EARLY PLANT DISEASE DETECTION BY USING TRANSFER LEARNING

Team: DeepMind

Team Members: Ankita Kadam, Sartaj Bhuvaji, and Siddheshwari Bankar

Problem Statement:

The annual worldwide crop loss is estimated to be USD 220 billion due to plant disease. This results in less food being produced overall and more people suffering from malnutrition. This problem is critical, especially in developing and underdeveloped countries that do not have the resources to save crops from the fast spread of disease. Hence, we propose a system that would be able to identify if the plant leaf is healthy or not. Further, the system would classify the unhealthy leaves into one of three disease classes. This detection would help farmers quickly identify the disease that the crop is affected by and take all necessary actions.

"Detecting and classifying corn (maize) leaves into one of four categories: healthy, northern leaf blight, common rust, and cercospora spot on gray leaf, using Convolution Neural Networks and Transfer Learning."

For the project, we would focus on the 'Corn' data set and balancing the data set, model building, and fine-tuning. We plan to try and create our model and train it over the data. Furthermore, we would also train the models of existing convolutional neural networks like VGG16, Google Net, Mobile Net, etc. We would also like to try different learning parameters and see how they affect accuracy over the board. This would help us fine-tune the model and overall achieve better accuracy.

Problem Type: Classification

Github: Data Science Project

Dataset: https://www.kaggle.com/datasets/abdallahalidev/plantvillage-dataset

- Corn - Cercospora leaf spot Gray leaf spot(513 files)
- Corn - Common rust (1192 files)
- Corn - Northern Leaf Blight(985 files)
- Corn - healthy (1162 files)

Image Statistics:

Dimensions	256 x 256 x 3
Classes	04
Color	RGB
Type	Jpg, png, jpeg
Total Images	3852
Mean	963.0(Mean of images in total class)
Standard Deviations	313.5

Methodology:

- **Image processing using CNN:**
  - Convolutional neural networks (CNN)-based image processing will be utilized to detect plant diseases. This methodology takes images of the leaves and contrasts them with a dataset that includes healthy, northern leaf blight, common rust and cercospora leaf spot images of leaves. A deep learning algorithm, CNN will be used to train the classification model and determine the leaf in one of the four classes.
- **Transfer learning:**
  - It is a machine learning technique that creates a new model using a pre-trained data model. As a result, we will use a few leaf images to train a model, which we will then apply to fresh leaf images in an effort to learn new things.

Risks:

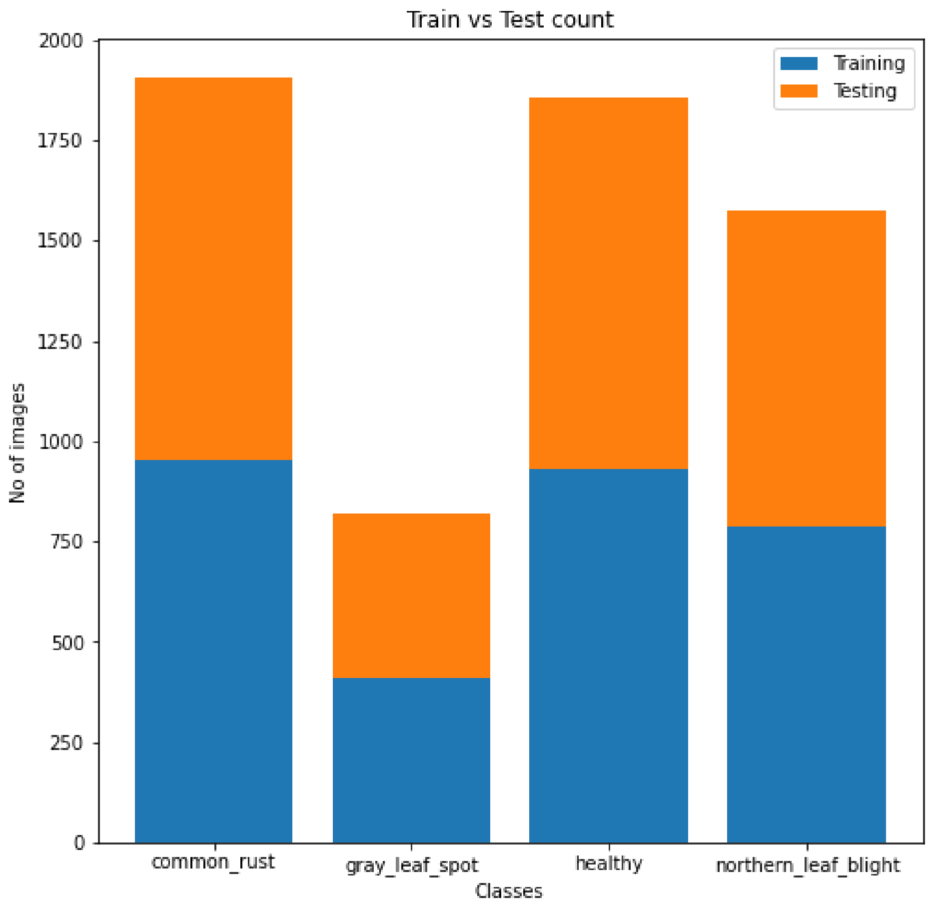
1. If the dataset is imbalanced, then it will be unbiased towards the class with fewer images.
2. While performing transfer learning on the dataset, there are chances of overfitting due to the complexity of the transfer learning models.

Challenges:

For the following problem statement, we are using the corn(maize) dataset to solve the four-class classification problem. The following are the challenges we faced when data preprocessing:

1. **Balancing data:**
    a. The dataset for corn (also known as maize) is imbalanced. The quantity of photos varies between the four classes. In contrast to the other three classes, the "**Cercospora spot gray leaf**" class contains fewer photos (513), as we have seen.
    b. We have performed exploratory data analysis on different leaf classes.

Bar Plot of the corn(maize) dataset



2. **View - Point Variation:**
    a. The images in the dataset are oriented and rotated in different directions depending on the photograph taken.
    b. These are the following images, taken from different angles.


3. **Computational Constraints**
    a. During model training with 5 k folds with 100 epochs google colab crashed

Assumptions:

- The images in the data set are correctly labeled.
- Different view-point of the images do not hinder the accuracy.
- Imbalance in the data set does not greatly affect the accuracy.

Summary on EDA:

- **Github:** Exploratory Data Analysis
- Performing EDA on the complete '**Plant**' dataset(link in Dataset point), we concluded to use the "**Corn**" dataset among the others due to it being a multi-class problem and a relatively balanced dataset. The 'Corn' dataset is a four-class problem.
- After performing EDA on the "**Corn**" dataset, we observe that the class "**gray_leaf_spot**" is an imbalanced class, consisting of **12.20%** of the total images.

Data-Preprocessing:

- **Github:** Data Preprocessing
- We have performed a **train-test split of 0.8:0.2** and we believe that this split captures the jest of data.

Algorithms:

- **Transfer Learning Algorithms** [1]:
  Transfer learning is a machine learning technique where we use a pre-defined and pre-trained neural network and train it again on the current data set. For this project, we plan to use the below models for transfer learning.
    - Mobile Net[5]
        - We have built multiple MobileNet Transfer Learning models with different freeze layers: 0, 25, 50, 75, 100
        - We have built multiple MobileNet Transfer Learning models with different unfrozen ratios : 0, 0.25, 0.50, 0.75, 1
        - We have also trained MobileNet models up till the threshold:
            - Validation accuracy > 0.90
            - Validation Loss < 1.00

- **Under Sampling**[4]
    - Undersampling is the method to remove the datapoints from majority classes to create a more balanced class distribution.
    - For the Corn dataset, we have undersampled the training data for classes: Common rust, Northern Leaf Blight, and healthy to 410 images to class - gray leaf spot

Results:

Best frozen layer output (FL0):

Loss: 0.01376260261
Accuracy: 0.996017313
Validation Loss:1.232747465
Validation Accuracy: 0.8706493735

Average classification report on Training data				
	precision	**recall**	**f1-score**	**support**
0	0.99	0.862	0.912	928
1	0.9	0.978	0.938	954
2	0.78	0.958	0.842	410
3	0.95	0.788	0.83	788
accuracy			0.89	3080
macro avg	0.904	0.896	0.88	3080
weighted avg	0.924	0.89	0.888	3080
Average classification report on Testing data				
	precision	**recall**	**f1-score**	**support**
0	0.984	0.862	0.91	234
1	0.912	0.978	0.942	238
2	0.714	0.866	0.756	103
3	0.914	0.764	0.798	197
accuracy			0.874	772
macro avg	0.882	0.868	0.852	772
weighted avg	0.906	0.874	0.87	772

Best unfrozen ration output (0.5):

Loss: 0.0069346
Accuracy: 0.998355
Validation Loss: 2.1671276
Validation Accuracy: 0.829091

Average classification report on Training data				
	precision	**recall**	**f1-score**	**support**
0	0.996	0.818	0.886	928
1	0.996	0.972	0.984	954
2	0.93	0.47	0.502	410
3	0.696	0.978	0.806	788
accuracy			0.86	3080
macro avg	0.904	0.812	0.792	3080
weighted avg	0.91	0.86	0.844	3080
Average classification report on Testing data				
	precision	**recall**	**f1-score**	**support**
0	0.992	0.772	0.852	234
1	0.998	0.958	0.976	238
2	0.862	0.348	0.366	103
3	0.626	0.94	0.742	197
accuracy			0.816	772
macro avg	0.868	0.756	0.736	772
weighted avg	0.884	0.816	0.796	772

Best Unfrozen ratio output with threshold (0.75):

Loss: 0.01491718059
Accuracy: 0.9959307313
Validation Loss: 0.5534560919
Validation Accuracy: 0.9212987065
Epochs: 56.4

Average classification report on Training data				
	precision	**recall**	**f1-score**	**support**
0	0.99	0.978	0.984	928
1	0.998	0.996	0.998	954
2	0.906	0.864	0.882	410
3	0.922	0.956	0.938	788
accuracy			0.964	3080
macro avg	0.954	0.95	0.95	3080
weighted avg	0.966	0.964	0.964	3080
Average classification report on Testing data				
	precision	**recall**	**f1-score**	**support**
0	0.99	0.978	0.984	234
1	1	1	1	238
2	0.792	0.734	0.756	103
3	0.856	0.9	0.878	197
accuracy			0.932	772
macro avg	0.908	0.902	0.906	772
weighted avg	0.932	0.932	0.932	772

Undersampling results:

Unfrozen ration 0.75:

Loss: 0.00861961707
Accuracy: 0.9969105721
Validation Loss:0.7328595161
Validation Accuracy: 0.908292675

Average classification report on Training data				
	precision	recall	f1-score	support
0	0.984	0.99	0.984	410
1	1	0.998	1	410
2	0.972	0.944	0.958	410
3	0.946	0.962	0.956	410
accuracy		0.98	0.974	1640
macro avg	0.974	0.974	0.974	1640
weighted avg	0.974	0.974	0.974	1640
Average classification report on Testing data				
	precision	recall	f1-score	support
0	0.962	0.988	0.974	234
1	1	0.998	1	238
2	0.764	0.84	0.8	103
3	0.91	0.828	0.866	197
accuracy			0.93	772
macro avg	0.91	0.916	0.908	772
weighted avg	0.934	0.93	0.93	772

Unfrozen ration 1:

Loss: 0.008645766729
Accuracy: 0.996910572
Validation Loss:0.6968380213
Validation Accuracy: 0.9219512343

Average classification report on Training data				
	precision	recall	f1-score	support
0	0.99	0.912	0.946	410
1	0.968	0.99	0.978	410
2	0.91	0.958	0.934	410
3	0.938	0.938	0.942	410
accuracy			0.946	1640
macro avg	0.952	0.946	0.946	1640
weighted avg	0.952	0.946	0.946	1640
Average classification report on Testing data				
	precision	recall	f1-score	support
0	0.992	0.894	0.94	234
1	0.976	0.994	0.984	238
2	0.72	0.866	0.782	103
3	0.894	0.866	0.878	197
accuracy			0.914	772
macro avg	0.896	0.906	0.896	772
weighted avg	0.926	0.914	0.916	772

Miscellaneous:

- Platform: Google Colab
- GPU: Nvidia A100-SXM4-40GB
- Total Training Time: 30hrs
- Loss: 'sparse_categorical_crossentropy'
- Learning rate:0.001
- Batch size: 32

Learnings:

- Applying EDA for better understanding of the dataset.
- Performing k-folds provides more metrics to help evaluate the model accuracy.
- Importance of balanced dataset
- Calculations of unfrozen ratios during model building phase
- Applications of transfer learning models to solve real-world problems

Team Contributions:

Tasks	Team Members
Initial EDA	Sartaj, Siddheshwari
Data Pre-processing	Ankita, Siddheshwari
Data Cleaning	Sartaj, Siddheshwari
Performing Test-Train Split	Ankita, Sartaj
Data Pickling	Ankita, Siddheshwari
Model Building - MobileNet	Sartaj, Siddheshwari, Ankita
Tuning hyperparameters to increase accuracy	Sartaj, Siddheshwari, Ankita
Under - Sampling	Siddhi, Sartaj
Reports	Sartaj, Siddheshwari, Ankita
Presentation PPT	Sartaj, Siddheshwari, Ankita

Conclusions:

- From all the models trained we can conclude that the model with unfrozen ratio 0.5 performs well.
- However, after having a threshold, the model with an unfrozen ratio of 0.75 did outperform the rest models.
- But in this case, the recall value for class 2 is substantially lower than for other classes.
- This issue was resolved using Undersampling, which has high recall scores for all classes.

Citations and Literature:

[1] “Keras API reference.” Keras Applications, https://keras.io/api/applications/

[2] Lau Suki, “Image Augmentation for Deep Learning” | by Suki Lau. “Towards Data Science, 2017, https://towardsdatascience.com/image-augmentation-for-deep-learning-histogram-equalization-a71387f609b2#:~:text=Image%20augmentation%20artificially%20creates%20training,%2C%20shear%20and%20flips%2C%20etc.

[3] Jialin, Sinno & Yang, Qiang. “A Survey on Transfer Learning.” *IEEE*, vol. 22, no. 10, 2010, p. 15,https://ieeexplore.ieee.org/abstract/document/5288526.

[4] Jason Brownlee, “Undersampling Algorithms for Imbalanced Classification” https://machinelearningmastery.com/undersampling-algorithms-for-imbalanced-classification/

[5]“Keras API reference.” Keras Applications, https://keras.io/api/applications/#usage-examples-for-image-classification-models