This is the preparation part

Firstly,

We import the required python packages, including pandas tensorflow, keras, matplotlib, imageDataGenerator

We set the basic parameter of **Matplotlib**

**Random seed**

We use 42 as our random seed to control variates.

**Loading the data**

**Train and valid**

We use tf.keras.preprocessing.image\_dataset\_from\_directory to

Generates a tf.data.Dataset from image files in the directory.

Because the dataset that we are going to use only has train and test. As a result, we need to split train into train and validation, and the ratio is 8 to 2.

**Test**

tf.keras.preprocessing.image.ImageDataGenerator

Generate batches of tensor image data with real-time data augmentation.

there are 12 classes at this dataset,

**Building the architecture of the model**

The first model is EfficientNetB0, and then we add an GlobalAveragePooling2D layer, and finally add output layer to fit the data. We use softmax as the activation because, the dataset is Multiclass classification.

**Freezing the pretrained layers**

pretrained model, we do this because it can save computation time.

**Callbacks**

**ModelCheckpoint callback**

Save the best model

**Early stopping**

terminate the training as soon as the accuracy stops improving for 2 consecutive epochs

**ReduceLROnPlateau**

Reduce learning rate when a metric has stopped improving.

**Compiling the model**

Loss function we use is categorical cross entropy,

Optimizer Adam learning rate 0.01

Metrics is accuracy

And then, we ran the model for 5 epochs

**Figure**

The accuracy and valid accuracy is increasing, and loss, valid loss is decreasing.

**Unfreezing the pretrained layers**

We do the above steps again

This time we can see after we unfreezing the model the valid accuracy is much higher than before, which means the model is better.