Plant Pathology Detection

Houssam El Cheairi

June 2020

Abstract

We implemented convolutional neural networks models to detect and classify pathologies in plants.

1 Problem statement

Misdiagnosis of the many diseases impacting agricultural crops can lead to misuse of chemicals leading to the emergence of resistant pathogen strains, increased input costs, and more outbreaks with significant economic loss and environmental impacts. Current disease diagnosis based on human scouting is time-consuming and expensive, and although computer-vision based models have the promise to increase efficiency, the great variance in symptoms due to age of infected tissues, genetic variations, and light conditions within trees decreases the accuracy of detection.

Our objective here is to implement effcient deep learning models to predict plant's pathologies from a leaf picture.

2 The Data

The training dataset provided by Kaggle consists of 1821 labeled 2048 $\times$ 1365 *png* images of various green leaves. They are either labeled as *healthy* ,*multiple diseases*, *rust*, or *scab*, with only one label having value 1 and the rest 0.

(a) Scab (b) Multiple diseases (c) Healthy (d) Rust

Figure 1: Sample of the data

A quick analysis of the distribution of labels shows that the label *Multiple diseases* is very under represented, which is not optimal for the models. We investigate this issue further in the Annex.

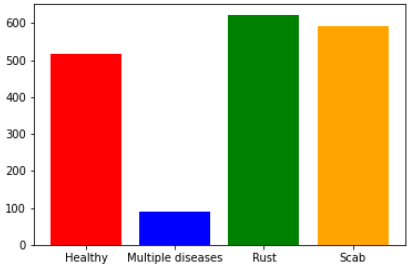


Figure 2: Training dataset structure

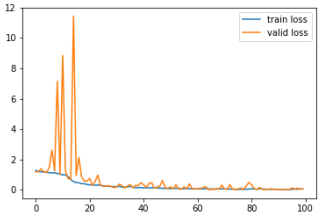
3 Implementation

3.1 First trained models

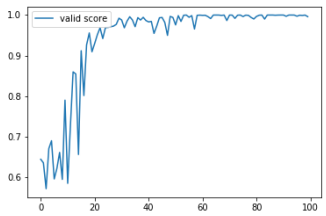
We implemented as part of an exploration strategy three CNN architectures:

- **ResNet18 & ResNet50**
- **Xception**

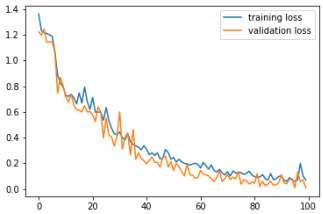
These models were trained on 80% randomly selected images from the training data using a 512×512 final resolution while the 20% left were used for validation. The run time varied from 2h to 6h.
The accuracy metric used to evaluate the models scores is the ROC AUC metric since the Kaggle API uses it to evaluate the submissions. The evolution of the models losses and scores during training is represented in the graphs below:



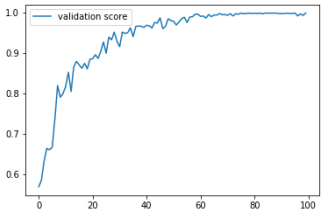
(a) ResNet18 loss



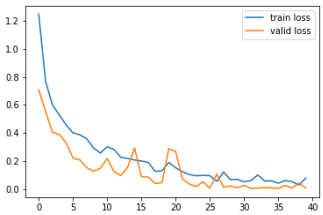
(b) ResNet18 score



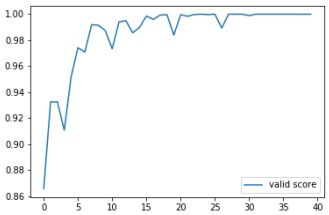
(a) ResNet50 loss



(b) ResNet50 score



(a) Xception loss



(b) Xception score

We then used these models to predict the test data labels, the scores given by the Kaggle API were respectively $90\%, 92.9\%, 94.6\%$ for ResNet18, ResNet50, Xception. We thus decided to drop the ResNet18 model as it had clearly a lower prediction power compared to the other two models.

3.2 Ensemble method

Ensemble modeling is a very common technique used in deep learning to improve the accuracy of prediction. For instance, if we have N models $\Gamma_1, ...\Gamma_N$, we could construct a new weighted model Γ_λ s.t:

$$\Gamma_\lambda = \sum_{i=1}^{N} \lambda_i \Gamma_i$$

under the condition that $\lambda_1 \geq 0, ..., \lambda_N \geq 0$ and $\sum_{i=1}^{N} \lambda_i = 1$

The ensemble method generally produces better models since we intuitively generate a *compromised* prediction that takes account of every model's *opinion*.

That's being said, there is a necessary optimisation routine to be done in order to find the most optimal λ value. In our case, we are working with two models : ResNet50 and Xception, the optimisation can be done using a dichotomic evaluation of the performances of the model:

$$\Gamma_\lambda = \lambda \times ResNet50 + (1 - \lambda) \times Xception, \lambda \in [0, 1]$$

3.3 Evaluation of the ensemble

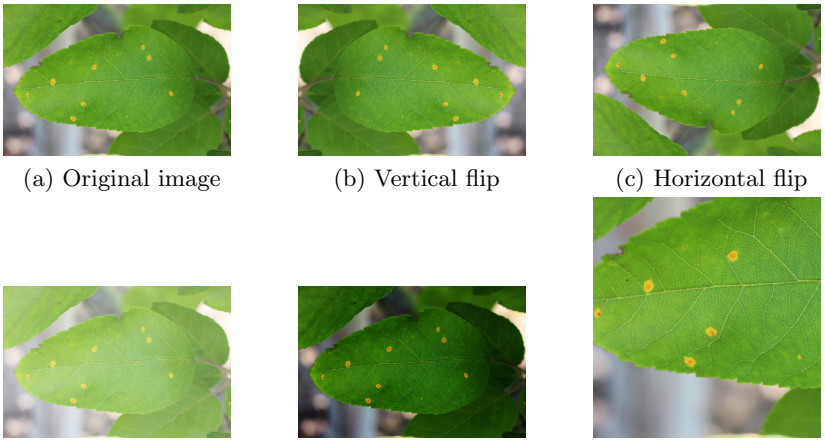
We first tried to evaluate the performance of the ensemble model over randomly shuffled validation dataset from the initial complete training dataset (including both the data used for training and the data used for monitoring/validating the training), however, this approach delivered little information about the real performance of Γ_λ as all the ensemble models had a ROC score equal to 100%.

We thus needed a better dataset that would challenge the trained models predictive ability, which motivated us to use data augmentation.

3.4 Data augmentation

We used 6 different image processing methods to enlarge the training data :

1. Adding noise
2. Adjusting the gamma parameter (darkening and brightening)
3. Horizontal and vertical flips
4. Random cropping



(a) Original image (b) Vertical flip (c) Horizontal flip

(d) Brightening (e) Darkening (f) Random crop

(g) Adding noise

Figure 3: Sample of the data

This gave us access to a 6 times bigger labeled dataset, and made the evaluation of the ensembele model more quantitatively meaningful. We thus

randomly selected a validation dataset from the augmented dataset, and used it to evaluate the Γ_λ model for various values of λ. The results obtained are described by the graph below:

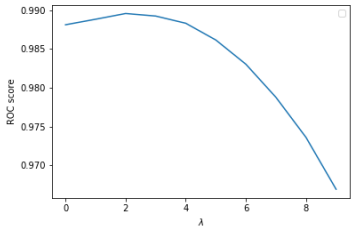


Figure 4: Ensemble evaluation

The obtained curve has the expected theoretical shape, and confirms that ensembling produces better models (indeed, the best ROC score is obtained for $\lambda \notin \{0, 1\}$). The best λ value found (corresponding to the maximal ROC score) is $\lambda = 0.22$.

We then used the model $\Gamma_{0.22}$ to predict the labels of the test dataset. The Kaggle API gave us a 95% score over the submitted predictions file, which is better than our last 94.6% best score.

3.5 Retraining the models on the augmented data

We retrained the past two models (ResNet50, Xception) over 10 epochs on the new augmented dataset as a classical approach to improve the models and evaluated their performance on the Kaggle test API (the run time varied from 6h to 10h), the new models had a score of 95.6% for the Xception model (compared to the past 94.6%) and 94% for the ResNet50 model (compared to the past 92.8%).

We then re-applied the past ensemble strategy by searching the best weighted model out of the two. The graph of the scores is given bellow

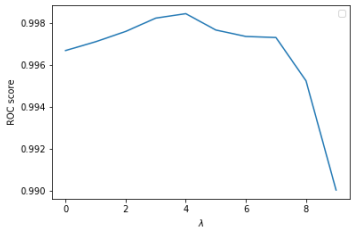


Figure 5: Ensemble evaluation

This analysis suggests that the best combination (using the past notations) is $\Gamma_{0.44}$. Evaluating the latter model on the Kaggle API gave a score

of 96.5% which is better than the past performances.

Unfortunately training the models further on the augmented dataset didn't improve the performances, and thus we decided to explore a different approach to get better results.

3.6 Pseudo labelling

Pseudo labeling (also known as pseudo training) is a training method that uses the test data as training data, more precisely some test data for which our models have very confident predictions can be labelled with those predictions and used as real labels. In our case, our models predict probabilities for each of the 4 classes, we could use the test data with a prediction probability over 99% as training data and monitor the evolution over time of the percentage of confident predictions.

In order for this method to make sense we would expect the number of confident predictions over the test data to be non-decreasing with time, otherwise we might have biased our models with false labels.

The choice of the confident threshold parameter (99%) depends on the distribution of probabilities (for the predicted class) for each test data. The graph bellow represents the distribution of these probabilities for the last generated predictions.

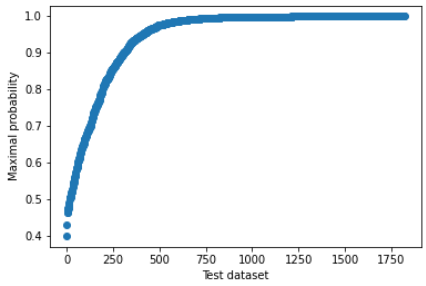


Figure 6: Ensemble evaluation

We trained our previous models (Xception and ResNet50) using pseudo labelling while keeping an eye on the evolution of the confidence of the predictions. The Kaggle API gave a 94.9% score for the new ResNet50 model, and 96% score for the new Xception model. however training these models more using this method didn't give better results

We then applied the ensemble routine as before, and got a new best performance of 96.8% using the model $\Gamma_{0.55}$.

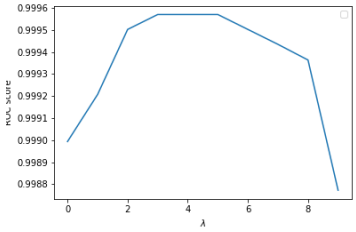


Figure 7: Ensemble evaluation

4 Results summary

The table below summarizes the different scores obtained during the project:

Step	ResNet50	Xception	Optimal ensemble
Direct training	92.9%	94.6%	95%
Training with data augmentation	94%	95.6%	96.5%
Pseudo training	94.9%	96%	96.8%

5 Conclusion and extensions

We implemented recent state-of-the-art (ResNet-2015 and Xception-2016) convolutional neural networks and achieved a 96.8% final score in the Kaggle competition (1.6% (1.3%) away from the 1st (2nd) place submission) using ensemble modeling and pseudo training strategies, however the methods and models implemented can be improved. For instance, we can use the same flow of ideas but with a more robust and complex model choice instead of ResNet and Xception like the recent EfficientNet architecture (2019).

Another possibility would be to use external and more balanced data to tune the models further such as the open sourced PlantVillage dataset.

The approach described in this project can be applied to similar problems that involve detecting anomalies in physical structures such as steel defects, skin sickness, etc.

References

[1] Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun. *Deep Residual Learning for Image Recognition.* (English) . 10 December 2015.

[2] François Chollet. *Xception: Deep Learning with Depthwise Separable Convolutions.* (English) . 7 October 2016.

[3] Muhammad Hammad Saleem , Johan Potgieter, Khalid Mahmood Arif. *Plant Disease Detection and Classification by Deep Learning.* (English) . 29 October 2019.

Annex

This part covers some advancements and adds to the project that can't be fit within the the above work.

Effect of unbalanced data

As mentioned earlier the dataset provided for training is unbalanced, which motivated us to investigate the effect of this unbalance on the final predictions. We thus took our best predictions submission (96.8%) and for each of the 4 prediction classes we computed the mean of the probabilities of those predictions as a way to quantitatively evaluate the confidence of our models on each class. The results are given in the figure below:

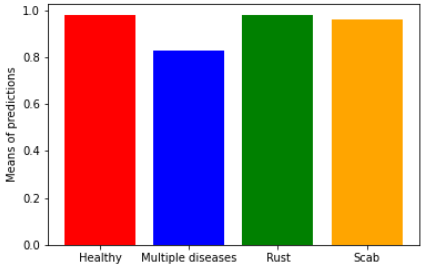
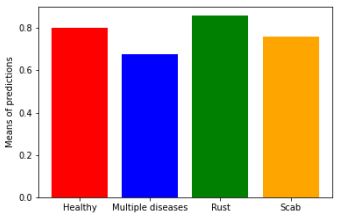


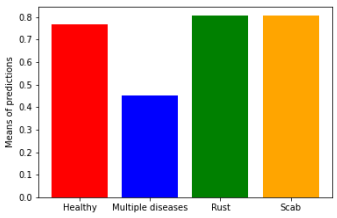
Figure 8: Mean of the predictions confidence per predicted class

As expected the class whose predictions are the least confident is the under sampled multiple diseases class.

We trained two ResNet50 models over 20 epochs on the original dataset and the balanced dataset respectively (using oversampling), the model trained over the balanced dataset gave more confident predictions for the multiple diseases class as shown in the figures below :



(a) Balanced dataset



(b) Unbalanced dataset

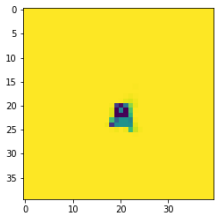
Figure 9: Mean of the predictions confidence per predicted class

Occlusion analysis

Occlusion analysis is a method that helps analyze what parts of the input image contributes the most to the model predictions. The idea is to iteratively hide portions of the image and feed the occulted image to the model to get the probabilities of belonging to each class, one can then create a heat map that associates the occulted parts with a score (based on the predictions). This heat map allows us to observe whether the model detects the pathology features that appear on the leaves.

We implemented this idea algorithmically and tested it using the final Xception model. The figures below show the heat map obtained using a sampled data image from the training dataset infected with rust.

(a) Rust infected leaf



(b) Heatmap obtained with Xception

A mutual training approach

The ensemble strategy proved to be effective in this project as the summary table of results shows.The idea behind ensembling motivated us to try a fundamentally different training approach that embodies a similar spirit as ensembling; assume we have trained two models Γ_1, Γ_2 that have different structures. Each of these models has a certain prediction ability on the data and each is better in detecting certain features, in this sense it could be interesting to mutually train them with each other as in a *debat* format, more precisely we could at each epoch:

1. Train Γ_1 with the training data
2. Train Γ_2 with the training data
3. Train Γ_1 using pseudo labelling based on Γ_2 predictions
4. Train Γ_2 using pseudo labelling based on Γ_1 predictions
5. Update the predictions of Γ_1, Γ_2

We tested this approach with our ResNet50 and Xception models, however this method failed to deliver better results.