Final Project Report

Cassava Leaf Disease Classification Deep Learning Approach

AI Programming Course

중앙대학교

Mithona Phou

6 월 2021

1. Introduction

What is the problem?

Cassava is an important crop that provides a basic diet for billion people in Africa. It is a major food source for the local farmers since it can be cultivated under severe conditions. Nowadays Southeast Asia countries are also major cassava crop produce and export. it shares the key for promising total food supply chain. However, a major cause of the loss of cassava is cassava leaf disease. To solve this problem, in this report we aim to develop several deep learning models to classify each cassava image into different categories of Cassava Leaf Diseases so they can be treated accordingly.

Why is it important?

Existing methods of disease detection require farmers to solicit the help of government-funded agricultural experts as well as serious pesticide treatments to visually inspect and diagnose the plants. This suffers from being labor-intensive, low-supply and costly. As an added challenge, effective solutions for farmers must perform well under significant constraints since farmers may only have access to mobile-quality cameras with low-bandwidth. Early diseases diagnose can help infected plants to prevent further spread. Effective solutions for farmers with the help of data science and possibly to identify common diseases so they can be treated.

2. Motivation and Goal

What is the source of this motivation?

Otherwise, In Cambodia we grow, and harvest Cassava crop every year much or less depends on their farmland resources. Cassava has become a regular job to us and my other farmers, helps add extra income after paddy rice season. Without Cassava, Cambodian farmers hard to earn their living expenses. So, they decide to do immigration, sometimes illegal to secure a job in neighboring countries (mostly in Thailand) to feed their poor family. I personally experienced life was hard especially during the harsh dry season.

Further step is to deploy a good AI model that could run and classify the diseases on web and mobile application. Follow this consistency, projected work is to develop and optimize this idea that could benefit for both farmers, government policy, NGO, agricultural specialist, and pest products supply vendors, etc. so they can locally work together toward an effective solution.

3. Source of Data

- InClass Prediction Competition: Cassava Disease Classification by Fine-Grained Visual Categorization 6 (FGVC6 workshop) at CVPR 2019.

https://www.kaggle.com/c/cassava-disease/data

- Research Code Competition: Cassava Leaf Disease Classification by Makerere University AI Lab ended Feb 19, 2021.

https://www.kaggle.com/c/cassava-leaf-disease-classification/data

- TensorFlow Image Classification: https://www.tensorflow.org/datasets/catalog/cassava

4. Data Description

Cassava Leaf Disease Prediction is a computer vision with a dataset of 21397 labeled images of cassava plants. Cassava consists of leaf images for the cassava plant depicting the healthy and four (4) disease conditions; Cassava Mosaic Disease (CMD), Cassava Bacterial Blight (CBB), Cassava Green Mite (CGM), and Cassava Brown Streak Disease (CBSD). The 21397 labelled images are split into a training set (19257) and a validation set (2140). The number of images per class are unbalanced [figure1] with the two disease classes CMD and CBSD having 72% of the images. The model expects 224x224 images with RGB channel values [0,1]. We found that dataset was mislabeled and contained a significant amount of noise.

5. Methods and Implementation

- Exploratory Data Analysis
- Image Augmentations
  - Tensorflow Preprocessing Layers provides basic augmentations such as cropping, flipping and rotation. In this project we tried with preprocessing methods such as: RandomCrop, RandomFlip, RandomRotation, and RandomZoom.
  - Albumentations, an external image augmentation library with much more functionality (through both ImageDataGenerator and ImageDataAugmentor). This library has augmented methods such as RandomCrop, CoarseDropout, Cutout, Flip, ShiftScaleRotate, HueSaturationValue, RandomBrightnessContrast, CenterCrop, and ToFloat.
  - Test Time Augmentation is a technique to increase the accuracy of the predictions.
- AI Modeling
  - Start with CNN TensorFlow & Keras Baseline Model
    - CNN
  - Improve model by using Transfer Learning from Pre-trained Models.
    - InceptionV3
    - Xception
    - VGG16
    - EfficientNetB3

      a good trade-off between accuracy and the number of parameters. It requires the least amount of computing power for inference.

- Validation
  - Stratified K-Fold Cross Validation ~ a method for splitting data into a training and validation set. We only do a stratified 2-fold cross validation due to time constraints.
- Ensemble and Prediction
  - Try Ensemble Models ~ multi-model predictions: combine the predictions from multiple models to improve the overall classification accuracy.
  - We can run code in the same or separated notebook. When training being that we needed the internet enabled to install ImageDataAugmentor, but for inferencing we can do it offline (without internet access). To do that we loaded the models, and then apply the image file to see the result.

6. Conclusion and Results

Model Name	Validation Accuracy
CNN – 4-conv layers	73%
VGG16	77%
InceptionV3	87%
Xception	87%
EfficientNetB3	88%
CropNet_MobileNetV3	73%
EfficientNetB7	too big model for local machine training
Resnext50	(torch_model): for web app inference

- This project we trained several deep neural networks including a deep CNN model, transferred learning with VGG16, InceptionV3, Xception, and do crop images prediction using CropNet_MobileNetV3 model.

- To deploy the project model on web app, we should conder using PyTorch customed ResNext50 (torch_model) for web app inferencing. This model has performed good accuracy and small weights size (roughly 100MB).

- This project has taught me how to search for state-of-the-art deep learning models and how to do custom training the last layers for Cassava Images Classification Dataset on notebooks. This helped me tried various model architectures and fine-tuning models.

7. Github Repository

https://github.com/pjunecau/aiprogramming.git

8. References

- End-to-End Cassava Disease Classification in Keras: kaggle notebook

- Helping African Farmers Increase Their Yields Using Deep Learning: towardsdatascience

- Cassava 2020 (Case Study) [Full Progress][Keras+GPU]: kaggle notebook

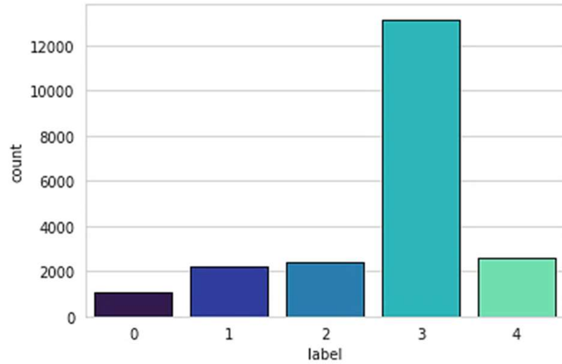
- My Experience Deploying an App With Streamlit Sharing: medium github

- Getting Started: TPUs + Cassava Leaf Disease: kaggle notebook

- Cassava Leaf Disease Detection (2020) Writeup: medium

- Cassava Leaf Disease Detection: github

9. Appendix



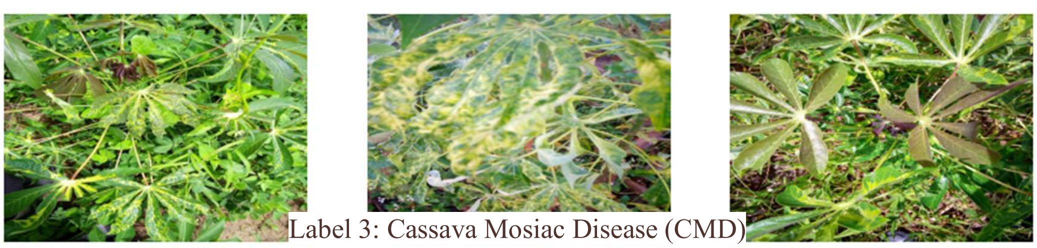
Cassava Mosaic Disease (CMD)	13158
Healthy	2577
Cassava Green Mottle (CGM)	2386
Cassava Brown Streak Disease (CBSD)	2189
Cassava Bacterial Blight (CBB)	1087
Name: label, dtype: int64	

Figure1: Count plot data distribution

Label 0: Cassava Bacterial Blight

Label 1: Cassava Brown Streak Disease

Label 2: Cassava Green Mottle (CGM)



Label 3: Cassava Mosiac Disease (CMD)

Label 4: Healthy

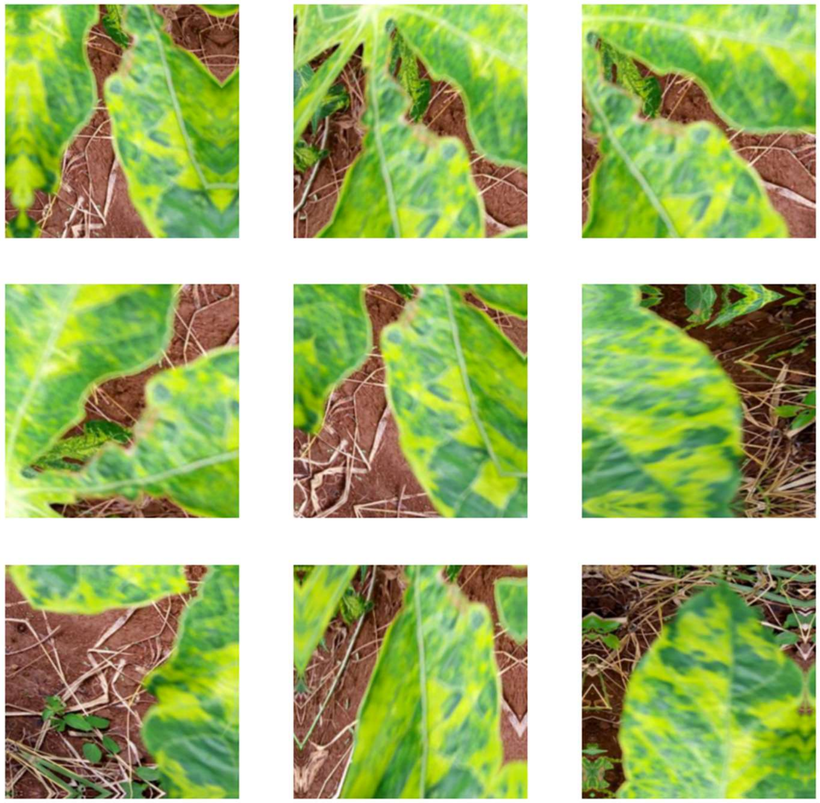


Figure2: Image Augmentation (Tensorflow)

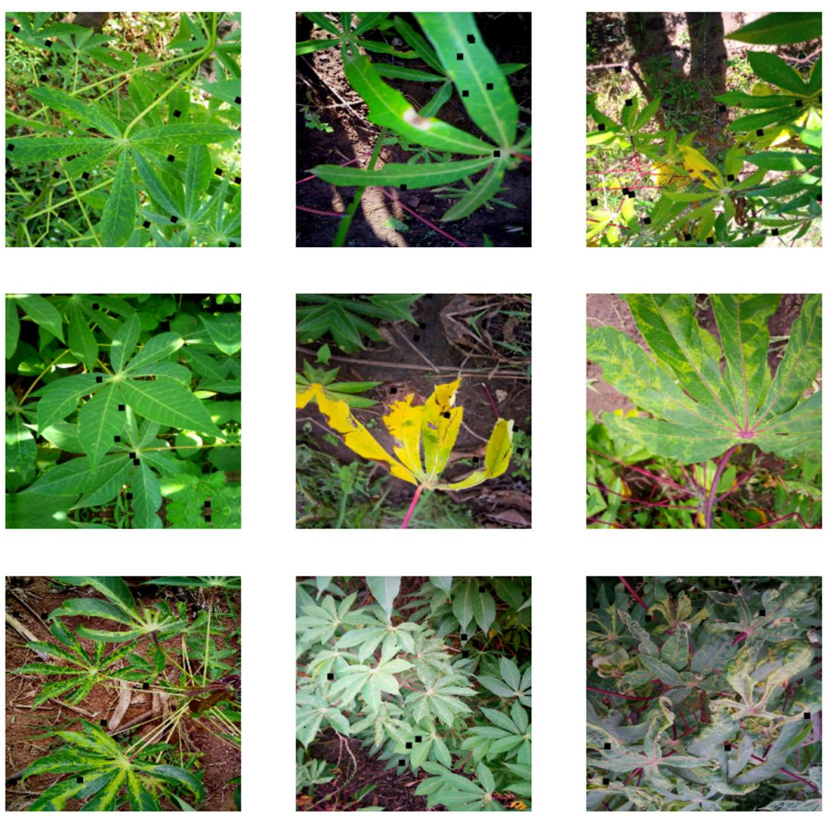
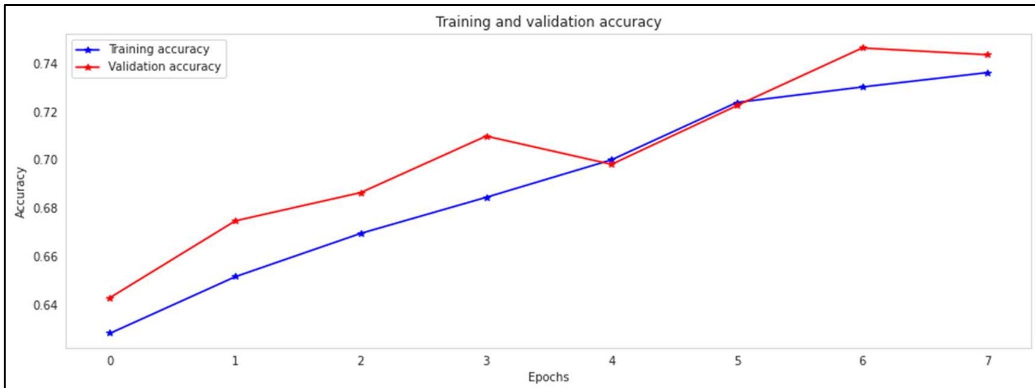
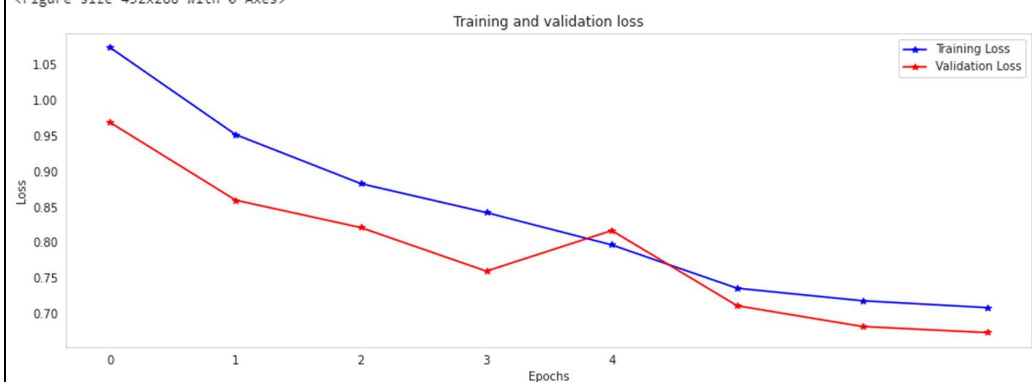


Figure3: Image Augmentation (Albumentations)



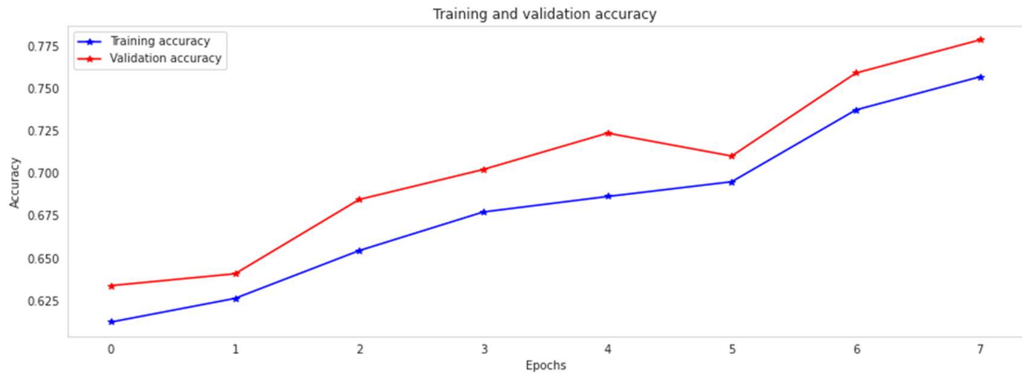

```
<Figure size 432x288 with 0 Axes>
```



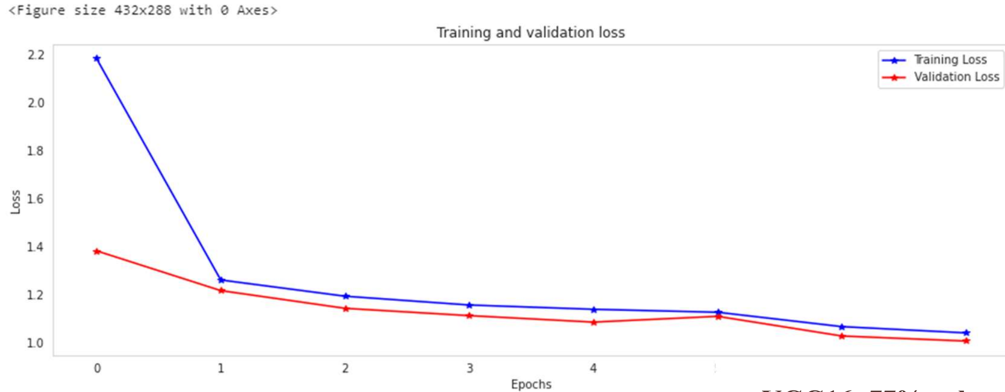
CNN: 74% val_acc

```
<Figure size 432x288 with 0 Axes>
```

```
Found 19257 validated image filenames belonging to 5 classes.
Found 2140 validated image filenames belonging to 5 classes.
Training fold no.: 1
Epoch 1/8
1204/1204 [==============================] - 434s 355ms/step - loss: 1.1206 - accuracy: 0.6202 - val_loss: 0.9379 - val_accuracy: 0.6575
Epoch 2/8
1204/1204 [==============================] - 269s 224ms/step - loss: 0.9723 - accuracy: 0.6489 - val_loss: 0.9047 - val_accuracy: 0.6650
Epoch 3/8
1204/1204 [==============================] - 269s 223ms/step - loss: 0.8987 - accuracy: 0.6683 - val_loss: 0.8062 - val_accuracy: 0.6818
Epoch 4/8
1204/1204 [==============================] - 271s 225ms/step - loss: 0.8320 - accuracy: 0.6871 - val_loss: 0.7814 - val_accuracy: 0.7098
Epoch 5/8
1204/1204 [==============================] - 269s 223ms/step - loss: 0.8199 - accuracy: 0.6952 - val_loss: 0.7447 - val_accuracy: 0.7126
Epoch 6/8
1204/1204 [==============================] - 267s 221ms/step - loss: 0.7856 - accuracy: 0.7041 - val_loss: 0.7209 - val_accuracy: 0.7112
Epoch 7/8
1204/1204 [==============================] - 267s 222ms/step - loss: 0.7420 - accuracy: 0.7160 - val_loss: 0.7011 - val_accuracy: 0.7383
Epoch 8/8
1204/1204 [==============================] - 267s 222ms/step - loss: 0.7419 - accuracy: 0.7150 - val_loss: 0.6614 - val_accuracy: 0.7458
Found 19257 validated image filenames belonging to 5 classes.
Found 2140 validated image filenames belonging to 5 classes.
Training fold no.: 2
Epoch 1/8
1204/1204 [==============================] - 270s 223ms/step - loss: 1.1230 - accuracy: 0.6225 - val_loss: 0.9684 - val_accuracy: 0.6430
Epoch 2/8
1204/1204 [==============================] - 268s 222ms/step - loss: 0.9705 - accuracy: 0.6454 - val_loss: 0.8593 - val_accuracy: 0.6748
Epoch 3/8
1204/1204 [==============================] - 267s 222ms/step - loss: 0.8998 - accuracy: 0.6630 - val_loss: 0.8207 - val_accuracy: 0.6864
Epoch 4/8
1204/1204 [==============================] - 262s 218ms/step - loss: 0.8501 - accuracy: 0.6813 - val_loss: 0.7595 - val_accuracy: 0.7098
Epoch 5/8
1204/1204 [==============================] - 264s 220ms/step - loss: 0.8053 - accuracy: 0.6949 - val_loss: 0.8166 - val_accuracy: 0.6981

Epoch 00005: ReduceLROnPlateau reducing learning rate to 1.9999999494757503e-05.
Epoch 6/8
1204/1204 [==============================] - 268s 222ms/step - loss: 0.7534 - accuracy: 0.7163 - val_loss: 0.7109 - val_accuracy: 0.7224
Epoch 7/8
1204/1204 [==============================] - 263s 218ms/step - loss: 0.7231 - accuracy: 0.7247 - val_loss: 0.6816 - val_accuracy: 0.7463
Epoch 8/8
1204/1204 [==============================] - 265s 220ms/step - loss: 0.7057 - accuracy: 0.7388 - val_loss: 0.6731 - val_accuracy: 0.7435
Training finished!
```



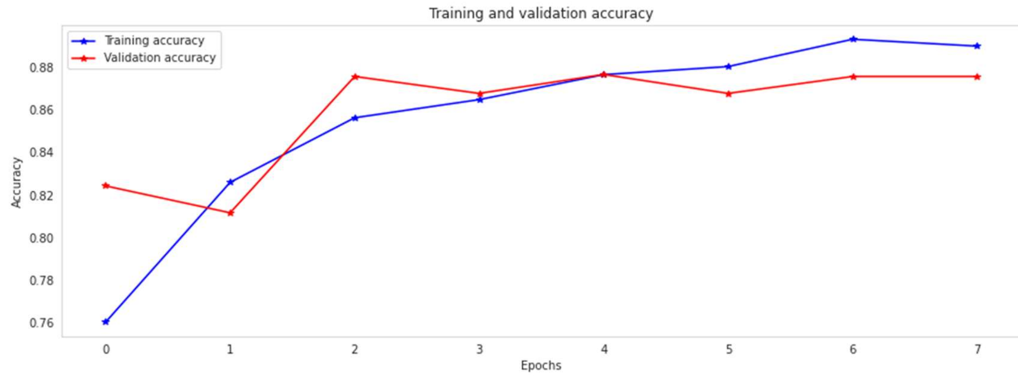

```
<Figure size 432x288 with 0 Axes>
```



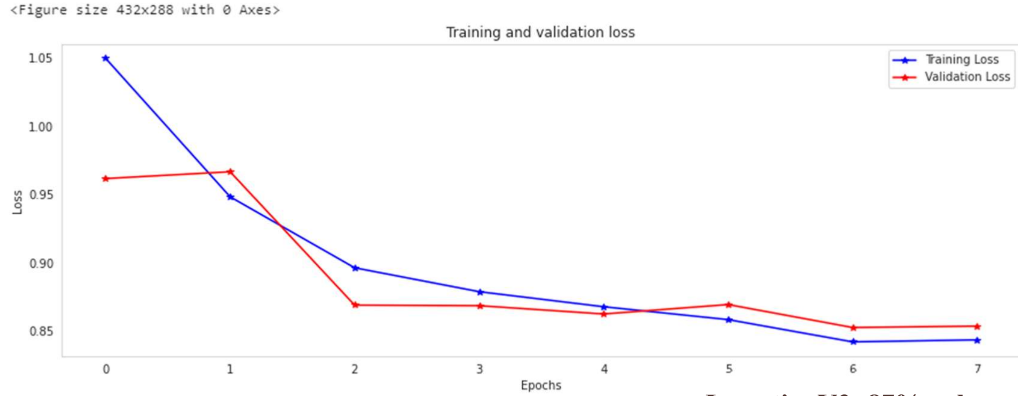

```
<Figure size 432x288 with 0 Axes>
```

VGG16: 77% val_acc

```
Found 19257 validated image filenames belonging to 5 classes.
Found 2140 validated image filenames belonging to 5 classes.
Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/vgg16/vgg16_weights_tf_dim_ordering_tf_kernels_notop.h5
58892288/58889256 [==============================] - 1s 0us/step
Training fold no.: 1
Epoch 1/8
1204/1204 [==============================] - 318s 261ms/step - loss: 15.6980 - accuracy: 0.6030 - val_loss: 1.3333 - val_accuracy: 0.6150
Epoch 2/8
1204/1204 [==============================] - 313s 260ms/step - loss: 1.3457 - accuracy: 0.6138 - val_loss: 1.3409 - val_accuracy: 0.6150

Epoch 00002: ReduceLROnPlateau reducing learning rate to 0.00020000000949949026.
Epoch 3/8
1204/1204 [==============================] - 311s 259ms/step - loss: 1.3351 - accuracy: 0.6167 - val_loss: 1.3324 - val_accuracy: 0.6150
Epoch 4/8
1204/1204 [==============================] - 310s 257ms/step - loss: 1.3333 - accuracy: 0.6137 - val_loss: 1.2577 - val_accuracy: 0.6299
Epoch 5/8
1204/1204 [==============================] - 310s 257ms/step - loss: 1.2492 - accuracy: 0.6325 - val_loss: 1.1905 - val_accuracy: 0.6565
Epoch 6/8
1204/1204 [==============================] - 313s 260ms/step - loss: 1.1913 - accuracy: 0.6537 - val_loss: 1.1060 - val_accuracy: 0.6944
Epoch 7/8
1204/1204 [==============================] - 310s 258ms/step - loss: 1.1344 - accuracy: 0.6931 - val_loss: 1.0779 - val_accuracy: 0.7182
Epoch 8/8
1204/1204 [==============================] - 310s 258ms/step - loss: 1.0927 - accuracy: 0.7187 - val_loss: 1.0418 - val_accuracy: 0.7421
Found 19257 validated image filenames belonging to 5 classes.
Found 2140 validated image filenames belonging to 5 classes.
Training fold no.: 2
Epoch 1/8
1204/1204 [==============================] - 311s 257ms/step - loss: 6.2362 - accuracy: 0.6013 - val_loss: 1.3820 - val_accuracy: 0.6336
Epoch 2/8
1204/1204 [==============================] - 310s 257ms/step - loss: 1.2840 - accuracy: 0.6217 - val_loss: 1.2168 - val_accuracy: 0.6407
Epoch 3/8
1204/1204 [==============================] - 311s 258ms/step - loss: 1.2059 - accuracy: 0.6493 - val_loss: 1.1432 - val_accuracy: 0.6846
Epoch 4/8
1204/1204 [==============================] - 312s 259ms/step - loss: 1.1598 - accuracy: 0.6751 - val_loss: 1.1128 - val_accuracy: 0.7023
Epoch 5/8
1204/1204 [==============================] - 313s 259ms/step - loss: 1.1464 - accuracy: 0.6807 - val_loss: 1.0858 - val_accuracy: 0.7238
Epoch 6/8
1204/1204 [==============================] - 320s 266ms/step - loss: 1.1365 - accuracy: 0.6916 - val_loss: 1.1099 - val_accuracy: 0.7103

Epoch 00006: ReduceLROnPlateau reducing learning rate to 0.00020000000949949026.
Epoch 7/8
1204/1204 [==============================] - 310s 257ms/step - loss: 1.0721 - accuracy: 0.7336 - val_loss: 1.0282 - val_accuracy: 0.7593
Epoch 8/8
1204/1204 [==============================] - 310s 258ms/step - loss: 1.0514 - accuracy: 0.7504 - val_loss: 1.0072 - val_accuracy: 0.7790
Training finished!
```



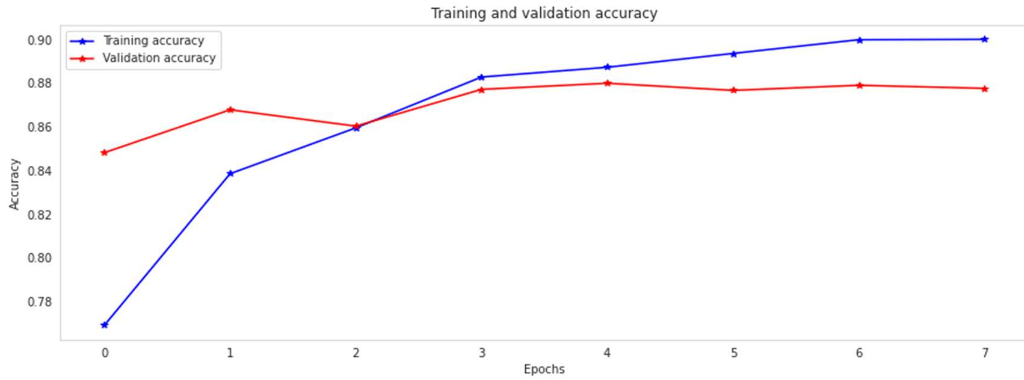

```
<Figure size 432x288 with 0 Axes>
```



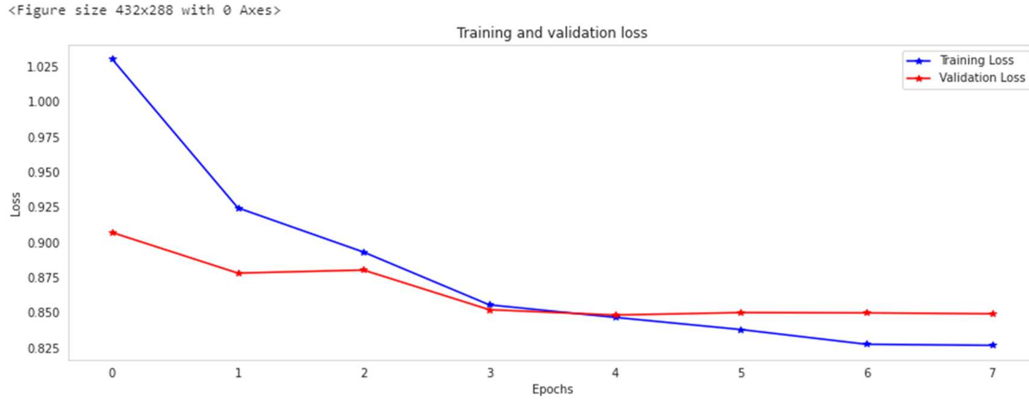

```
<Figure size 432x288 with 0 Axes>
```

InceptionV3: 87% val acc

```
Found 19257 validated image filenames belonging to 5 classes.
Found 2140 validated image filenames belonging to 5 classes.
Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/inception_v3/inception_v3_weights_tf_dim_ordering_tf_kernels_notop.h5
87916544/87910968 [==============================] - 1s 0us/step
Training fold no.: 1
Epoch 1/8
1204/1204 [==============================] - 316s 254ms/step - loss: 1.1237 - accuracy: 0.7125 - val_loss: 1.0583 - val_accuracy: 0.7430
Epoch 2/8
1204/1204 [==============================] - 303s 251ms/step - loss: 0.9488 - accuracy: 0.8262 - val_loss: 1.0512 - val_accuracy: 0.7393
Epoch 3/8
1204/1204 [==============================] - 310s 258ms/step - loss: 0.9203 - accuracy: 0.8433 - val_loss: 0.9566 - val_accuracy: 0.8252
Epoch 4/8
1204/1204 [==============================] - 304s 252ms/step - loss: 0.8920 - accuracy: 0.8597 - val_loss: 0.9476 - val_accuracy: 0.8271
Epoch 5/8
1204/1204 [==============================] - 307s 255ms/step - loss: 0.8922 - accuracy: 0.8596 - val_loss: 0.9099 - val_accuracy: 0.8379
Epoch 6/8
1204/1204 [==============================] - 308s 255ms/step - loss: 0.8752 - accuracy: 0.8713 - val_loss: 0.9397 - val_accuracy: 0.8360

Epoch 00006: ReduceLROnPlateau reducing learning rate to 0.0019999999552965165.
Epoch 7/8
1204/1204 [==============================] - 305s 253ms/step - loss: 0.8465 - accuracy: 0.8889 - val_loss: 0.8661 - val_accuracy: 0.8762
Epoch 8/8
1204/1204 [==============================] - 306s 254ms/step - loss: 0.8248 - accuracy: 0.9026 - val_loss: 0.8696 - val_accuracy: 0.8762

Epoch 00008: ReduceLROnPlateau reducing learning rate to 0.0003999999724328518.
Found 19257 validated image filenames belonging to 5 classes.
Found 2140 validated image filenames belonging to 5 classes.
Training fold no.: 2
Epoch 1/8
1204/1204 [==============================] - 310s 251ms/step - loss: 1.1312 - accuracy: 0.7117 - val_loss: 0.9618 - val_accuracy: 0.8243
Epoch 2/8
1204/1204 [==============================] - 300s 249ms/step - loss: 0.9488 - accuracy: 0.8271 - val_loss: 0.9668 - val_accuracy: 0.8117

Epoch 00002: ReduceLROnPlateau reducing learning rate to 0.0019999999552965165.
Epoch 3/8
1204/1204 [==============================] - 301s 250ms/step - loss: 0.9053 - accuracy: 0.8495 - val_loss: 0.8691 - val_accuracy: 0.8757
Epoch 4/8
1204/1204 [==============================] - 301s 250ms/step - loss: 0.8750 - accuracy: 0.8669 - val_loss: 0.8687 - val_accuracy: 0.8678
Epoch 5/8
1204/1204 [==============================] - 301s 250ms/step - loss: 0.8631 - accuracy: 0.8787 - val_loss: 0.8626 - val_accuracy: 0.8766
Epoch 6/8
1204/1204 [==============================] - 298s 248ms/step - loss: 0.8606 - accuracy: 0.8773 - val_loss: 0.8695 - val_accuracy: 0.8678

Epoch 00006: ReduceLROnPlateau reducing learning rate to 0.0003999999724328518.
Epoch 7/8
1204/1204 [==============================] - 297s 246ms/step - loss: 0.8467 - accuracy: 0.8911 - val_loss: 0.8527 - val_accuracy: 0.8757
Epoch 8/8
1204/1204 [==============================] - 299s 248ms/step - loss: 0.8459 - accuracy: 0.8887 - val_loss: 0.8537 - val_accuracy: 0.8757

Epoch 00008: ReduceLROnPlateau reducing learning rate to 7.999999215826393e-05.
Training finished!
```



<Figure size 432x288 with 0 Axes>



<Figure size 432x288 with 0 Axes>

Xception: 87% val acc

```
Found 19257 validated image filenames belonging to 5 classes.
Found 2140 validated image filenames belonging to 5 classes.
Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/xception/xception_weights_tf_dim_ordering_tf_kernels_notop.h5
83689472/83683744 [==============================] - 1s 0us/step
Training fold no.: 1
Epoch 1/8
1204/1204 [==============================] - 641s 523ms/step - loss: 1.1019 - accuracy: 0.7243 - val_loss: 0.9405 - val_accuracy: 0.8299
Epoch 2/8
1204/1204 [==============================] - 623s 518ms/step - loss: 0.9347 - accuracy: 0.8320 - val_loss: 0.9144 - val_accuracy: 0.8463
Epoch 3/8
1204/1204 [==============================] - 634s 526ms/step - loss: 0.8916 - accuracy: 0.8622 - val_loss: 0.9203 - val_accuracy: 0.8439

Epoch 00003: ReduceLROnPlateau reducing learning rate to 0.0019999999552965165.
Epoch 4/8
1204/1204 [==============================] - 624s 518ms/step - loss: 0.8654 - accuracy: 0.8742 - val_loss: 0.8659 - val_accuracy: 0.8808
Epoch 5/8
1204/1204 [==============================] - 623s 518ms/step - loss: 0.8476 - accuracy: 0.8857 - val_loss: 0.8597 - val_accuracy: 0.8780
Epoch 6/8
1204/1204 [==============================] - 623s 517ms/step - loss: 0.8415 - accuracy: 0.8921 - val_loss: 0.8610 - val_accuracy: 0.8766

Epoch 00006: ReduceLROnPlateau reducing learning rate to 0.0003999999724328518.
Epoch 7/8
1204/1204 [==============================] - 623s 517ms/step - loss: 0.8289 - accuracy: 0.9002 - val_loss: 0.8615 - val_accuracy: 0.8771

Epoch 00007: ReduceLROnPlateau reducing learning rate to 7.999999215826393e-05.
Epoch 8/8
1204/1204 [==============================] - 623s 517ms/step - loss: 0.8314 - accuracy: 0.8998 - val_loss: 0.8603 - val_accuracy: 0.8794

Epoch 00008: ReduceLROnPlateau reducing learning rate to 1.599999814061448e-05.
Found 19257 validated image filenames belonging to 5 classes.
Found 2140 validated image filenames belonging to 5 classes.
Training fold no.: 2
Epoch 1/8
1204/1204 [==============================] - 630s 518ms/step - loss: 1.1080 - accuracy: 0.7162 - val_loss: 0.9070 - val_accuracy: 0.8481
Epoch 2/8
1204/1204 [==============================] - 623s 517ms/step - loss: 0.9261 - accuracy: 0.8388 - val_loss: 0.8781 - val_accuracy: 0.8678
Epoch 3/8
1204/1204 [==============================] - 623s 517ms/step - loss: 0.8985 - accuracy: 0.8559 - val_loss: 0.8803 - val_accuracy: 0.8603

Epoch 00003: ReduceLROnPlateau reducing learning rate to 0.0019999999552965165.
Epoch 4/8
1204/1204 [==============================] - 623s 517ms/step - loss: 0.8641 - accuracy: 0.8773 - val_loss: 0.8521 - val_accuracy: 0.8771
Epoch 5/8
1204/1204 [==============================] - 623s 518ms/step - loss: 0.8493 - accuracy: 0.8857 - val_loss: 0.8483 - val_accuracy: 0.8799
Epoch 6/8
1204/1204 [==============================] - 623s 517ms/step - loss: 0.8384 - accuracy: 0.8937 - val_loss: 0.8501 - val_accuracy: 0.8766

Epoch 00006: ReduceLROnPlateau reducing learning rate to 0.0003999999724328518.
Epoch 7/8
1204/1204 [==============================] - 628s 521ms/step - loss: 0.8236 - accuracy: 0.9039 - val_loss: 0.8499 - val_accuracy: 0.8790

Epoch 00007: ReduceLROnPlateau reducing learning rate to 7.999999215826393e-05.
Epoch 8/8
1204/1204 [==============================] - 629s 522ms/step - loss: 0.8259 - accuracy: 0.9001 - val_loss: 0.8492 - val_accuracy: 0.8776

Epoch 00008: ReduceLROnPlateau reducing learning rate to 1.599999814061448e-05.
Training finished!
```