ECE5420
Kaggle Competition Report
Priyam Abhaybhai Patel
Net ID: pp386

I started off with using Librosa for conversion of audio to spectrogram and mel spectrogram. I tried converting the spectrograms to tensors but I wasn't able to create a pytorch data loader with the data. So, I tried using torchaudio for the same but again wasn't able to create a data loader to fit into the pytorch model.

So, I finally came to using scipy for the conversion of audio to spectrogram. Thereafter I used a simple convolutional neural network using 2 layers to get test accuracy of 90.8%. After doing this I went back to the data again and realized that the training data had no 43 labels but the testing data had 43 labels. We had all combinations of numbers except 43 which led me to think about the problem in a different way. I modified the training labels to be 1,2,3,4 and treated the given problem as a multilabel classification problem instead of multiclass. I trained the same Convnet on these 4 labels and was able to increase the accuracy on the test set to 92%. This increase led me to believe I was going the right direction. I decided to apply a MultiOutputClassifier (MOC) with XGBOOST backbone using SKLearn. MOC trains 4 XGB classifiers, one for each label and predicts according to the learned features. At the output of the MOC, I used the top 2 labels that were predicted. Using this bumped up the accuracy to 93%. After that I tried replacing XGB with Random Forest and KNN however the results were poorer. Now I felt I reached a bottleneck as I thought I did everything I could. During the discussion session, Sourbh suggested I try to create synthetic data for the model. I tried using ydata-synthetic, however, I wasn't able to create any synthetic data.

So, I created independent classifiers for the labels 3 and 4. I trained 2 XGB classifiers one for 3 and one for 4 where since the initialization of the XGB parameters was different, the classifier was different than that of the MOC with 4 XGB classifiers. After this, I used a kind of voting ensemble to combine the outputs of the 2 individual classifiers and the MOC. I used an if condition block for the predictions. I decided that if both the individual predicted the presence of 4 and 3 and the MOC didn't predict 43, then the label is 43. This bumped up the accuracy significantly to 94.5%. I decided to create more individual models functioning as a quality check over the MOC.

I trained more classifiers as quality check over my MOC. I trained 2 more XGB classifiers for the labels 1 and 2 respectively.

Now I thought all the models being XGB wouldn't give the best results because some times even the quality check models would go wrong on the 43 labels. So, I decided to train some neural networks as well to support the MOC. I trained a Multi label Resnet18 as a supporting model to the MOC and also trained 2 more ResNet18s to classify 3 and 4.

Modifying the quality check conditions, I was able to get 98.4% accuracy on the test data which got me 7th place in the competition. Code for quality check can also be found in the best model notebook attached herewith. I could've gotten better results with synthetic data however, I wasn't able to get the solution ready in time.

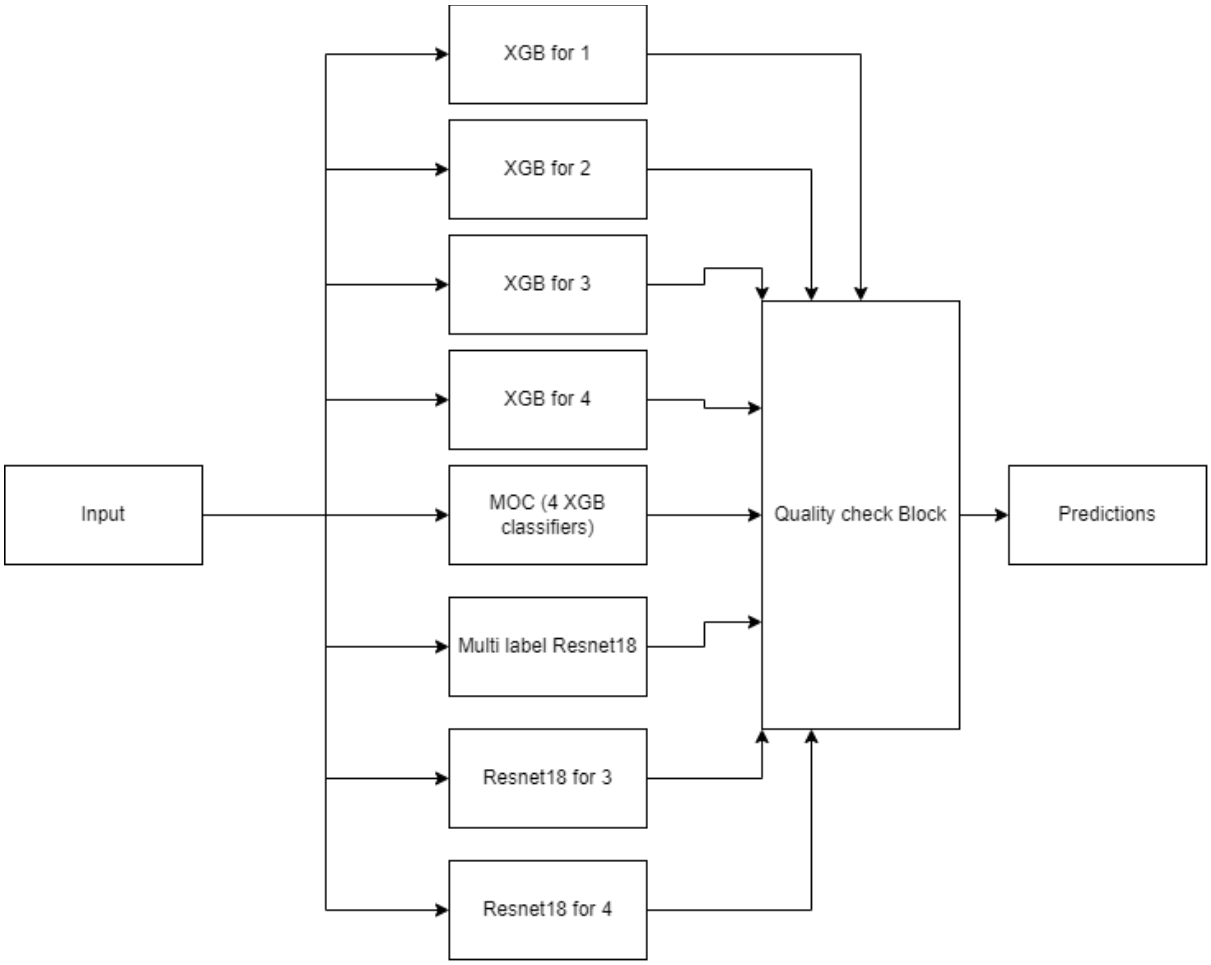

Figure 1 Block diagram of models. I used a total of 11 models (MOC has 4 component models) to get my best result