Assignment #1 – Object Classification Assignment: Diseased vs. Non-Diseased Leaf

Rhichard Koh (100842848)

Faculty of Science, Engineering & Information Technology, Durham College

Advanced Neural Networks: Deep Learning for Spatial Analysis (COSC 41001)

Dr. Noopa Jagadeesh

September 25, 2024

1. Dataset and Preprocessing Steps

The images that make up the dataset for classification contain both diseased and non-diseased leaves. These are then split into directories named "./training_set", "./validation_set", and "./test_set". Several preprocessing steps have been conducted to increase variety and enlarge this dataset. The augmentations for training data involve resizing the images to 256 × 256 pixels, performing random resized center cropping to 224 × 224 pixels, random horizontal flipping, random rotations up to 20 degrees, random affine transformations of shear and scaling, brightness and contrast adjustments, and normalizing with ImageNet statistics (mean: [0.485, 0.456, 0.406]; std: [0.229, 0.224, 0.225]). It resizes the validation and test data to 256×256 pixels, center-crops it to 224×224 pixels, and normalizes it, similar to how it was done during training. These transformations have been performed with the view to preventing overfitting, since this will make the model more robust to changes in orientation, scaling, and color.

2. Model and Techniques Used

The pre-trained model used in this assignment is the ResNet-50, a deep convolutional neural network which has preliminarily been trained on the ImageNet dataset. This means that it generally understands image features very well. Adjustments have been made to this model for the binary classification between diseased and non-diseased leaves. The pre-trained CNN layers were frozen so that their learned weights are preserved. Finally, the last full-connected layer was replaced with a new one that was composed of a full-connected layer, followed by a ReLU activator, and finally a dropout regularization was applied on the layer output with a probability equals to 0.5 in order to avoid overfitting. Finally, a fully connected layer with only one output was added, and a sigmoid activation function was used to transform the output into a probability between 0 and 1. It was a binary classification problem, so the activation and the loss function were chosen to be Sigmoid and Binary Cross Entropy, respectively.

3. Training and Evaluation Results

It is trained on the augmented training dataset using the Adam optimization algorithm, Binary Cross Entropy for the loss function, complemented by a learning rate scheduler, ReduceLROnPlateau, which regulates the learning rate during training to achieve better convergence when it is plateauing. While training whenever there is an epoch that scores a lower validation loss, I save the model weights as a checkpoint as the best model. When the model's validation loss does not go lower 5 times in a row, early stopping is achieved, and I load the best model's weights. The epoch, training loss, training accuracy, validation loss and validation accuracy are all saved in a CSV logging file every

epoch. After training, several metrics are used to test the performance of the model: accuracy, precision, and recall, f1 score, and a confusion matrix is generated showing the classification results for each class. These are indicative of how well the model differentiated diseased versus non-diseased leaves. The best model scored an accuracy of 100%, precision score of 1.00, a recall score of 1.00, and F1 score of 1.00. Thus, this model predicted every image correctly, and once more scored full in the confusion matrix.

4. Challenges Faced and How They Were Addressed

Working with a small, imbalanced dataset, overfitting was one challenge to overcome. To address overfitting, random rotation, flipping, and affine transformation were used for data augmentation to further increase the diversity of the training data. Besides this, the addition of dropout layers during training using 0.5 as the dropout probability makes the model less likely to not memorize the data it was trained on and helps in better generalization on unseen data. The technical challenges were installing the correct version of python, pytorch, matplotlib, and pil that were compatible to work with each other. These were resolved by Googling the errors found on Stack Overflow with trial and error methods. Finally, things came together, and everything began working with one another.

5. Other Applications or Domains for the Classification CNN Model

This model of classification CNN can find applications in several other domains as well. Medical imaging, for instance, would come very close in classifying between tissues that are diseased and non-diseased. For example, tumor detection from radiology images such as CT scans, MRI, or X-rays. For agriculture, the model could be elaborated to detect various plant diseases across crops with the help of farmers through early disease detection. For a manufacturing use case, defects in products on a production line, such as cracks or scratches, are other disorders that may be noticed. For a wildlife use case, CNNs may be used to confirm animal species and signs of disease using camera trap images or other sensor data in monitoring wildlife. These can also be applied to security systems, enabling them to monitor an area by detecting any suspected or unusual activities through anomaly detection in video feeds.