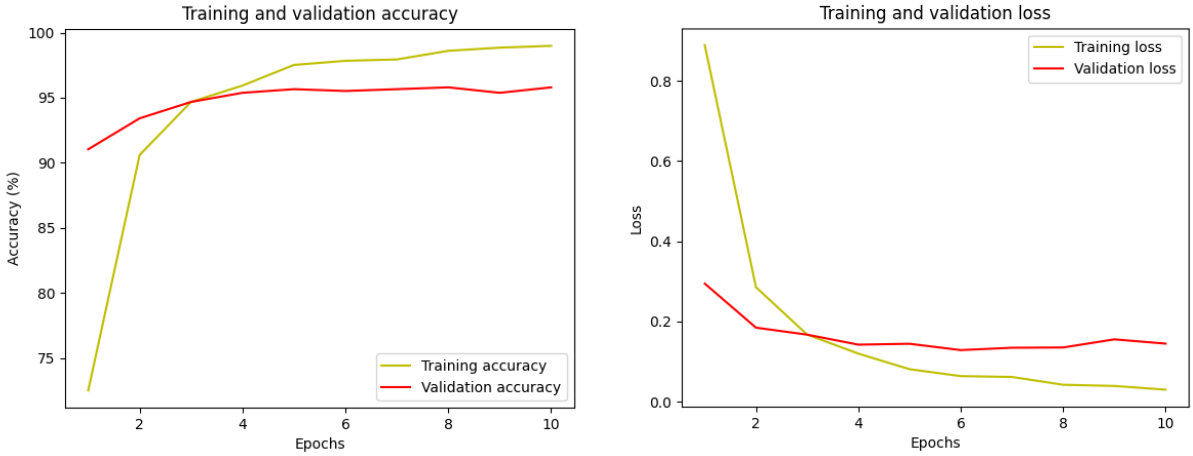
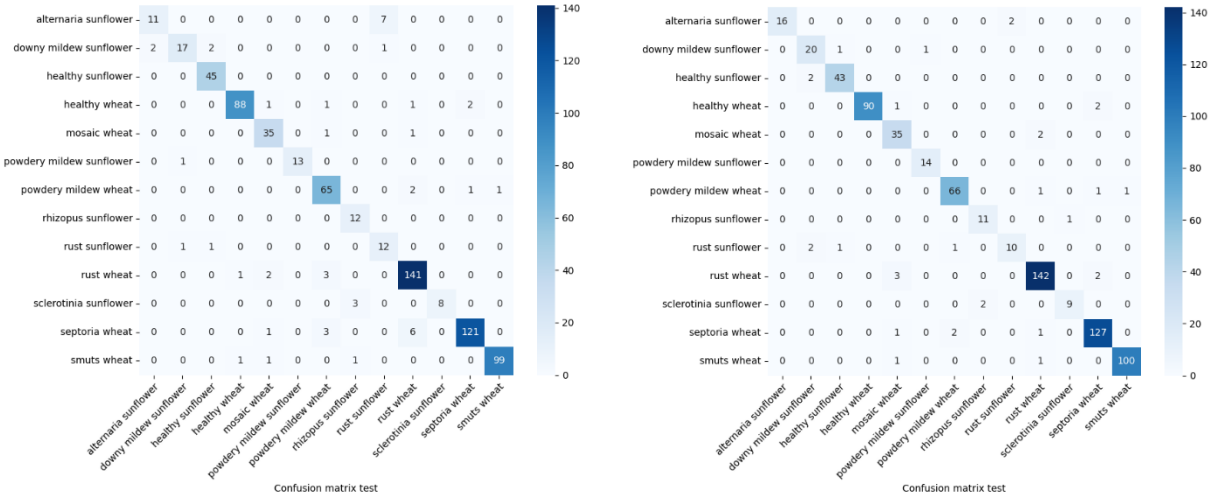


Рисунок 3.14 – графік точності (a) та похибки (b) тренувальної, валідаційної вибірок для мережі з Backbone Adam (2).

Тип	**Тренувальна точність (%)**	**Валідаційна точність (%)**	**Трен. похибка**	**Валід. похибка**	**Час виконання (с)**
Один оптимізатор (full fine-tuned)	95.26	94.26	0.14	0.20	**1340.01**
Різні оптимізатори					

(1)	94.51	93.98	0.17	0.18	1345.98
Різні оптимізатори (2)	**98.98**	**95.79**	**0.02**	**0.14**	1352.61

Таблиця 3.8 – значення точності та похибки тренувальної, валідаційної вибірок для мереж з єдиним та двома оптимізаторами.



(a) (b)

Рисунок 3.15 – матриця конфузій тестової вибірки для мереж з двома оптимізаторами версії (1) на (а) та версії (2) на (б).

Тип	Влучність	Повнота	F1-бал
Single optimizer	0.93	0.93	0.93
Different optimizers (1)	0.94	0.93	0.93
Different optimizers (2)	**0.96**	**0.96**	**0.96**

Таблиця 3.9 – таблиця показників влучності, повноти та F1-балу на основі тестової вибірки для мереж з єдиним та двома оптимізаторами.

Основна перевага, що виділяє другу конфігурацію від решти двох, це — значно більша конвергенція точності класифікації та похибки в тренувальній та валідаційній вибірці. З різними рейтингами навчання та затримками ваг прогрес навчання EfficientNet B3 є «плавнішим» і послідовним. На останній епосі значення точностей та похибок практично однакові, різниця складає всього 0,53% та 0,1% відповідно. Стосовно третьої версії (параметри $1e^{-4}$ та $1e^{-5}$) її точності та похибки збігаються менше. Однак окремо точності є більшими, а

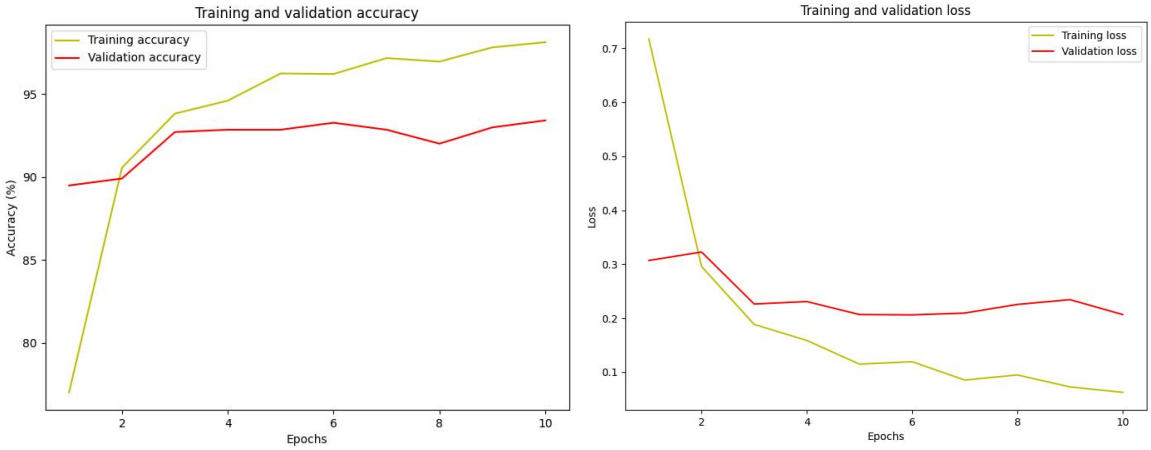
похибки меншими що для тренувальної, що для валідаційної підвибірки. Це можна сказати й про оцінку роботи на тестовій частині. Варто зауважити, що третя версія тренувалася найдовше з усіх досліджених, хоча затримка є невелика – близько 12 секунд відносно першої моделі та 7 — відносно другої. У підсумку можна сказати, що конфігурацію зі значеннями рейтингу навчання та затримки ваг $1e^{-4}$ та $1e^{-5}$ за відносно однаковий час досягає найвищої точності класифікації зображень попри найвищі затрати часу.

3.4.3. Робота моделей EfficientNet B3 та побудованої з нуля без ауґментації даних

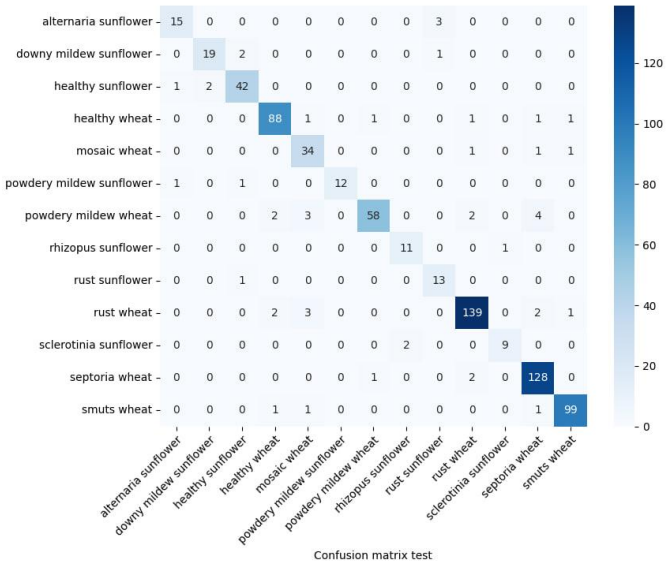
Для демонстрації впливу ауґментації даних окремо були проведені запуски спроектованих моделей без ауґментації даних. Тобто, попередньо навчальна вибірка піддавалася лиш зміні розширення до 300×300 та базовій нормалізації RGB каналів зображень із ImageNet. Моделі використовували ідентичні гіперпараметри, що й у попередніх конфігураціях:

- кількість тренувальних епох: 10
- рейтинг навчання для уточнених моделей: 10^{-3}
- затримка ваг для уточнених моделей: 10^{-4}

Для моделей з різними гіперпараметрами – моделі з Backbone Adam (1) та Backbone Adam (2) – для повнозв’язаного шару та бекбон-частини встановлені такі самі значення гіперпараметрів, що й у таблиці 3.7.

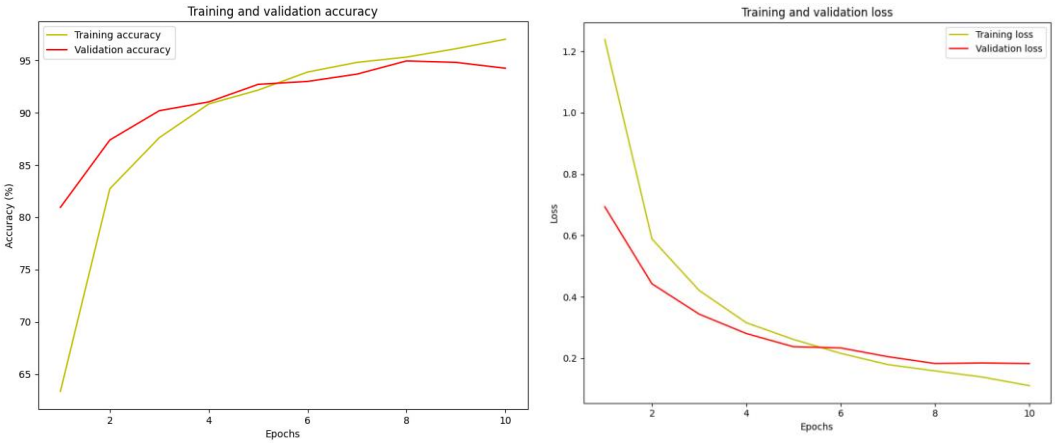


(a) (b)

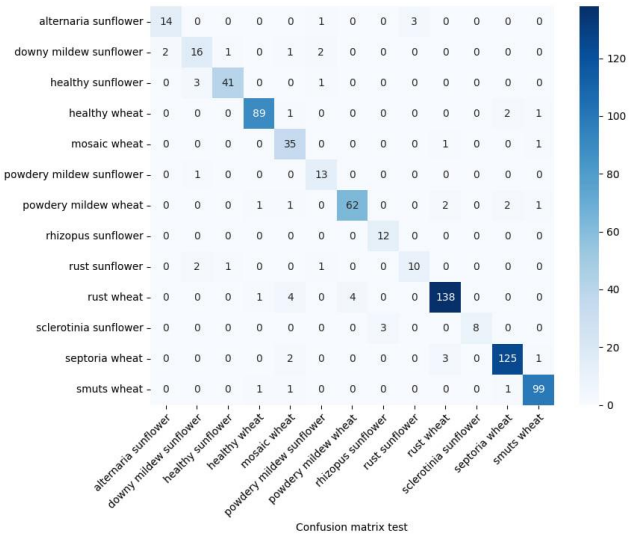


(c)

Рисунок 3.16– графіки точності (a), похибки (b) та матриці конфузій (c) частково уточненої моделі.



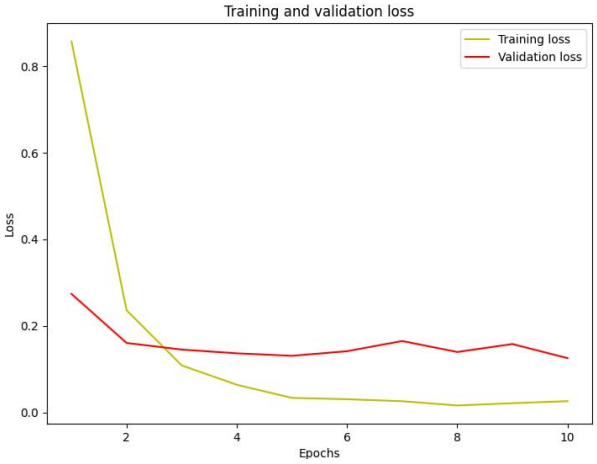
(a) (b)



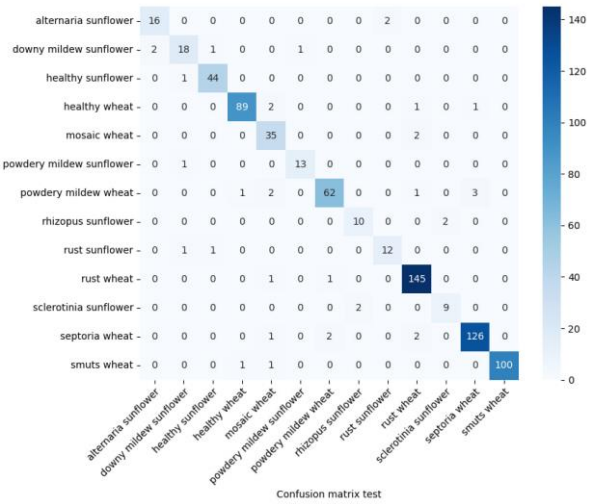
(c)

Рисунок 3.17 – графіки точності (a), похибки (b) та матриці конфузій (c) моделі з оптимізатором Backbone Adam (1).

(a)

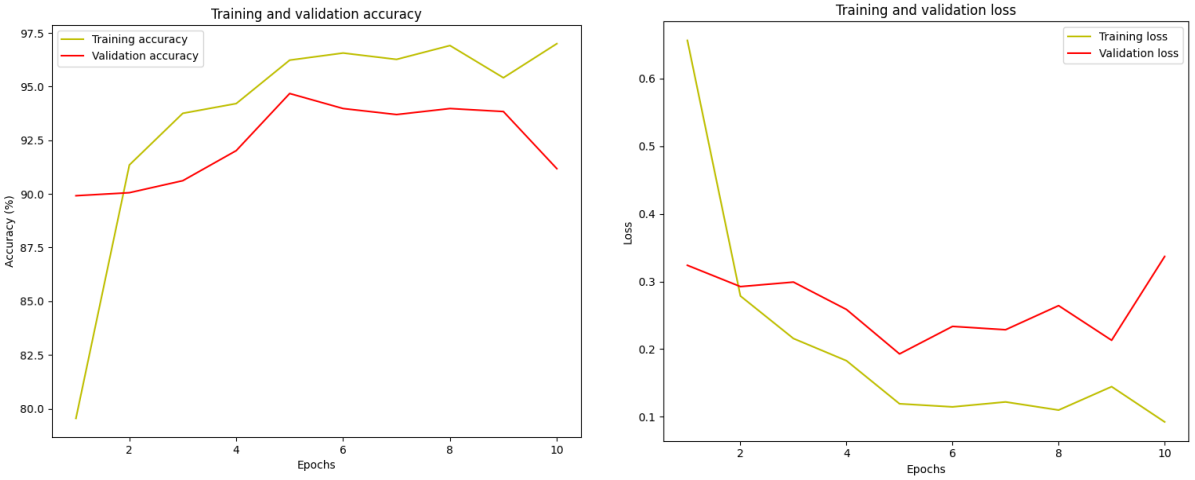


(b)

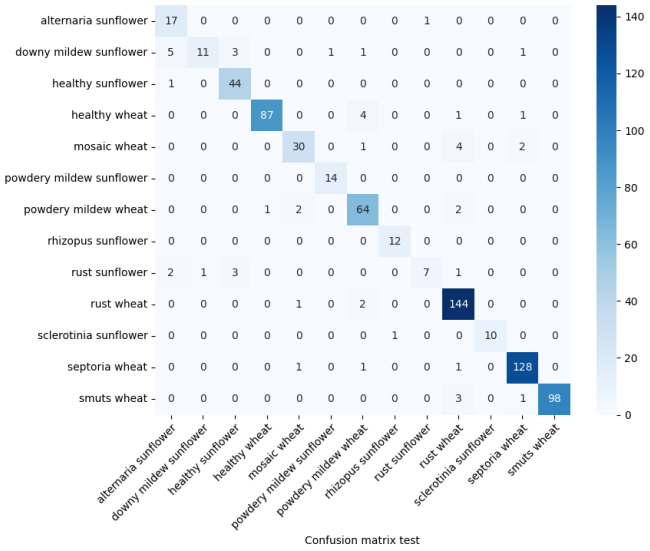


(c)

Рисунок 3.18 – графіки точності (a), похибки (b) та матриці конфузій (c) моделі з оптимізатором Backbone Adam (2).

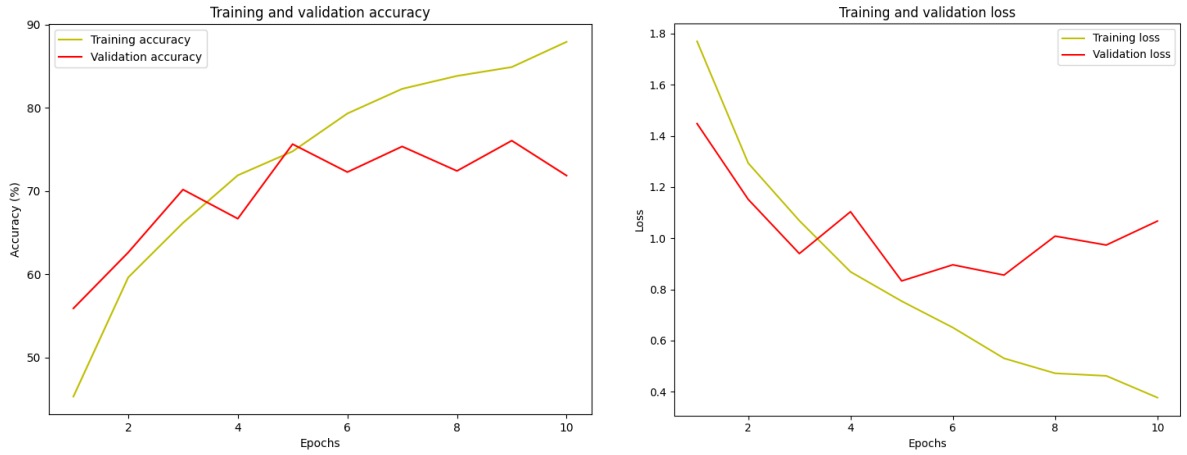


(a) (b)

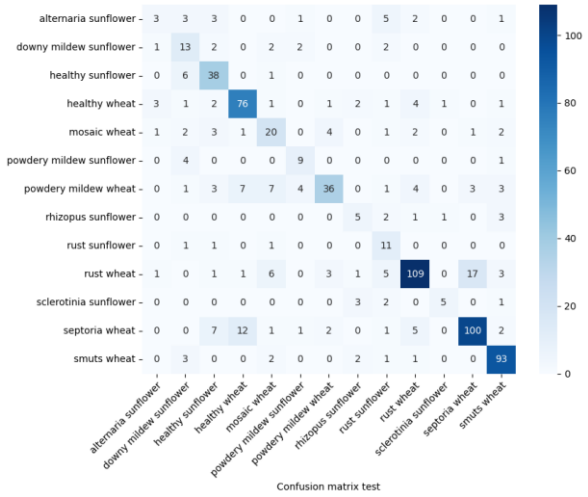


(c)

Рисунок 3.19 – графіки точності (a), похибки (b) та матриці конфузій (c) повністю уточненої моделі EfficientNet B3.



(a) (b)



(c)

Рисунок 3.20 – графіки точності (a), похибки (b) та матриці конфузій (c) моделі, побудованої з нуля.

	З ауґментацією			**Без ауґментації**		
Тип мережі	**Трен. точність**	**Валід. точність**	**Різниця (%)**	**Трен. точність**	**Валід. точність**	**Різниця (%)**
Повністю уточнена	95.26	**94.26**	**1**	**96.99**	91.18	5.81
Частково уточнена	95.55	**93.97**	**1.58**	**98.13**	93.41	4.72
Модель з Backbone Adam (1)	94.51	93.98	**0.53**	**97.03**	**94.26**	2.77
Модель з Backbone Adam (2)	98.98	**95.79**	**3.19**	**99.23**	95.52	3.71

Побудована з нуля	73.17	**74.93**	**1.76**	**87.92**	71.84	16.08

Таблиця 3.10 – таблиця показників тренувальної та валідаційної точностей моделей із застосуванням ауґментації даних та без неї.

	З ауґментацією			Без ауґментації		
Тип мережі	Трен. похибка	Валід. похибка	Різниця	Трен. похибка	Валід. похибка	Різниця
Повністю уточнена	0.14	**0.20**	**0.06**	**0.09**	0.34	0.25
Частково уточнена	0.13	**0.18**	**0.05**	**0.06**	0.2	0.14
Модель з Backbone Adam (1)	0.17	**0.18**	**0.01**	**0.11**	**0.18**	0.07
Модель з Backbone Adam (2)	**0.02**	0.14	0.12	**0.02**	**0.11**	**0.09**
Побудована з нуля	0.82	**0.78**	**0.04**	**0.37**	1.06	0.69

Таблиця 3.11 – таблиця показників тренувальної та валідаційної похибок моделей із застосуванням ауґментації даних та без неї.

	З ауґментацією			**Без ауґментації**		
Тип мережі	**Влучність**	**Повнота**	**F1-бал**	**Влучність**	**Повнота**	**F1-бал**
Повністю уточнена	0.93	0.93	0.93	0.93	0.93	0.93
Частково уточнена	**0.94**	**0.94**	**0.94**	**0.94**	0.93	0.93
Модель з Backbone Adam (1)	**0.94**	0.93	0.93	0.93	0.93	0.93
Модель з Backbone Adam (2)	**0.95**	**0.95**	**0.95**	**0.95**	**0.95**	**0.95**
Побудована з нуля	**0.76**	**0.76**	**0.75**	0.74	0.72	0.73

Таблиця 3.12 – таблиця показників влучності, повноти та F1-балу моделей із застосуванням ауґментації даних та без неї на основі тестової вибірки

Результати навчання показують, що без ауґментації моделі на основі EfficientNet B3 мають більшу розбіжність між тренувальними та валідаційними метриками

–для точностей у межах близько 3-6% та для похибок – блзько 0,1-0,25. Ба більше, мережа, побудована з нуля, має явні ознаки перенавчання – різниця між тренувальною та валідаційною точностями, похибками становить близько 16% та 0,69. Очікувано для моделей, що використовували тонке налаштування, та моделі, побудованої з нуля, спостерігається тенденція збільшення тренувальних та зменшення валідаційних точностей та похибок.

На мережі з різними гіперпараметрами для повнозв'язаного шару та бекбон-частини ауґментація не впливає значним чином, оскільки перенавчання долається власне зменшенням гіпепараметрів для бекбон-частини, внаслідок ваги нейронів оновлюються з меншою зміною, точності на епохах прогресують «плавніше». Встановлення ауґментації до таких мереж може радше зашкодити навчанню через ще повільніше оновлення ваг. Тим не менш, за результатами тестування мережі з ауґментацію демонструють кращу успішність, аніж ідентичні без ауґментації.

Графіки навчання підтверджують, що модель із оптимізатором Adam версії 1 (Backbone Adam 1) – рейтинг навчання та затримка ваг складають 10^{-5} та 10^{-6} для backbone-частини та 10^{-3} та 10^{-4} для повнозв'язаного шару відповідно – найкраще протидіє перенавчанню. Тренування відбувається найбільш послідовно та плавно. До того ж, ця мережа має найнижчі серед інших мереж різниці між тренувальною та валідаційною точностями та похибками. Однак модель із оптимізатором Adam версії 2 (Backbone Adam 2) – для backbone-частини рейтинг навчання та затримка ваг складають 10^{-4} та 10^{-5} відповідно – демонструє найкращі абсолютні значення точностей, похибок та тестових метрик влучності, повноти та F1-балу серед усіх інших моделей.

Отже, ауґментація даних позитивно впливає на навчання та роботу ЗНМ – що переднавчених, що побудованих із нуля, – покращує точність класифікації на небачених даних, запобігає перенавченню. Проте у випадку мереж із різними налаштуваннями гіперпараметрів для бекбон-частини та повнозв'язаного шару вона може зашкодити навчанню та погіршити успішність моделей.

Розділ 4. Результати. Подальші покращення

Експерименти показали, що тонке налаштування та диференційні рейтинги навчання є ефективними методиками, що покращують успішність згорткових нейронних мереж в задачі класифікації хвороб рослин. При використанні обох технік претренована мережа EfficientNet B3 досягла високої точності в межах 93-96% та високого F1-балу і діапазоні 0.94-0.96.

Серед версій EfficientNet B3, що використовували тонке налаштування, найефективнішою показала себе частково уточнена її модифікація. У даній модифікації модель все ще використовувала значну набуту інформацію з уже налаштованих нейронів і зуміла досягти таких же високих точностей, що й повністю налаштована, проте вона мала незначну кількість шарів перед повнозв'язаним шаром, що адаптовувалися до нових даних. Окрім цього, частково уточнена модель зробила це на 309.4 секунд або на близько 23% швидше за повністю уточнену та на 49.14 секунд або близько 4.5% швидше ніж за треновану з нуля.

В експериментах зі встановленням різних значень гіперпараметрів найкращі результати отримала модифікація, де рейтинг навчання та затримка ваг для бекбон-частини складали $1e^{-4}$ та $1e^{-5}$ для повнозв'язаного $1e^{-3}$ та $1e^{-4}$. Значення, вказані для обох частин моделі, сприяли більшій дивергенції тренувальної та валідаційної точностей, ніж моделі з меншими значеннями. Тим не менш, обидві вони вищі, ніж у моделей з єдиним оптимізатором та двома, але з меншими гіперпараметрами. Ця версія змогла класифікувати зображення хвороб пшениці та соняшнику з точністю 95.8% на валідаційних даних, з F1-балом у 0.95 на тестових зображеннях.

В експериментах із навчанням конфігурації ЗНМ без ауґментації даних було підтверджено, що це є ефективною технікою підвищення якості класифікації зображень ЗНМ, зокрема, у випадку з недостатньо репрезентативним або незбалансованим датасетом. Ауґментація даних значно зменшує розбіжність

між тренувальною та валідаційною точностями, таким чином усуваючи проблему перенавчання.

Результатами цієї роботи є представлення різних перспективних шляхів імплементації рішення класифікації хвороб агрокультур із застосуванням ЗНМ. Продемонстровані рішення показали високу якість класифікації зображень зі споживанням малого обсягу пам’яті та обчислювальних ресурсів. У порівнянні з подібними дослідженнями, зокрема охарактеризованими в розділі 2, ця робота досліджує можливості імплементації рішення із застосуванням ЗНМ для класифікації хвороб декількох культур. До того ж, датасет містить зразки важливих, хоча й не таких поширених, хвороб пшениці та соняшнику, що не були розглянуті у зазначених роботах та публічно доступних датасетах.

Подальший розвиток дослідження в основному передбачає наступні покращення. Перше — це ще більше покращення точності класифікації. Наступне можливо реалізувати інтеграцією кращих уважних механізмів, вкрапленням генеративних технік, зокрема вказаних у дослідженні [16]. Друге — це розширення датасету. Додавання нових зразків до мінорних класів, що мають малу кількість зразків відносно інших, вкраплення зображень з різними погодними умовами, додавання нових рослин або розширення переліку наявних їх хвороб вирішить проблему незбалансованості вибірки та розширить функціональність, сферу застосування оптимальної моделі для обраної предметної області. У тому числі, збалансований широкий датасет є сильним чинником зростання точності моделі для якісного розпізнавання тої чи іншої хвороби. Також потенційним вектором розвитку дослідження є створення прикладної програми, що є важливим етапом до реалізації бізнес-рішення в агропромисловості.

Висновки

У курсовій роботі було представлено різні методи застосування згорткових нейронних мереж для розпізнавання образів в обраній предметній галузі, а саме — в класифікації хвороб пшениці та соняшнику. Були висвітлені актуальні питання впровадження згорткових нейронних мереж у сільському господарстві, наявні сучасні ефективні рішення. У цій галузь точне розпізнавання є критично важливим фактором для розвитку агробізнесу, оскільки прямо впливає на відповідну врожайність агрокультур. Для України впровадження технологій класифікації хвороб культур, зокрема ЗНМ, є прямим чинником економічного зростання. Тож в експериментах були розглянуті перспективні для такої задачі методи, що показали високі ефективність та потенціал застосування ЗНМ у галузі. Серед імплементованих методик часткове налаштування та інтеграція різних гіперпараметрів для backbone-частини та повнозв’язаного шару ЗНМ показали найбільшу ефективність у навчанні – такі моделі досягли високої точності розпізнавання зображень у 94-96% із низькими показниками втрат. Не менш важливим є те, що рішення є малоємними стосовно обсягу пам’яті, необхідної для їх застосування, та швидкими в навчанні, що робить їх оптимальними для інтеграції в прикладні застосунки.

Список використаних джерел

1. https://customs.gov.ua/web/content/15871?unique=57cb97c3e51a76b0a71c68514836c600474e86b4&download=true

2. Ankile, L.L., Heggland, M.F., & Krange, K. (2020). Deep Convolutional Neural Networks: A survey of the foundations, selected improvements, and some current applications. ArXiv, abs/2011.12960.

3. Ruder S. An overview of gradient descent optimization algorithms. *ArXiv, abs/1609.04747v2*.

4. Diederik P. Kingma, Jimmy Ba: Adam: A Method for Stochastic Optimization. arXiv:1412.6980v9

5. https://www.ibm.com/think/topics/transfer-learning

6. https://www.ibm.com/think/topics/unsupervised-learning

7. Sanghoon Myung, In Huh, Wonik Jang, Jae Myung Choe, Jisu Ryu, Dae Sin Kim, Kee-Eung Kim, Changwook Jeong (2022). PAC-Net: A Model Pruning Approach to Inductive Transfer Learning. arXiv:2206.05703

8. Russakovsky, O., Deng, J., Su, H. et al. ImageNet Large Scale Visual Recognition Challenge. Int J Comput Vis 115, 211–252 (2015). https://doi.org/10.1007/s11263-015-0816-y

9. Tugrul, B.; Elhoucine, E.; Eryigit, R. Convolutional Neural Networks in Detection of Plant Leaf Diseases: A Review. Agriculture 2022, 12, 1192. https://doi.org/10.3390/agriculture12081192

10. Mohanty SP, Hughes DP and Salathé M (2016) Using Deep Learning for Image-Based Plant Disease Detection. Front. Plant Sci. 7:1419. doi: 10.3389/fpls.2016.01419

11. Shi, T., Liu, Y., Zheng, X. *et al.* Recent advances in plant disease severity assessment using convolutional neural networks. *Sci Rep* **13**, 2336 (2023). https://doi.org/10.1038/s41598-023-29230-7

12. Lu, J.; Tan, L.; Jiang, H. Review on Convolutional Neural Network (CNN) Applied to Plant Leaf Disease Classification. Agriculture 2021, 11, 707. https://doi.org/10.3390/agriculture11080707

13. Fang, X.; Zhen, T.; Li, Z. Lightweight Multiscale CNN Model for Wheat Disease Detection. Appl. Sci. 2023, 13, 5801. (посилання),

14. Citation: Jouini, O.; Aoueileyine, M.O.-E.; Sethom, K.; Yazidi, A. Wheat Leaf Disease Detection: A Lightweight Approach with Shallow CNN Based Feature Refinement. AgriEngineering 2024, 6, 2001–2022. (посилання),

15. Gulzar, Y.; Ünal, Z.; Akta¸s, H.; Mir, M.S. Harnessing the Power of Transfer Learning in Sunflower Disease Detection: A Comparative Study. Agriculture 2023, 13, 1479 (посилання)

16. S. T. Y. Ramadan *et al.*, "Improving Wheat Leaf Disease Classification: Evaluating Augmentation Strategies and CNN-Based Models With Limited Dataset," in *IEEE Access*, vol. 12, pp. 69853-69874, 2024, doi: 10.1109/ACCESS.2024.3397570.

17. De Wolf, E. D., & Shroyer, J. P. (2017). Wheat disease identification (Publication No. MF2994). Kansas State University Agricultural Experiment Station and Cooperative Extension Service.

18. Markell, S., Harveson, R., Block, C., Gulya, T., & Mathew, F. (2023). Sunflower disease diagnostic series (Publication No. PP1727). North Dakota State University Extension Service.

19. Large Wheat Disease Classification Dataset, 2020 (посилання)

20. https://www.kaggle.com/datasets/kushagra3204/wheat-plant-diseases

21. Rajbongshi, Aditya; Sara, Umme ; Akter, Bonna ; Shakil, Rashiduzzaman ; Sazzad, Sadia (2022), “Sun Flower Fruits and Leaves dataset for Sunflower Disease Classification through Machine Learning and Deep Learning”, Mendeley Data, V1, doi: 10.17632/b83hmrzth8.1

22. https://www.ibm.com/think/topics/data-augmentation

23. Sergey Ioffe and Christian Szegedy. 2015. Batch normalization: accelerating deep network training by reducing internal covariate shift. In Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37 (ICML'15). JMLR.org, 448–456.

24. https://pytorch.org/vision/main/models/generated/torchvision.models.efficientnet_b3.html

25. Tan, Mingxing and Quoc V. Le (2019). EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. *ArXiv* abs/1905.11946

26. Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, Liang-Chieh Chen (2019). MobileNetV2: Inverted Residuals and Linear Bottlenecks. arXiv:1801.04381v4

27. Jie Hu, Li Shen, Samuel Albanie, Gang Sun, Enhua Wu (2019). Squeeze-and-Excitation Networks. arXiv:1709.01507v4

28. Huang, G., Sun, Y., Liu, Z., Sedra, D., Weinberger, K.Q. (2016). Deep Networks with Stochastic Depth. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds) Computer Vision – ECCV 2016. ECCV 2016. Lecture Notes in Computer Science(), vol 9908. Springer, Cham. https://doi.org/10.1007/978-3-319-46493-0_39

29. https://keras.io/examples/vision/image_classification_efficientnet_fine_tuning/#:~:text=EfficientNetB3,300

30. https://www.ibm.com/think/topics/fine-tuning