**Image Neural Networks**

Advanced Analytics – D604

Task 1: Neural Networks

Student Name: Biljana Temovska

Student ID: 002871584

Date 06/18/2025

**A1)** The research question is: Can we create an image classification model that identifies crop seedlings from weed species using images?

**A2)** The goal is to train Convolutional Neural Network (CNN) to classify seedling images into one of the 12 plant categories.

**A3)** One industry-standard model for handling image classification is the Convolutional Neural Networks (CNN). Manev for Analytics Vidhya writes, “Since the 1950s, AI researchers have worked on systems to understand visual data, leading to Computer Vision. In 2012, a breakthrough came with AlexNet, developed by Alex Krizhevsky at the University of Toronto. It achieved 85% accuracy in the ImageNet contest, far surpassing previous models. This success was driven by Convolutional Neural Networks (CNNs), which mimic human vision. CNNs are now vital for tasks like image classification, object detection, and segmentation. “(Manav, 2025)

**A4)** I chose the CNN model as they have been designed to handle visual data, making them highly effective for identifying crop seedlings from weeds. CNN uses layers to learn patterns like edges, shape, color which features are essential to recognize weed species. CNN model has also proven its capabilities in the agricultural industry as well. Mohammad El Sakka for Taylor & Francis Online writes, “Specifically, Convolutional Neural Networks (CNNs), a specialized type of deep learning and computer vision models, demonstrated remarkable proficiency in analyzing crop imagery, whether sourced from satellites, aircraft, or terrestrial cameras.“ (Mohammad El Sakka, 2024)

**B1)**

**A graph of different types of plants

AI-generated content may be incorrect.**

**A collage of different plants

AI-generated content may be incorrect.**

**B2)** GeekForGeeks in an article about data augmentation writes, “Data augmentation is the process of increasing the amount and diversity of data. We do not collect new data, rather we transform the already present data.”(GeekForGeeks, 2023)

Using ImageDataGenerator from tensorflow to help with the Data augmentation process the parameters rotates the image 30 degrees, shifts the image width by 10%, shifts the image height by 10%, zooms the image by up to 20%, flips the image horizontally and adjust the brightness 80% to 120%, all these parameters were done randomly. All of these parameters help the model on unseen data and overall helps reducing overfitting.

**B3)** Normalizing the images - Why? Because the model learns faster when input features are on same or similar scale. The original data contains pixel values and this normalization division by 255.0 takes every pixel value and turns it into floating number between 0 and 1. This is also why it is important to print the min and the max value.

Piyush Kashyap for the Medium writes, “An 8-bit image has pixel values in the range [0, 255]. After normalization:

* Values in [0, 255] can be scaled to [0, 1] or [-1, 1].

Why Normalize Images?

1. Improves Model Convergence  
   Normalizing pixel values ensures data is centered around zero, leading to faster and more efficient training.
2. Prevents Overfitting  
   Scaling features to a similar range ensures that all features contribute equally, preventing dominance by any single feature.
3. Supports Transfer Learning  
   Pre-trained models expect inputs normalized to a specific range. Matching the normalization ensures consistency and better transfer learning performance.” (Kashyap, 2024)

**B4)** Perform train-test split – this proportion of 80/20 split is very common starting point for machine learning models; it provides ample training data and enough data for testing and evaluating.

**B5)** Encode target feature – this is already done in Step B4 before the train-test split and it is often recommended be done first before the split.

* Label Encoder transform the name of the plants into numbers. For example, Common Chickweed becomes 0, Scentless Mayweed becomes 1, Loose Silky-bent becomes 2 and so on
* One-Hot Encoding takes those numbers and turns them into binary format which is suitable format for the CNN model.

**B6)** These have been saved with .npy binary file.

**E1)**

A screenshot of a computer

AI-generated content may be incorrect.

**E2)**

* Number of layers = 10
* The types are: 2 Conv2D it is very common in CNN design, 2 Max\_Pooling2D (avoids overfitting), 3 Dropout (prevents overfitting), 1 Flatten, and 2 Dense
* Number of nodes:
  + 32 filters on the first Conv2D layer, 64 filters on the second Conv2D
  + 256 for the Dense hidden layer and 12 (which is the number of classes) for the Dense output layer
* Number of parameters: First Conv2D = 2,432, second Conv2D 18,496, Dense hidden 16,777,472 and Dense output 3,084. Overall, the model has 16.8 million parameters.
* In the hidden layer, both Conv2D use activation of relu (it turns negative number into zero and positive number is kept which helps the model learn faster), and output layer uses activation of softmax (which turns the outputs into probabilities).
* For building the model I used the Kaggle tutorial which also had 10 layers

**E3)** The loss function is categorical\_crossentropy as it works very well when the model has to choose one class of many classes like the 12 weeds of classes. Optimizer is Adam, it uses default learning rate of 0.001, and the Adam optimizer helps the model learn faster and accurately without needing much tuning. Learning rate of 0.001 is a good starting point which helps the model learn at a steady pace. Lastly, the stopping criteria is 15 (is number of epochs), which gives the model enough rounds to learn well and avoid overtraining.

**E4)**

**A screenshot of a computer

AI-generated content may be incorrect.**

**F1a)**

The stopping criteria (or number of epochs) for this model was 15. This is the number of when to stop the model from training. Training too short might result in the model producing poor accuracy, where training for too long the model might learn of all the noise. The reasonable number for epochs starting is either 15 or 50.

**A screenshot of a computer

AI-generated content may be incorrect.**

**F1b)** Comparing the model’s performance on the validation test the model achived 70% accuracy vs 68% on the test set. The results are quite close, meaning model is generalizing well and is not overfitting to the training data. Most of the classes perform consistently between both sets, except Black-grass has low recall and F1 score which the models does struggle in this class to identify them. The high performing classes are Common Chickweed and Small-flowered Cranesbill as they show strong precision, recall and F1 scores. Overall the model is making good predictions and learning useful features of the different classes.

**Against test set**

**A screenshot of a computer screen

AI-generated content may be incorrect.**

**Against validation set**

**A screenshot of a computer screen

AI-generated content may be incorrect.**

**F1c)**

**A graph of a graph with a line

AI-generated content may be incorrect.**

**F2**) Even though the validation accuracy came at70%, and test accuracy came at 68%, very close — this is good as well as the F1-scores are fairly balanced across classes just looking at the numbers would suggest the model has a good fit not overfitting or underfitting significantly. However, looking at the losses for each epoch clearly shows sign that the model is overfitting. Where the training loss is decreasing suggests the model is learning the training data well, however, on the validation loss it stays higher and improves thus the model is not generalizing well on unseen data.

**F3)** To evaluate the predictive accuracy I used the test set which the model has not seen during training or validation. The model delivered 68% accuracy which is correctly classified plant species against the test images. F1 scores were also examined for each class, with Common Chickweed and Small-flowered Cranesbill showed highest F1 scores suggesting the model does a good job predicting these two classes. In comparison to the Black-grass which showed the lowest F1 score thus the model is struggling to identify this class and needs its performance improved. Overall, the model shows moderate predictive accuracy with room of improvement.

**G1)** model.save('plant\_classifier\_model.h5')

**G2)** The chosen model is Convolutional Neural Network (CNN) which classifies plant species with data that comes from images. Automatically learns about the plant edge, shape, color, etc. (which are the convolutional layers) from 4750 images. It starts by looking at small parts of the images using filters (there are 32 filters on first convolutional layer and 64 filters on the second one) to find things those edges, shapes, and colors. It then shrinks the image with pooling layers to focus on the most important details and make the model faster. In order to stop the model from just memorizing the training images, it uses dropout, which randomly “switches off” some parts of the network during training so it learns more general patterns. Finally, the model flattens the information into a list and uses a dense layer to decide which plant species the image is most likely to be and it gives probabilities for each class.

**G3)** The model addresses the business question by classifying crop seedling from weed with accuracy of 68% on unseen test data. The model performs very well on common species like Common Chickweed and Small-flowered Cranesbill, which are important for accurate field identification. However, some species such as Black-grass are harder for the model to classify correctly, showing lower accuracy and indicating areas where the model can be improved.

**G4)** While this CNN model can classify the weeds reasonably well, some of the classes are harder to identify accurately. Overfitting is definitely a challenge here, as it is evident in the visualization of training loss and validation loss thus the model did very well on the training dataset, but not great on the test unseen data. Perhaps collect better photos especially on those underperforming classes like Black-grass, and/or experiment with different parameters in the model’s architecture.

**G5)** The first next course of action would be to collect more diverse images for the underperforming plans (like Black-grass). Perhaps images that shows different growth stages, and their backgrounds would help the model learn better and become more robust. Where for the plans like Common Chickweed and Small-flowered Cranesbill the images are sufficient and the model is doing very well.

**I)**

GeekForGeeks (03 Aug, 2023). *Python | Data Augmentation*

<https://www.geeksforgeeks.org/machine-learning/python-data-augmentation/>

Kaggle, (Oct 29, 2020). *Convolutional Neural Network (CNN) Tutorial*

<https://www.kaggle.com/code/kanncaa1/convolutional-neural-network-cnn-tutorial>

**J)**

Manav. (2025, May 1). *Convolutional Neural Networks (CNN) – Basics and applications*. Analytics Vidhya. <https://www.analyticsvidhya.com/blog/2021/05/convolutional-neural-networks-cnn/>

(Mohammad El Sakka, 2024). *Images and CNN applications in smart agriculture* <https://www.tandfonline.com/doi/full/10.1080/22797254.2024.2352386#abstract>

Piyush Kashyap. (2024, Dec 2) *Image Normalization in PyTorch: From Tensor Conversion to Scaling*

<https://medium.com/@piyushkashyap045/image-normalization-in-pytorch-from-tensor-conversion-to-scaling-3951b6337bc8>