FINAL STATUS REPORT:

Speech Emotion Recognition

CENG 3522, Applied Machine Learning

Fatih Kıyıkçı - 170709032

Beyza Kurt - 160709017

Contents

1 OVERVIEW

In our final study, we used another data set in addition to the data set we used in our midterm study. We also recorded our own voice recordings. Models that we trained made prediction over these records.

The data set (**RAVDESS**) we use in our midterm study contains **1440 sound files** that voiced by 24 actors. The data set (**TESS**) we decided to use additionaly contains **2800 sound files** that voiced by 2 actresses.

- Both of the data sets' files in the form of **.wav**.
- The RADVESS emotions includes **calm, happy, sad, angry, fearful, surprised, neutral** and **disgust**. The TESS emotions includes RADVESS' all emotions except calm.
- The RADVESS' file naming format is ‘modality-vocalChannel-**emotion**-emotionalIntensity-statement-repetition-actor.wav’. Example: 03-01-06-01-02-01-12.wav
- The TESS file naming format is ‘**emotion_**number.wav’. Example: angry_1.wav

The goal of the project is to *predict a reliable and accurate guess* for the dominant emotion of the speaker based on a model that is trained on this dataset.

2 PREPARATION

We used both data sets as one 'train' and one as 'test'. As a result, we decided to **combine the two data sets**. We determined the sets of 'train' and 'test' by separating the combined set randomly.

While making our emotion predictions, we have decided to try our models in two different approaches. First approach was to divide the emotions into two main groups in order to increase the accuracy and second one was to only train on 3 of the main emotions; happy, angry and calm. In our final studies, we decided not to use these approaches.

We trained our models with all emotions except 'disgust' and 'neutral' and they made their predictions accordingly.

We have used scikit-learn library for Naive Bayes and Multi Perceptron Classifier, And Keras with TensorFlow backend for the Convolutional Neural Network in our midterm study.

We continue to use MLP and 1D CNN but not Naive Bayes. Additionally, we started to use xgboost library for XGBoost Classifier.

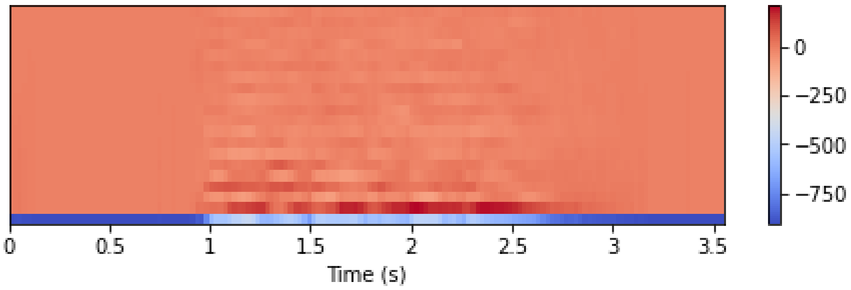
2.1 Data Cleaned Up, Transformed and Prepared Steps

To test a new audio record outside the dataset, we added noise to each record since it ‘standardizes’ the data and helps to get more accurate results.

1. Determine the sound’s emotion from the filename
   We used different methods because of difference in the file naming formats of both data sets.

2. If the emotion is in the unobserved emotions, no action is taken and continue
3. Turn the data to MFCC
   a. Read the sound data using SoundFile library as float format
   b. Add noise with our noise function, using the Librosa library
   c. Calculate the data's sample rate to use while turning the data to MFCC
   d. Extracted the features using mel-frequency spectrum, (MFCC) with 30 vectors, than taking the mean of these vectors and storing in an array.

Results of step 5 on file “03-01-02-01-01-01-01.wav”:



(1) *“03-01-02-01-01-01-01.wav”s MFCC Figure*

```
[-4.62363465e+02,  1.93141065e+01,  8.87577370e+00,  7.14135243e+00,
  2.43993952e+00, -9.74089950e-01, -4.25932592e+00, -5.82157993e+00,
  5.44761171e+00, -2.27374031e+00, -1.89958467e+00, -1.22407293e+00,
 -1.22379556e+00, -1.50669793e-01, -1.82045304e+00, -1.94762835e+00,
 -1.20001645e-01, -1.81948987e+00, -3.26427262e+00, -1.66190014e+00,
 -2.76245221e+00, -3.24186520e+00, -1.11511686e+00, -2.37879555e+00,
 -1.21561270e+00, -9.86078240e-01, -1.09960832e+00, -1.14329250e+00,
 -1.00317273e+00, -1.00232412e+00]
```

(2) *“03-01-02-01-01-01-01.wav”s MFCC’s Array Form*

3 THE MACHINE LEARNING PROBLEM

Supervised learning uses labeled data to learn the mapping function that turns input variables into the output variable (labels) . Which allows the model to accurately guess the output (more or less). Classification is one of supervised learning types.

Since for every audio file, we have the corresponding emotion labeled, We used the data to build a classifier model to predict the corresponding emotion for an unseen audio file.

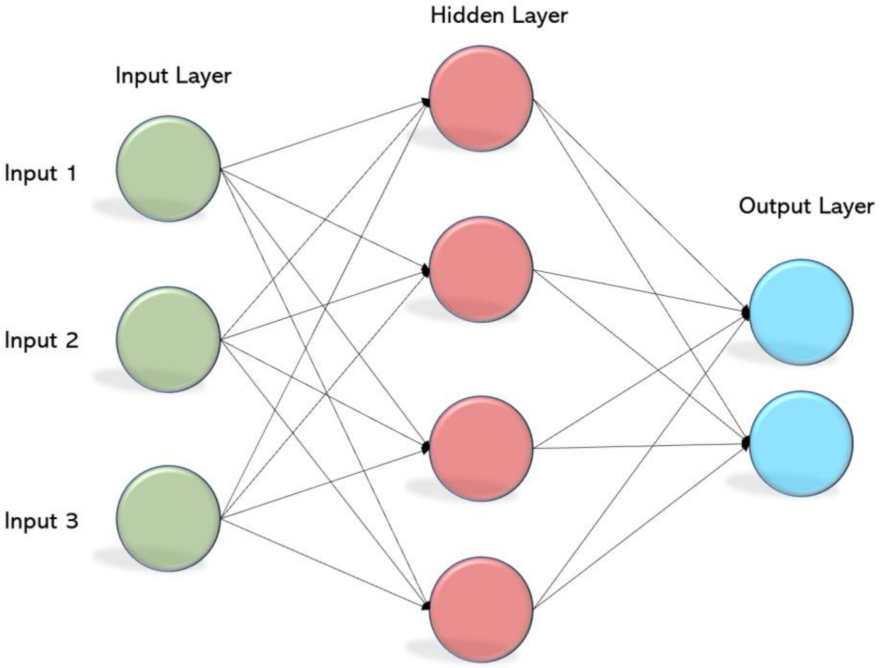
The problem in our case is to learn the feature and label relationship for every corresponding audio-label, which will build a classifier model, then classify unseen audio files, according to their respective emotions.

We experimented with the following classifier algorithms to classify our data:

1) MLP Classifier
2) XGBoost
3) Voting Classifier
4) 1D Convolutional Neural Networks

3.1 MLP Classifier

MLP Classifier is an artificial neural network. It is often seen as a simple but effective neural network classifier relative to the Convolutional Neural Networks.



Parameters:

alpha: Is called as the ‘penalty term’, high alpha values reduces the chance of overfitting, similarly lower alpha values reduces the chance of under fitting. In our experiments, 0.001 was seem to be the optimal value.

hidden_layer_size: The number of hidden layers and number of nodes in each layer. We used 5 layers with 64, 32, 32, 16, 8 nodes respectively, for the model architecture of hidden layers, Based on our experiments, this was the most successful architecture for MLP.

batch_size: 32 samples for each epoch

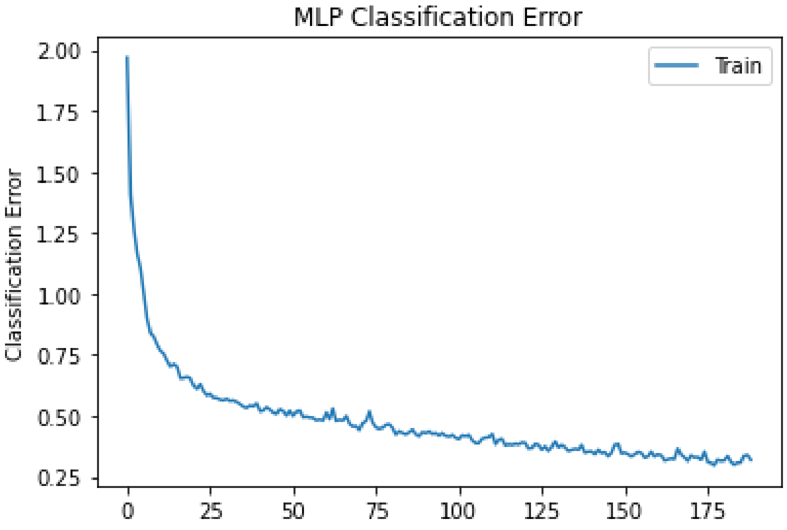
max_iter: Maximum number of iterations.

learning_rate: Learning rate schedule for weight updates.

‘adaptive’ keeps the learning rate constant to ‘learning_rate_init’ as long as training loss keeps decreasing.

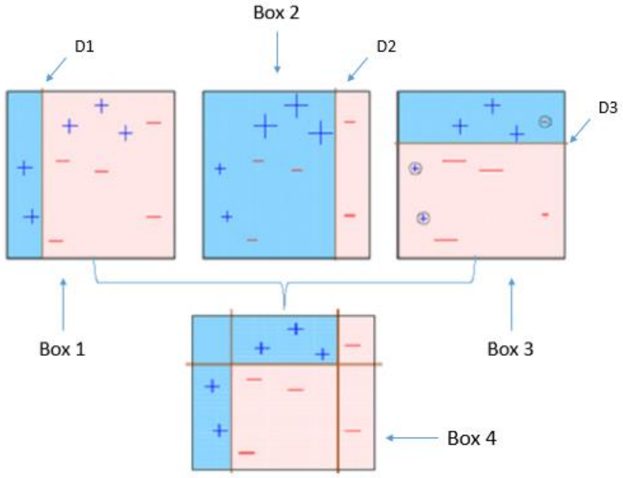
Average values for 6 classes:

```
              precision    recall  f1-score   support

        calm       0.85      0.88      0.87       121
       happy       0.67      0.76      0.71        34
         sad       0.94      0.68      0.79       114
       angry       0.73      0.88      0.80       120
     fearful       0.83      0.77      0.80       119
   surprised       0.82      0.85      0.84       123

    accuracy                           0.81       631
   macro avg       0.81      0.81      0.80       631
weighted avg       0.82      0.81      0.81       631
```



3.2 XGBoost

We used *XGBoost Classifier* algorithm, it is a very popular and successful algorithm for classifying and regression. (especially in kaggle competitions.). It is an ensemble algorithm and uses boosting, as the name suggests.



We tuned the parameters on the experimentation on the data and prior experience about the algorithm.

Parameters:

colsample_bytree = 0.2, ratio for subsampling, when constructing each tree. We kept it low due to the quick convergence and overfitting.

max_depth = 4, Again, high max depth values will lead to overfitting, due to increase in complexity.

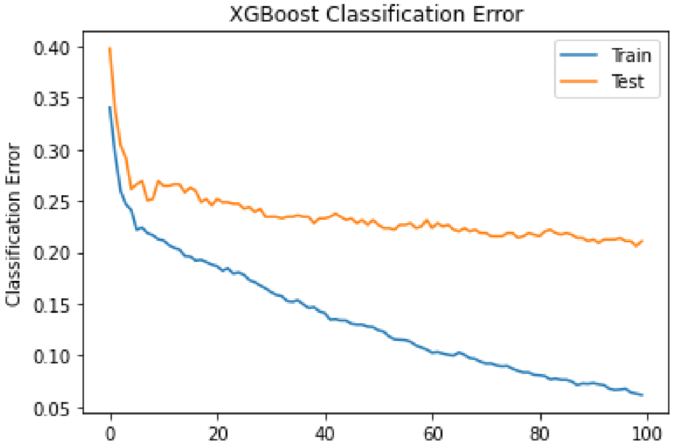
min_child_weight = 1.7817, the least number of samples for a node to represent, increasing can help the model generalize more.

n_estimators = 100, number of trees, increasing will increase complexity, but it can be beneficial with the according use of learning_rate.

subsample = 0.2, ratio of data per tree. Every tree gets a random %20 of data and train on that. Again, we kept it low to avoid overfitting

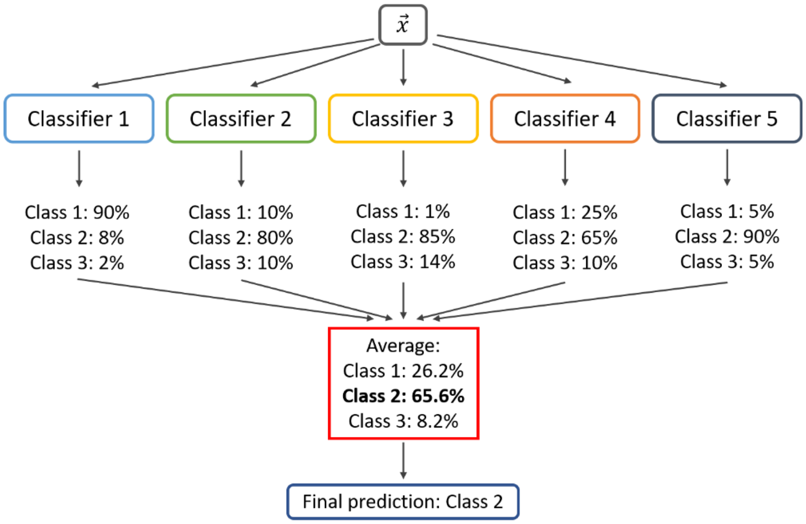
Prediction values for 6 features;

```
              precision    recall  f1-score   support

        calm       0.85      0.83      0.84       121
       happy       0.51      0.59      0.55        34
         sad       0.80      0.75      0.78       114
       angry       0.77      0.76      0.76       120
     fearful       0.76      0.72      0.74       119
   surprised       0.79      0.87      0.83       123

    accuracy                           0.78       631
   macro avg       0.75      0.75      0.75       631
weighted avg       0.78      0.78      0.78       631
```



3.3 VotingClassifier

We used *Voting Classifier* algorithm of sklearn, We were happy with some of the results with mlp and some of the results with xgboost. We thought it would be a good idea to use ensembling here.



Parameters:

estimators: XGBoost and MLP models

voting: Hard. Hard voting is takes the probabilities for each class and uses their average for the prediction.

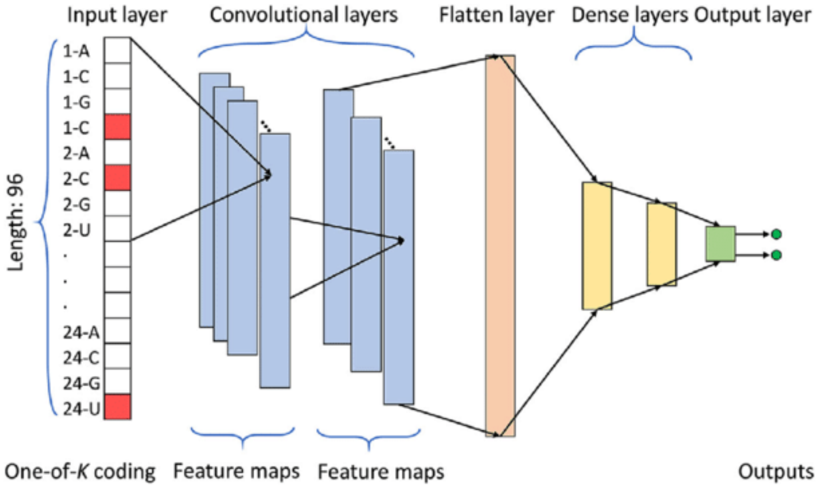
Prediction values for 6 features;

```
              precision    recall  f1-score   support

        calm       0.82      0.93      0.87       121
       happy       0.42      0.79      0.55        34
         sad       0.84      0.77      0.80       114
       angry       0.83      0.76      0.79       120
     fearful       0.88      0.71      0.78       119
   surprised       0.86      0.84      0.85       123
    accuracy                           0.80       631
   macro avg       0.77      0.80      0.77       631
weighted avg       0.82      0.80      0.81       631
```

3.4 1D Convolutional Neural Networks

Another algorithm to try our data was one dimensional convolutional neural network.

Before feeding, we reshaped the data and feed into the CNN. CNN is known to be good with image classification, our vectors are not much different, we thought CNN can make a good job at classifying audio features. We changed the n_mfcc parameter to 64, which specifies the number of mfcc samples to return just to make a good fit for max pooling.



Parameters:

Kernel size: is 8 since it is a 1d CNN.

Activation function: We used relu as our activation function for the hidden layers and softmax for the output layer.

Layers: We used 8 1D Convolutional Layers depending on the “deeper is better” approach in CNN’s

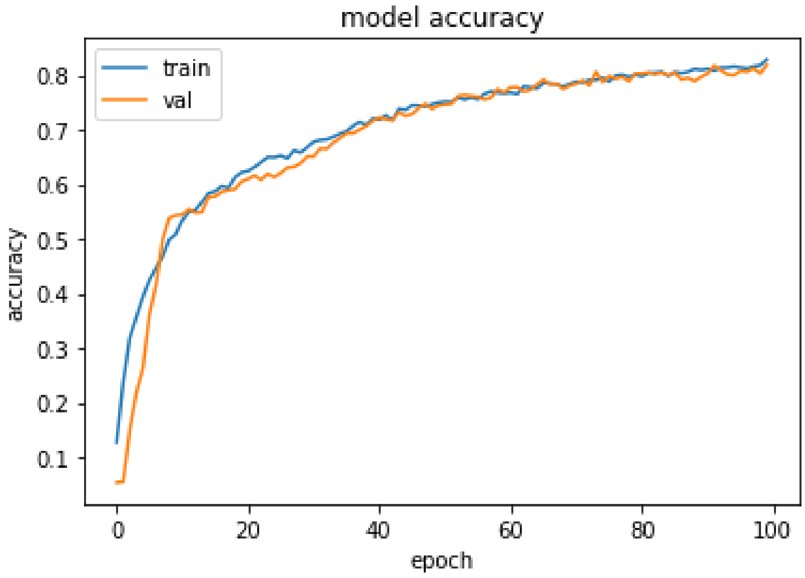
Max pooling: We used 2 max poolings of 6-5 since we have 30 mfcc’s for each data object.

Batch size: 128

epochs:100

Experimentation results for 1D-CNN, predicting 6 features:

```
              precision    recall  f1-score   support

        calm       0.89      0.89      0.89       121
       happy       0.51      0.82      0.63        34
         sad       0.79      0.84      0.82       114
       angry       0.84      0.81      0.82       120
     fearful       0.88      0.71      0.79       119
   surprised       0.85      0.84      0.84       123

    accuracy                           0.82       631
   macro avg       0.79      0.82      0.80       631
weighted avg       0.83      0.82      0.82       631
```



4 COMPARISON OF RESULTS FOR DIFFERENT MODELS

We used the models to predict the emotions of the sound recordings we recorded. In our data set, there are sound recordings containing 2 'angry', 3 'calm', 1 'fearful', 2 'sad' and 1 'surprised' emotions. Sound files can be found in the ‘sounds’ drive folder.

Results:

mlp: ['sad' 'angry' 'fearful' 'sad' 'happy' 'sad' 'angry' 'happy' **'angry'**]

xgb: [**'calm' 'calm'** 'fearful' 'fearful' **'angry'** 'calm' 'fearful' 'surprised' **'angry'**]

ensemble: [**'calm' 'calm' 'fearful'** 'fearful' **'angry'** 'calm' 'angry' **'sad' 'angry'**]

1d-cnn: ['sad' 'sad' 'angry' 'fearful' 'sad' **'sad'** 'fearful' **'sad' 'angry'**]

true: [**'calm' 'calm' 'fearful'** 'calm' **'angry' 'sad'** 'surprised' **'sad' 'angry'**]

Accuracies:

mlp:	0.11
xgb:	0.44
ensemble:	0.66
1d-cnn:	0.33

When we combened TESS and RADVESS and randomly selected %25 as a test from it the **accurisies** for each model are:

mlp:	82.57%
xgBoost:	77.65%
ensemble(xgb, mlp):	80.51%
1D-CNN:	81.93%