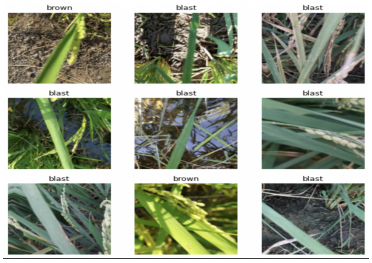
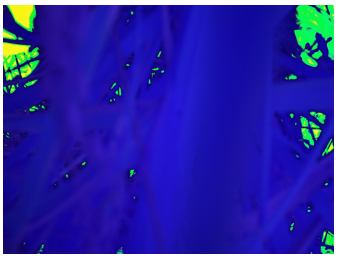
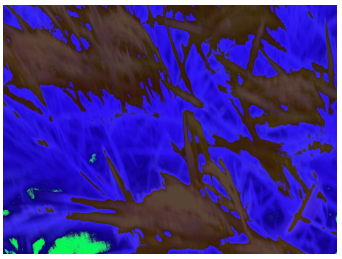
Groupe 7: Microsoft Rice Disease Classification Challenge
Fibi Annan Metiki
Gilda Rech BANSIMBA
Mouhamadou Bamba Diop
Abdou Niang

Report_Microsoft Rice Disease Classification Challenge

Introduction

We implement what is covered in the computer vision class to learn a model that can predict the disease of a rice plant in RGB and Infrared images.Early detection is the best way to manage crop diseases like rice blast, but rice blast is easily misclassified as brown spot disease - both are fungal diseases and have similar appearances in their early stage.We use a dataset with 5340 images to train our model and 1145 images to test .



They registered RGB images to RGNIR images since the two cameras have different field-of-view parameters.
The provided data contains only the registered images.
The covered classes are Brown Spot disease, Blast disease and healthy.

Model Architecture

We decided to use pre-trained models as they have been trained on ImageNet datasets which are sufficiently large and the parameters have already been optimised. Since these output 1000 classes, we changed the output of the fully connected layer for each model from 1000 to 5 to fit our problem ,i.e.
We used a total of 18 models, below we will outline a few of the models that we noticed great improvement in Log loss with :

ResNet50: The first model we tried was the **resnet50** with the following parameters:

1. **lr=0.009**
2. **loss=1.098**
3. **epoch=10**

After we have tried the model with the parameters above , we had decrease the learning rate to have the result below :

1. **lr=0.001**
2. **loss=0.40**
3. **epoch=10**

For this model we had tried to optimise the learning rate to have the better loss like :

1. **lr = 0.0025**
2. **loss = 0.2025**
3. **epoch= 20**

ResNet152 :This was the first model we tried, it saw an improvement in the evaluation metric of Log loss from **0.22**, which was the log loss we obtained from the model **Resnet152**, to a Log loss **0.20** While using this model with decrease le learning rate from **0.009** to **0.001** it gave us the best log loss with equal to **0.20**

We used several other models, after this one and they all were not providing improvement it means that for these models the loss are increase **1.098**

ResNet101: we also tried this models with the parameters and the prediction loss below:

1. **lr=0.009**
2. **loss=0.36**
3. **epoch=10**

After we have not that the loss above is better than the loss of second result for the **Resnet50** we had tried to use this model again with :

1. **lr=0.001**
2. **loss=0.28**
3. **epoch=10**

We next see improvement after switching to **Resnet 101** , loss moved from **0.40** to **0.28**. We had not made any other changes to the architecture of the model.

We continued trying out different models and we were not seeing any improvements from doing so. We then decided to do data manipulation as outlined in the pre-processing section.

Discussion:

From the few models we tried we noticed that the accuracy of the prediction is dependent on the model of choice, some models tend to perform better than others. However it is wise to be cautious as some models tend to overfit, for instance while training using the models like wide ResNet101_2 we got good accuracy when evaluating our model, but this accuracy dropped drastically when we submitted.

We also noted that after some point changing the model is not enough to ensure we get good predictions that our models are highly dependent on the training data. We noticed a huge improvement in accuracy when we increased the sizes of the images, or when we performed data augmentation.

Conclusion

We were able to explore different architectures and implement many of what we learnt in class. We saw that complex model does not equate to more accuracy and that combining simpler models with the right data pre-processing can lead to a more precise model