Deep learning homework 1.

1. Introduction

For this first homework, we were asked to classify plants into two categories defined by their health status: healthy and unhealthy. To solve the binary classification problem, we were provided with a dataset containing 5,200 RGB images of size 96x96, along with their associated labels.

2. Data preparation

2.1. Cleaning of the data

Among the 5,200 supplied images, we initially observed the frequent presence of intrusive images that did not align well with plants but rather featured Mr. Trololo and Shrek. To address this issue, we employed the Fastdup tool [1] to identify and remove duplicate images from the dataset, along with any outliers. The cleaned dataset now comprises 4,532 images, with 2,900 labeled as healthy and 1,632 as unhealthy.

2.2. Balancing data

Considering the outcomes of our initial tests, we are of the opinion that the imbalance in label distribution has caused our model to overemphasize the number of healthy plants. As a result, we experimented with various methods to address this issue.

2.2.1. Class weights approach

Initially, we attempted to tackle this issue by adjusting class weights. Using an unbalanced dataset, we calculated weights for each class to mitigate the imbalance. While this method yielded positive results on our training dataset, it proved ineffective on the testing set due to differences in label distribution. Consequently, we opted to explore an alternative method.

2.2.2. Data augmentation

We opted to augment our data to rebalance the proportion of healthy and unhealthy images in the dataset. We applied augmentation techniques, including random flips, random translations, and random rotations. These augmentation methods were selected to maintain the realism of the data. Our primary goal was to prevent the introduction of irrelevant data that could lead the model to form unintended correlations. We were particularly cautious about modifying properties such as luminosity, as it could potentially make unhealthy images appear healthier due to color-related associations with plant health.

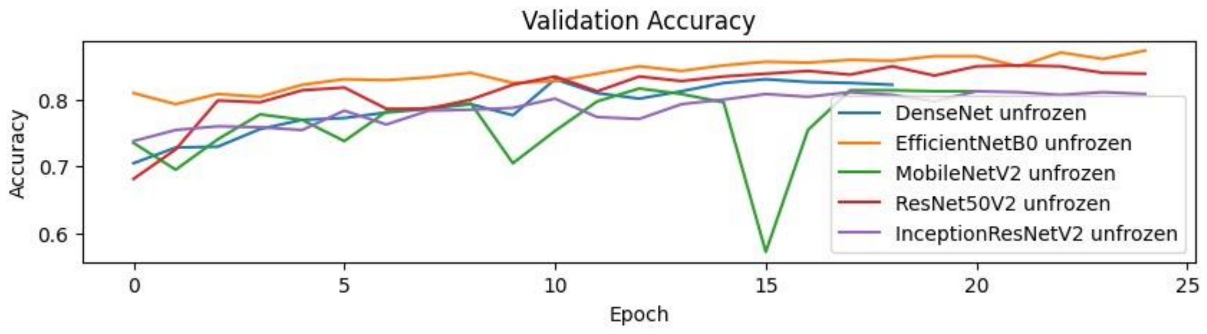
3. Building the model

3.1. *Transfer learning*

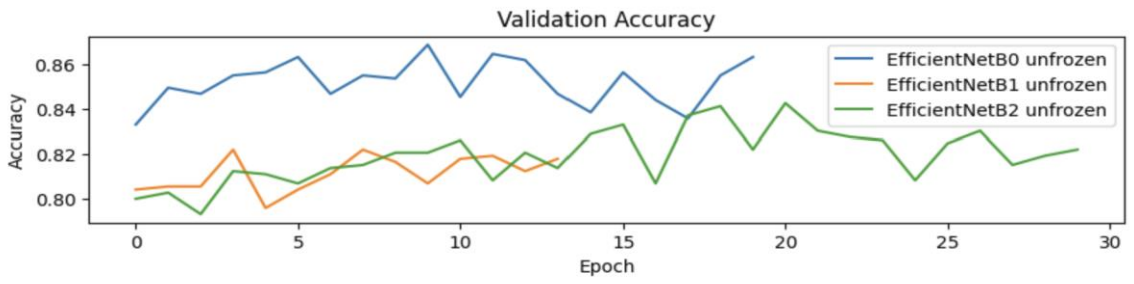
In the initial phase, we experimented with simple CNN models. However, it became evident that models constructed through transfer learning were significantly more effective and quicker to train. Consequently, the decision was made to prioritize this approach.

3.2. *Transfer learning models*

We began by testing multiple transfer learning models from Keras Applications [2] and compared the results to identify the most suitable model for our problem. As shown in the table and chart, the EfficientNet model yielded the best results.



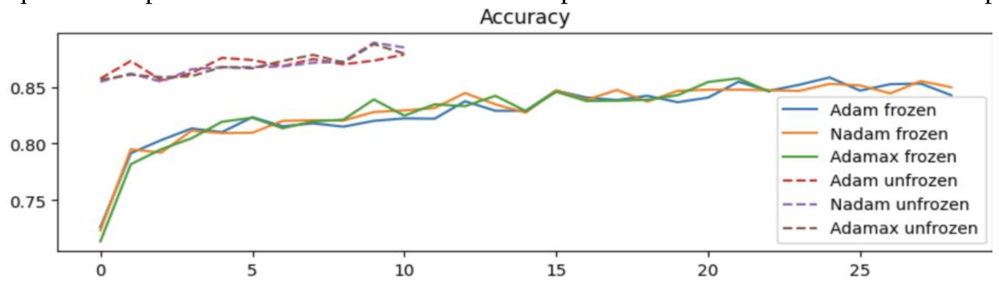
Then we tried different smalls EfficientNetModels and kept the best one wich was EfficientNetB0.



3.3. Parameters

3.3.1. *Optimizers*

Models employing transfer learning undergo two training phases. In the first phase, the layers of the pretrained model are frozen, while in the second phase, all layers are unfrozen. Consequently, it becomes feasible to use two different optimizers for a single model. Initial tests highlighted Adam, Nadam, and Adamax as promising candidates. Subsequent in-depth tests were conducted on these optimizers to determine the two best-performing ones.



The difference between the optimizers is small but the tests seemed to show that the best option is to use Nadam for the two phases.

3.3.2. *Dense layers*

After testing various dense layer configurations, as illustrated in our DenseLayerChoice notebook, the optimal combination was found to be using GlobalAveragePooling, Dropout(0.3), Dense Layer(10, relu), and *Batch Normalization.*

3.3.3. *Unfreezing layers*

The layers of the transfer learning models can be unfrozen and optimized on a different dataset than the one on which it was pre-trained. We experimented with unfreezing 20, 25, 30, 35, and 40 layers, and the results indicated that the highest accuracy was achieved when 30 layers were unfrozen. The details of this testing are presented in the UnFreezeLayerChoice notebook.

3.3.4. *Callbacks*

To enhance model training and mitigate overfitting, we implemented two callback functions: EarlyStopping [3] and ReduceLROnPlateau [4]. Both callbacks monitor the accuracy on the validation set, with EarlyStopping set to a patience value of 5 and ReduceLROnPlateau to 10.

4. Final implemented solution

4.1. Model and parameters

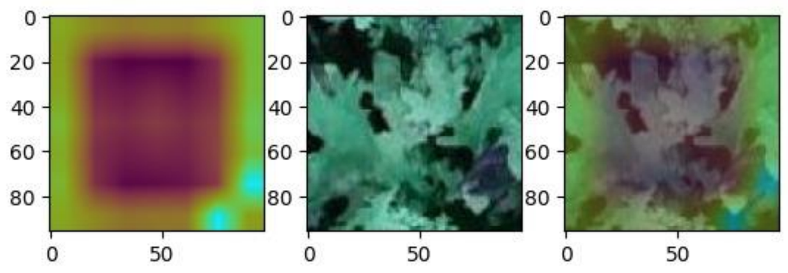
The results from the previous chapter show that the best model should be build as follow :

Optimizers	Transferlearning	Dense layers	Unfrozen depth
Nadam-Nadam	EfficientNetB0	*GlobalAveragePooling, Dropout(0.3), Dense Layer(10, relu)* and *Batch Normalization*	30

However since the outcome of some tests where pretty close and the final test dataset will be different from our dataset, we decided to test some variations that can be found in the final notebook.

4.2. HeatMaps

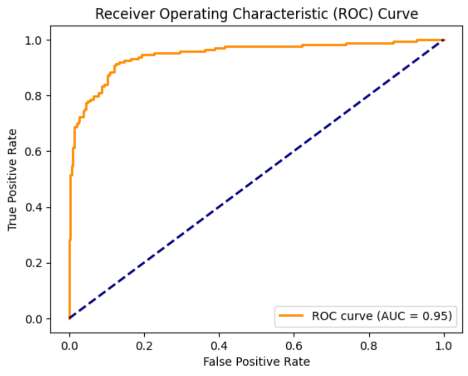
From top convolutional layer we also created heat map from which we were able to conclude that centre regions of an input image contributed the most to the prediction of a particular class.

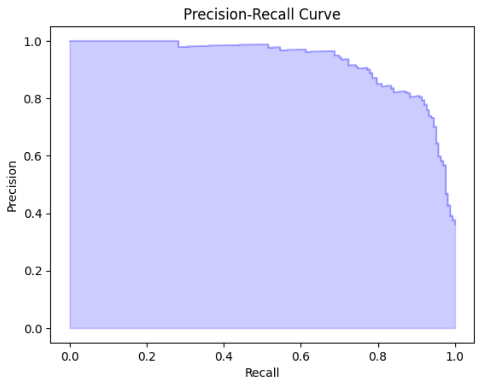


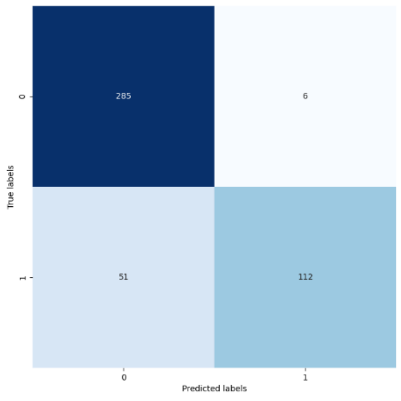
4.3. Results

4.3.1. ROC AUC, confusion matrix, precision-recall curve

Other than accuracy, the model was judged by the ROC AUC curve, in which it has the result of 0.95. This results shows that the model has a great ability to distinguish between different classes in a given dataset. Also, the Precision-recall curve shows the model's effectiveness. The confusion matrix is representing the results of predicting on the test set, out of 454 images, there were 285 true positives (healthy plants), 112 true negatives (unhealthy plants), 51 false positives and 6 false negatives.







4.3.2. *Accuracy, precision, recall, F1 score*

The main results by which the model was chosen as the best are the accuracy, precision, recall and F1 score. Accuracy shows how many images were correctly classified, and that score was 0.8744. Precision is a characteristic which describes how many of the predicted positive instances were actually correct. For this model, precision = 0.9492. Recall measures how many of the actual positive instances were correctly predicted by the model, and this model has a score recall = 0.6871. Lastly, F1 score describes the balance between precision and recall, and in this case the value is F1 = 0.7972.

Accuracy	Precision	Recall	F1
0.8744	0.9492	0.6871	0.7972

5. Conclusion

The obtained model can classify images with an accuracy of 0.8744 (on the test set). In the competition rankings, some groups have achieved significantly higher scores, indicating the potential for further improvement. We believe that the most promising avenues for enhancement include using superresolution models for image resizing, employing grid-fine-tuning for hyperparameters instead of a 'greedy' approach, incorporating K-fold or other cross-validation techniques, and exploring the use of multiple models with a voting system, such as a random forest (ensemble model).

6. Contributions

- Filip Fabris - models testing, dense layers, transfer learning, unfreeze depth, heat map, report, research
- Maxime Pellichero – models testing, optimizer choice, model comparaison,
- Jana Penic – testing models, cleaning dataset, report, heatmap
- Mai Lan Pho - testing models, cleaning dataset, report

7. References

[1] https://visual-layer.github.io/fastdup/
[2] https://keras.io/api/applications/
[3] https://keras.io/api/callbacks/early_stopping/
[4] https://keras.io/api/callbacks/reduce_lr_on_plateau/