# Застосування згорткових нейронних мереж для розпізнавання образів в обраній предметній області

Виконав: Матійчик Т. Я.

Науковий керівник: Олецький О. В.

# Постановка задачі

- Сучасні проблеми та рішення із застосування згорткових нейронних мереж (ЗНМ) у сільському господарстві
- Ефективність тонкого налаштування, встановлення різних оптимізаторів на повнозв’язаний шар та опорну частину із використанням переднавченої ЗНМ **EfficientNet B3**
- Порівняння з моделлю, побудованою з нуля
- Вплив ауґментації даних на успішність моделей

# Вступ

- Згорткові нейронні мережі застосовуються в розпізнаванні медичних зображень, мікроскопічних знімків, обличь, моніторингу дорожнього руху, задачах із детекції смуг руху тощо
- У агропромисловості поширені проблеми великих затрат часу при детекції людьми, необізнаність працівників у визначенні стану рослини, складні умови середовища, фінансові збитки
- У роботі досліджено застосування ЗНМ в аграрній сфері, зокрема, класифікації хвороб пшениці та соняшнику, як одних із основних світових вирощуваних культур, зокрема в Україні

# Проблематика та сучасні рішення обраної теми

4 публікації:

- *https://doi.org/10.3390/agriculture12081192*
- *https://doi.org/10.3389/fpls.2016.01419*
- *https://doi.org/10.1038/s41598-023-29230-7*
- *https://doi.org/10.3390/agriculture11080707*

Основні проблеми:

- Незбалансованість навчальних вибірок
- Слабка репрезентативність зразків у датасетах
- Проблема багатошаблонної **(multilabel)** класифікації

# Проблематика та сучасні рішення обраної теми

4 публікації:

- *https://doi.org/10.3390/app13095801*
- *https://doi.org/10.3390/agriengineering6030117*
- *https://doi.org/10.3390/agriculture13081479*
- *https://ieeexplore.ieee.org/document/10521470*

Основні рішення:

- Залишкові зв’язки (Residuals) – ResNet
- Inception
- Уважні механізми (SE, CBAM, ECA)
- Методи ауґментації даних (CycleGAN, ADASYN, SMOTE)

# Будова згорткових нейронних мереж

Більшість ЗНМ мають наступні основні складові:

- Шари згортки (convolutional layers)
- Шари об’єднання (pooling layers)
- Шар згладжування (flatten)
- Повнозв’язаний шар (fully-connected)

# Дані для дослідження

- Датасет містить 7162 зображення
- Розподіл на тренувальну, валідаційну та тестову підвибірки зі співвідношенням 8-1-1
- Пакетна нормалізація зі ImageNet-значеннями середнього та стандартного відхилення
- Зміна розмірів зображень до 300x300
- Аугментація тренувальної вибірки з фільтрами RandomRotation(30), RandomHorizontalFlip()

Альтернаріоз

Здоровий соняшник

Іржа соняшника

Здорова пшениця

Ризопусна гниль

Мозаїка

Альтернаріоз

Борошниста роса

Пероноспороз

Борошниста роса соняшника

Альтернаріоз

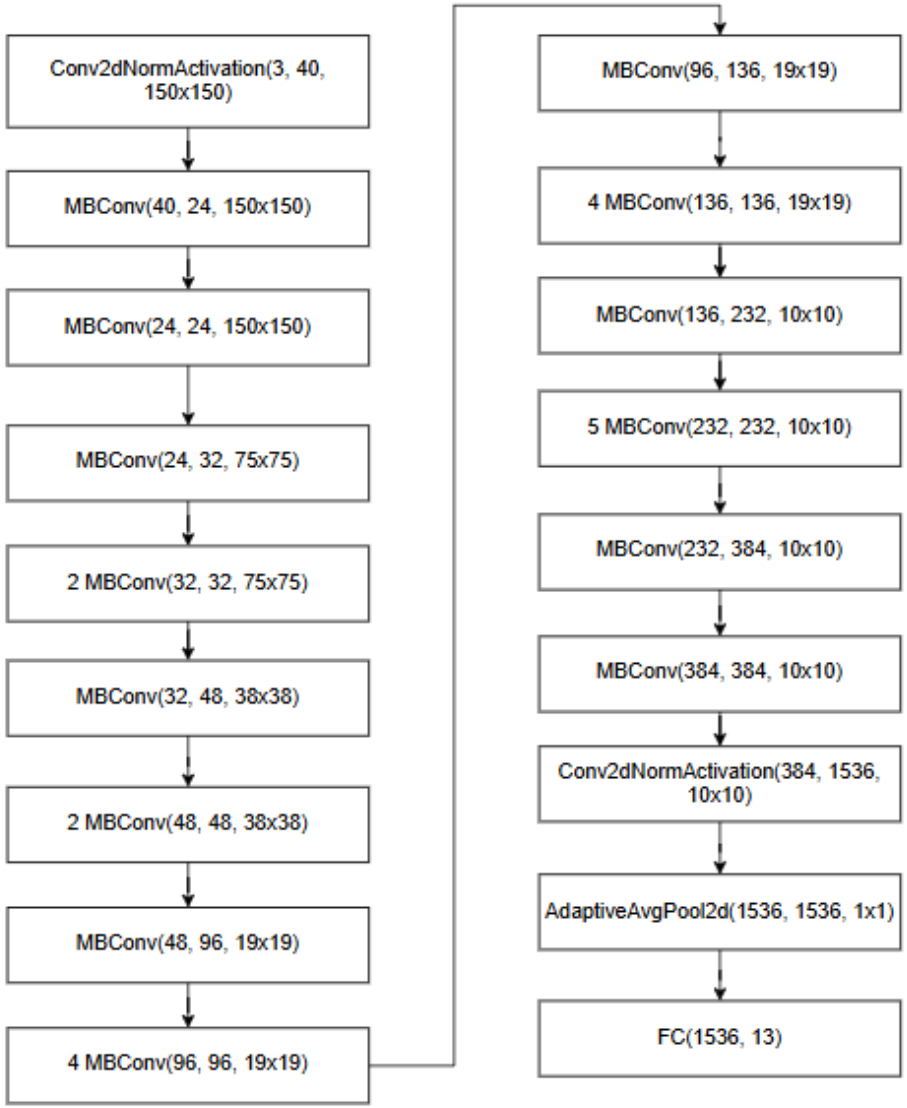
Септоріоз

Іржа пшениці

Склеротиніоз

| 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 | 40 765 668 | 89.91% | 1’574.5 |
| B7 | 8 009.18 | 5.34 | 63 820 | 83.33% | 2840.6 |

Порівняння моделей сімейства EfficientNet, 5 епох, всі шари треновані, learning rate = 0.001, weight decay = 0.0001




Архітектура EfficientNet B3

```python
7 usages  Tymofii Matiichyk
class MBConv(nn.Module):
    Tymofii Matiichyk
    def __init__(self, in_channels, out_channels, expand_ratio, kernel_size, stride):
        super().__init__()
        hidden_dim = in_channels * expand_ratio
        self.use_residual = in_channels == out_channels and stride == 1

        self.conv = nn.Sequential(
            nn.Conv2d(in_channels, hidden_dim, kernel_size: 1, bias=False),
            nn.BatchNorm2d(hidden_dim),
            nn.SiLU(),
            nn.Conv2d(hidden_dim, hidden_dim, kernel_size, stride, kernel_size // 2, groups=hidden_dim, bias=False),
            nn.BatchNorm2d(hidden_dim),
            nn.SiLU(),
            nn.Conv2d(hidden_dim, out_channels, kernel_size: 1, bias=False),
            nn.BatchNorm2d(out_channels)

        )

    Tymofii Matiichyk
    def forward(self, x):
        if self.use_residual:
            return x + self.conv(x)
        else:
            return self.conv(x)
```

```python
2 usages  Tymofii Matiichyk
class ScratchCNN(nn.Module):
    Tymofii Matiichyk
    def __init__(self, num_classes=13):
        super().__init__()
        self.stem = nn.Sequential(
            nn.Conv2d( in_channels: 3,  out_channels: 40,  kernel_size: 3,  stride: 2,  padding: 1, bias=False),
            nn.BatchNorm2d(40),
            nn.SiLU()
        )

        self.blocks = nn.Sequential(
            MBConv( in_channels: 40,  out_channels: 24,  expand_ratio: 1,  kernel_size: 3,  stride: 1),
            MBConv( in_channels: 24,  out_channels: 32,  expand_ratio: 6,  kernel_size: 3,  stride: 2),
            MBConv( in_channels: 32,  out_channels: 48,  expand_ratio: 6,  kernel_size: 5,  stride: 2),
            MBConv( in_channels: 48,  out_channels: 96,  expand_ratio: 6,  kernel_size: 3,  stride: 2),
            MBConv( in_channels: 96,  out_channels: 136,  expand_ratio: 6,  kernel_size: 3,  stride: 1),
            MBConv( in_channels: 136,  out_channels: 232,  expand_ratio: 6,  kernel_size: 5,  stride: 2),
            MBConv( in_channels: 232,  out_channels: 384,  expand_ratio: 6,  kernel_size: 3,  stride: 1)
        )
        self.head = nn.Sequential(
            nn.Conv2d( in_channels: 384,  out_channels: 1536,  kernel_size: 1, bias=False),
            nn.BatchNorm2d(1536),
            nn.SiLU(),
            nn.AdaptiveAvgPool2d(1),
            nn.Flatten(),
            nn.Dropout(0.3),
            nn.Linear( in_features: 1536, num_classes)
        )

    Tymofii Matiichyk
    def forward(self, x):
        x = self.stem(x)
        x = self.blocks(x)
        x = self.head(x)
        return x
```

Архітектура мережі,
побудованої з нуля

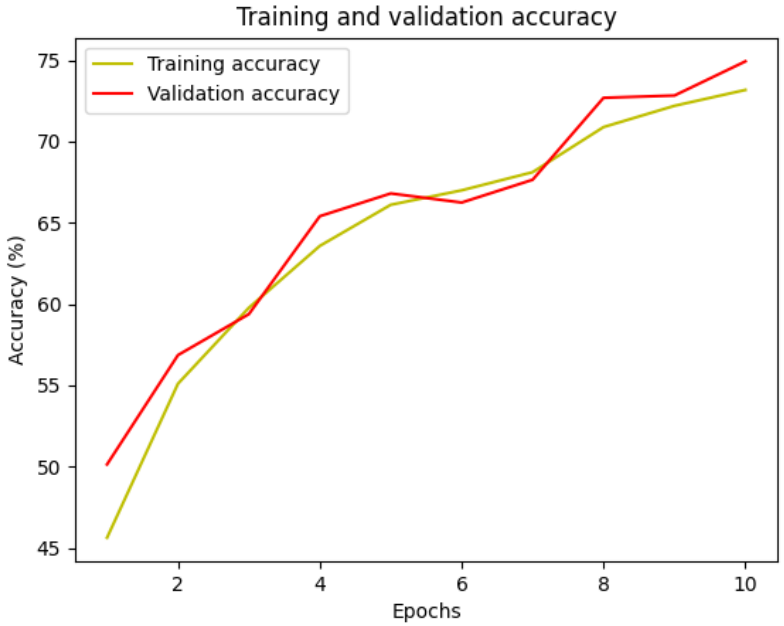
# Тонке налаштування

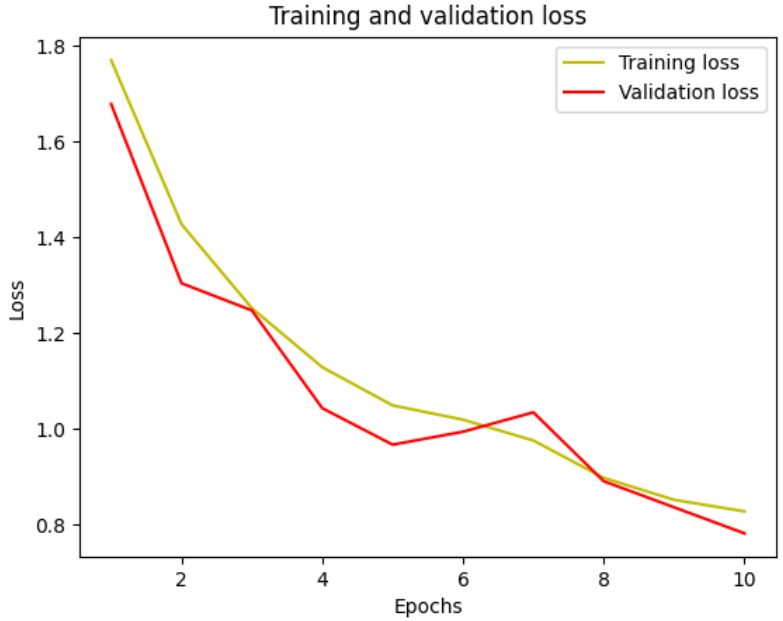
Конфігурації ЗНМ для порівняння:

- Повністю уточнена мережа (всі шари треновані)
- Частково уточнена – тренованими є останні 2 шари **MBConv**, **AdaptiveAvgPool2d** у **backbone**-частині та **FC**.
- Мережа, побудована з нуля, без перенесеного навчання

| 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 |

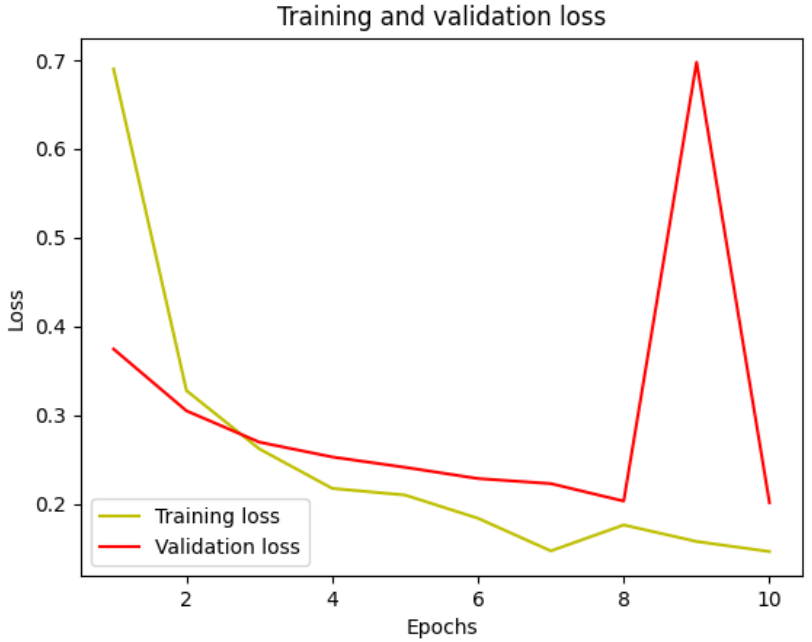
Обсяг, GFLOPs та кількість параметрів для уточнених мереж

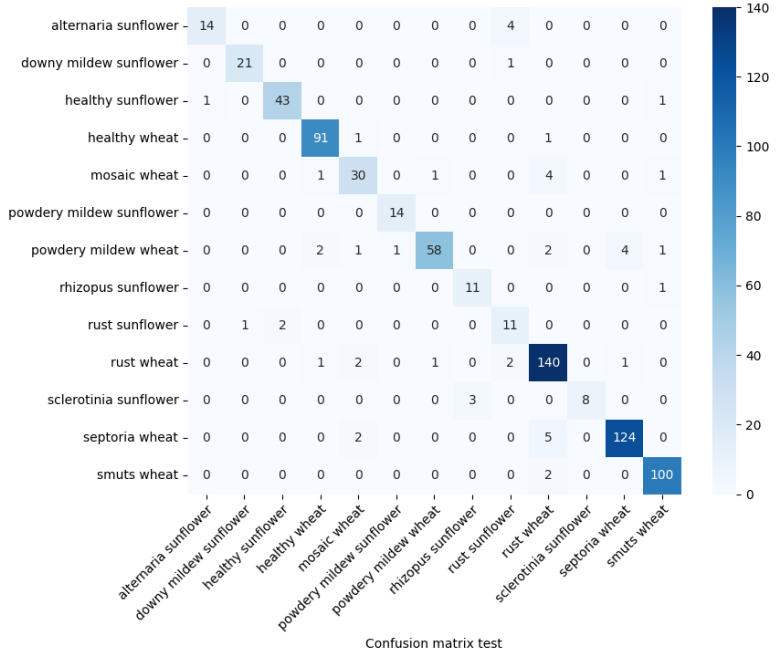


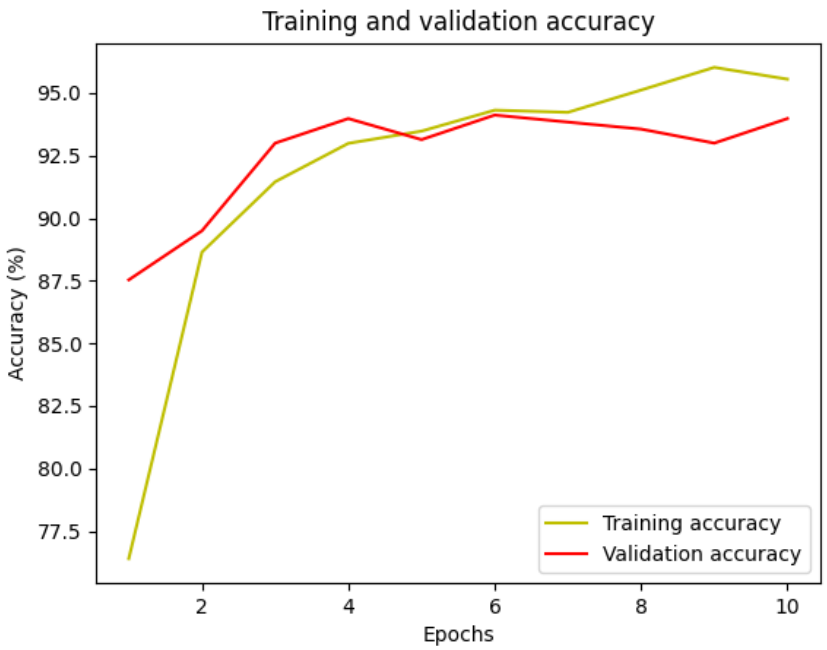


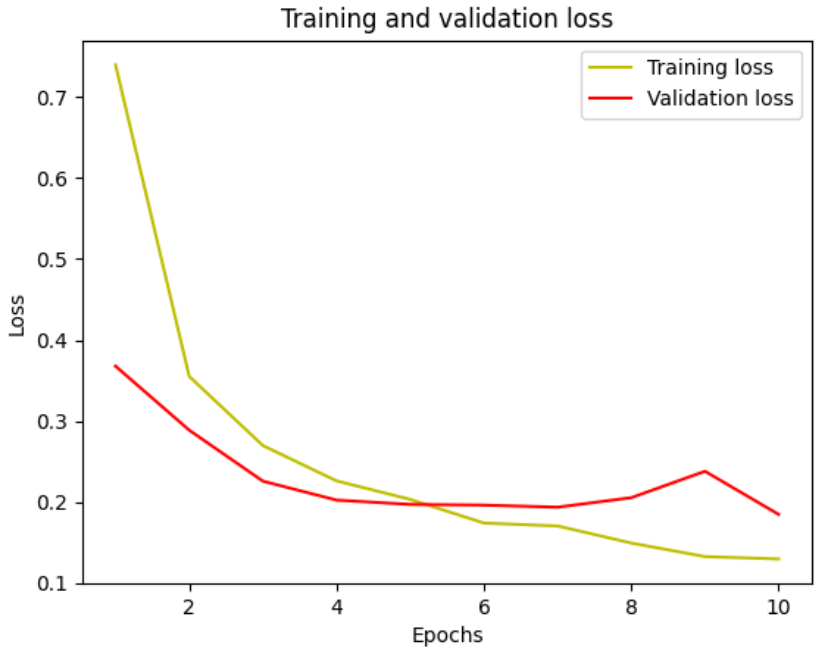
Графіки тренувальної та валідаційної точностей, матриці конфузій для мережі, побудованої з нуля

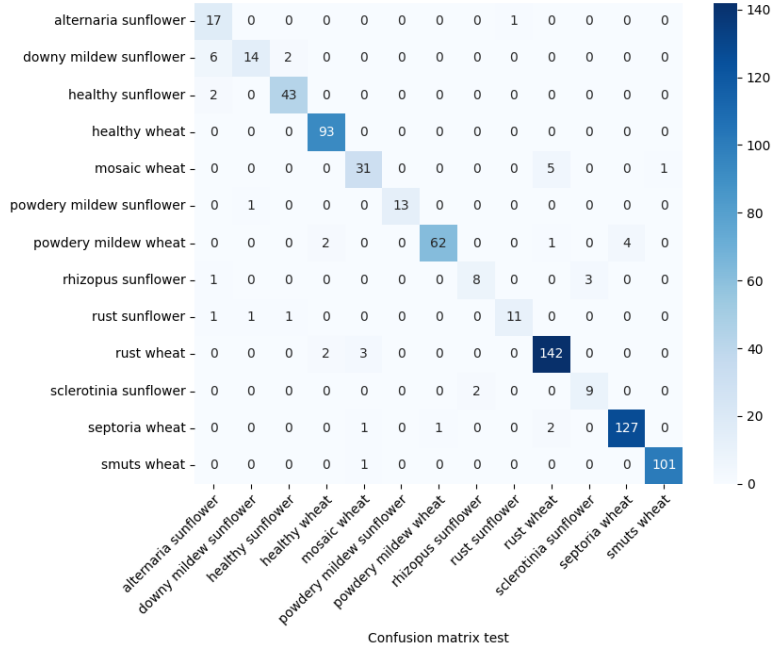




Графіки тренувальної та валідаційної точностей,
матриці конфузій для повністю розмороженої
**EfficientNet B3**




Графіки тренувальної та валідаційної точностей, матриці конфузій для частково розмороженої **EfficientNet B3**

| 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 |

| 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 |

Таблиці успішності уточнених мереж

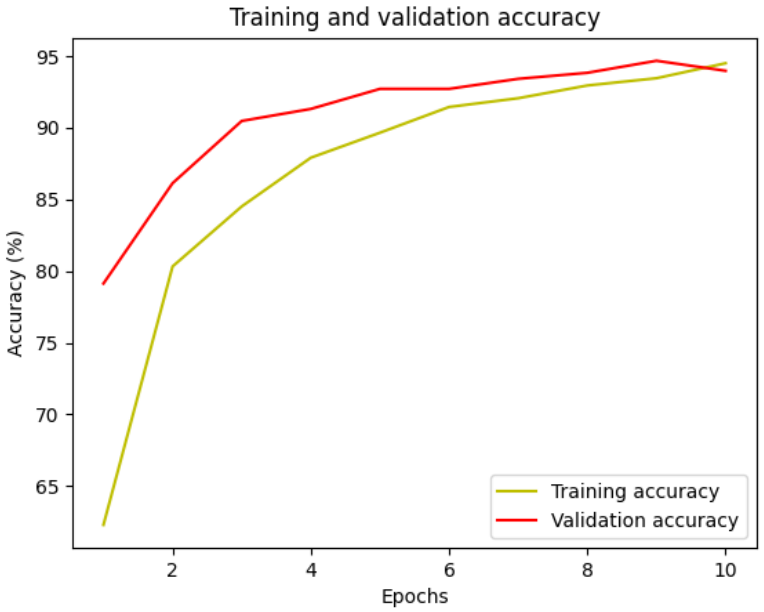
# Різні гіперпараметри для backbone-частини та повнозв’язаного (FC) шару

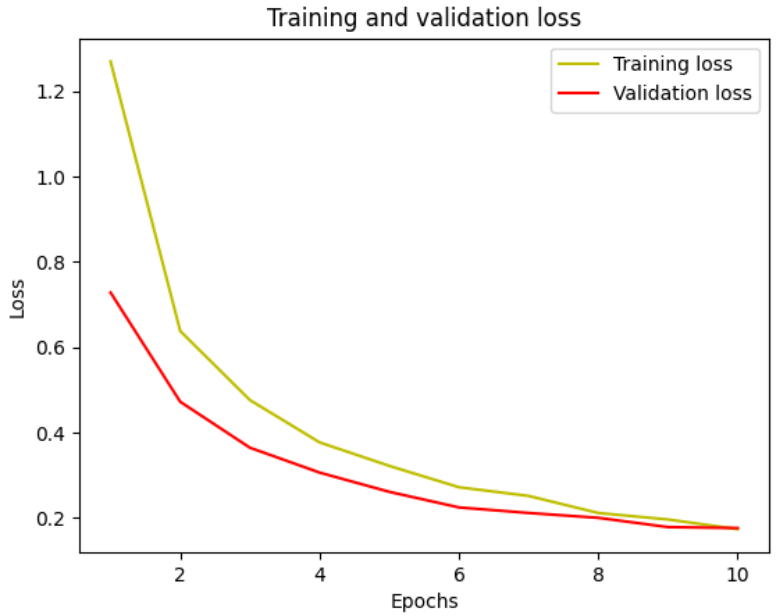
Конфігурації ЗНМ для порівняння:

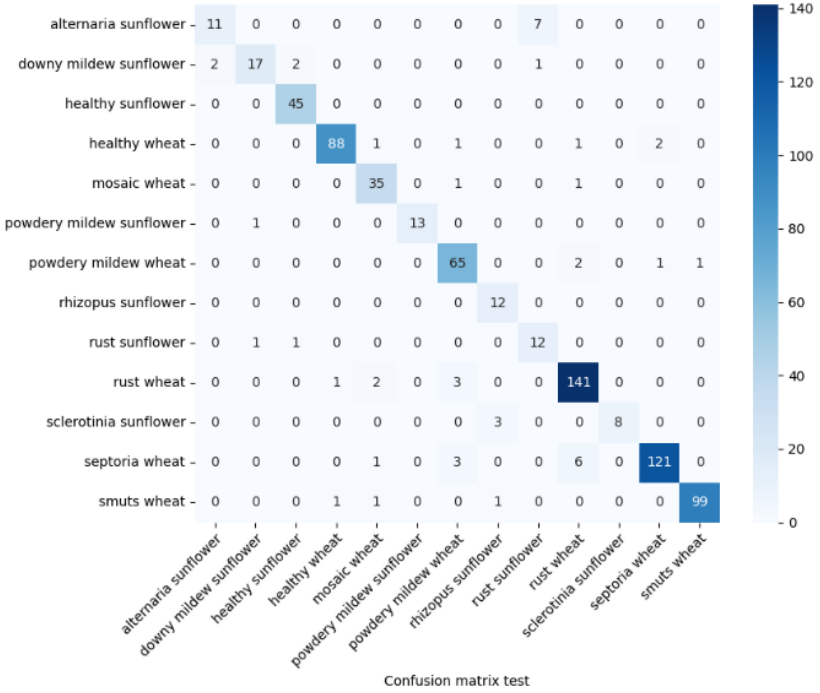
- **Single Adam** – єдиний оптимізатор на всю модель
- **FC Adam + Backbone Adam (1)**
- **FC Adam + Backbone Adam (2)**

**Backbone –** частина будови ЗНМ перед повнозв’язаним шаром

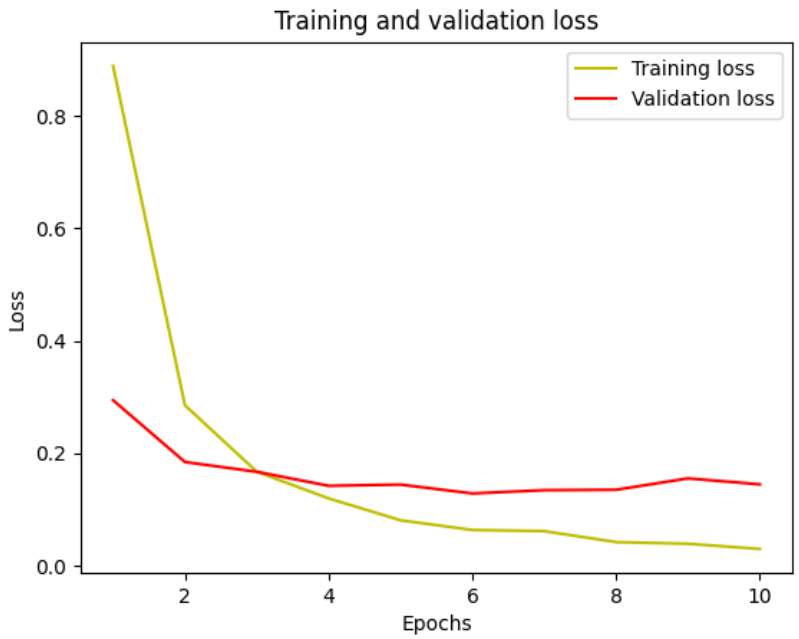
| Тип оптимізатора | Рейтинг навчання | Затримка ваг |
|---|---|---|
| Single Adam | $1e^{-3}$ | $1e^{-4}$ |
| FC Adam | $1e^{-3}$ | $1e^{-4}$ |
| Backbone Adam (1) | $1e^{-5}$ | $1e^{-6}$ |
| Backbone Adam (2) | $1e^{-4}$ | $1e^{-5}$ |

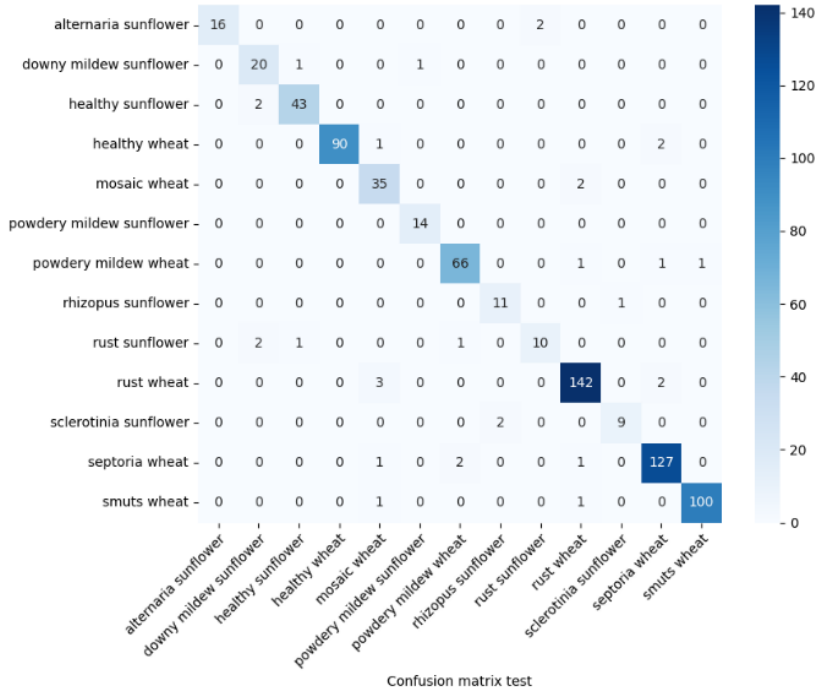



Графіки тренувальної та валідаційної точностей, матриці конфузій для EfficientNet B3 із Backbone Adam (1)


Графіки тренувальної та валідаційної точностей, матриці конфузій для EfficientNet B3 із 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 |

| 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** |

Таблиці успішності мереж з різними оптимізаторами

| | 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 |

Таблиці порівняння точностей мереж під час тренування із та без ауґментації даних

| | З ауґментації | | | 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 |

Таблиці порівняння похибок мереж під час тренування із та без ауґментації даних

| | 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 |

Таблиці порівняння тестових метрик мереж під час тренування із та без ауґментації даних

# Результати

- **Частково уточнена мережа** показала найкращі результати на метриках під час навчання та тестування серед моделей з тонким налаштуванням
- Серед моделей з різними оптимізаторами **найбільш стійкою до перенавчання** виявилась модель із оптимізатором **Backbone Adam (1).** Проте **найкращою з точки зору показників** – модель із оптимізатором **Backbone Adam (2)**
- Без аугментації даних точність моделей на небачених даних падає, більше спостерігається перенавчання. Модель із оптимізатором **Backbone Adam (2)** здобула найвищу точність із та без аугментації

# Подальші покращення та розвиток

- Підвищення точності класифікації (інтеграція кращих уважних механізмів як CBAM, генеративні техніки)
- Удосконалення навчальної вибірки (урізноманітнення та балансування класів у датасетах)
- Створення прикладної програми для застосування в реальних умовах

# Висновки

- Представлено наявні проблеми та рішення із застосування ЗНМ в обраній предметній області
- Досліджено різні методи застосування згорткових нейронних мереж для розпізнавання образів в обраному домені – класифікації хвороб пшениці та соняшнику
- Серед імплементованих методик часткове тонке налаштування та інтеграція різних оптимізаторів виявились найбільш ефективними – такі моделі досягли **94-96%** валідаційної точності та **0.94-0.96** значення F1-балу
- Ауґментація даних показала себе ефективною технікою підвищення якості навчання ЗНМ та розпізнавання образів

# Дякую за увагу!