ECE539 Project Report

PROJECT TITLE: Singing livestream segmentation assistant

TEAM MEMBERS:

- Avi BALAM, 9086006591, abalam@wisc.edu, Undergraduate
- YUSHANG JIN, 9083280140, yjin248@wisc.edu
- ZALISSA ZONGO KAFANDO, 9084045047, zongokafando@wisc.edu

Date of submission: 11/18/2023

ABSTRACT

A comprehensive approach to audio classification was developed and implemented. The project involved the loading and preprocessing of audio data, transforming the waveform into a spectrogram using short-time Fourier transform (STFT). A pre-trained convolutional neural network (CNN) model was then employed to predict class probabilities for each chunk of the audio stream, distinguishing between singing and speech. The predictions were further refined using a multilayer perceptron (MLP) model to reduce noise. The accuracy of the predictions was evaluated by comparing them with actual labels, demonstrating the effectiveness of the approach.

Introduction

Our target is to assist in slicing singing livestream videos, that is help cutting the singing parts out from the whole livestream recording, by highlighting the potential singing partitions. The core is audio classification that distinguishes between singing and speech. "Classifying Music and Speech with Machine Learning" by Code AI demonstrated an approach to classifying audio into music or speech categories. They transfer audio into spectrogram using FFT and use a CNN model for classification. With gtzan dataset (see below) the validating accuracy reaches 100%.

Method

We divide the task into several Notebooks:

- `music_speech_clf.ipynb`: train the classifier CNN
- `pred_series_mlp_preprocess.ipynb`: MLP data preprocessing
- `pred_series_mlp_training.ipynb`: MLP training
- `slicer.ipynb`: predict (inference) using the whole model

And block diagrams are shown:

- Singing speech classifier training

```
[xxm_singing]-->(FFT)-->[Spectrogram]-->input ------\
                        'singing'(1)-->label ---\   \
                                               (Classifier CNN)
```

```
[xxm_speech] -->(FFT)-->[Spectrogram]-->input ---/   /
                        'speech' (0)-->label ------/
```

- Singing speech classifier inference

```
[Long audio]-->[CHUNK 1][CHUNK 2] ... [CHUNK N]
                    |
                    V
                (FFT)-->[Spectrum]-->(CNN)-->[Probability of each chunk]
```

- Predicted probability series MLP preprocessing

  Extract prob. series of 150 seconds around each marker as positive input:

```
                                    'singing'(1)  -->label
[.........M..........] --> (map to probabilities)-->input
|<--60s-->|<---90s--->|

start chunk = (m-60) * SAMPLE_RATE / STEP_SIZE
end chunk  =  (m+90) * SAMPLE_RATE / STEP_SIZE

MLP input size = 150 * SAMPLE_RATE / STEP_SIZE
            = 150 * SAMPLE_RATE / (CHUNK_SAMPLE/2)
            = 150 * 22050 / (132300/2) = 50
```

  And at time where there is no marker, we choose them as negative input:

```
                                   'speech' (0) --> label
[.....................] -->(divide into pieces) --> input(s)
end[i]              start[i+1]
```

- Predicted probability series MLP training

  Train the MLP with data obtained by above preprocessing.

- predicted probability series MLP inference

```
[Prob. of CHUNK1] [Prob. of CHUNK2] ... [Prob. of CHUNK N]
        |                 |         |
        V                 V         V
    [Prob. series of CHUNK i to i+M] --> (MLP) --> [MLP Prob. of CHUNK i]
```

- Predict using the whole program

```
[Mixed audio]-->(Singing speech classifier)-->[prob. series]
                                                    |
  [outcome probability]<--(predicted series MLP)  <--
```

Sorry that to save time here I use this way to present. I will draw them clearly using Visio or Draw.io in our final report and presentation.

Results

We strictly followed the rule that test data DO NOT appear in training data.

Below are four outcomes with our current program. Not bad.