[Fall 22] Large Scale Multimedia Analysis **Name:** Ji In Kwak
Homework 1
September 28, 2022

Homework 1 Report

1 Task 2

1.1 With the default setting, which classifier performs better? Why?

Both SVM and MLP had a default setting for the sklearn classifier. With those default setting, MLP classifier performs better with accuracy 34.76% than the SVM classifier for the given test set.

1.2 Which default parameter do you want to tune? Why? How do you change it what's the difference?

Although the accuracy of MLP model was 34.76%, that of SMV classifier was 23.26% which is lower than the 30%. Therefore, I changed some input parameters. First of all, the gamma value is changed from 'auto' to 'scale'. And for the kernel option, I changed it as 'ovr'. Finally, the class weight option is added as 'balanced' because their might be the class imbalance problem between the 15 classes. With the revised setting of SVM classifier, the test set accuracy was increased to 31.55 which is still lower than the MLP classifier model.

2 Task 3

2.1 How do you generate a single-vector representation for each video using SoundNet features?

To generate a feature vector from the video data, the pre-trained SoundNet[1] model was used as a feature extractor. In the implementation step, I tried two approaches to generate those features (snf) for each video file. First was to directly extract the vector from the .wav files which we used in Task 2. From the python script, extract_soundnet_feats.py, it can get an input as wav format file, and then extract the single vector from SoundNet layers. The implementation code is as follows.

```
python extract_soundnet_feats.py --input_dir wav --output_dir
    snf --file_ext .wav --feat_layer conv7
```

Through this code, we can get the feature vectors of every video files not converting to the mp3 format files. While extracting the features, the error occured because the range of audio data exceeds [-256:256]. Hence, I added the normalization code for the input audio file in the load_audio function.

The second approach is adjusting the given ffmpeg scripts in Task 1. The SoundNet[1] paper represented that they preprocessed audio data by converting video to mp3 and set some parameters including sampling rate. Hence, the ffmpeg script converts from video files (.mp4) to the mp3 files (.mp3). The implementation code is as follows.

```
$ for file in videos/*;do filename=$(basename $file .mp4);
   ffmpeg -y -i $file -ac 1 -ar 22050 -f mp3 mp3/${filename}.
   mp3; done
```

Then, through the given script extract_soundnet_feats.py with the default argument setting, we can get the snf files which contains the 1024 dimension vectors for each video file.

2.2 You extract features from which layer? Why you choose this layer? What's the performance on the public test set?

I extracted the features from the Convolution layer 7 of the SoundNet. The last layer Conv8 is divided into two layers, so I used the previous one which is Conv7. However, we can try the other layers such as layer 5 because the pre-trained model is trained for the other audio dataset which is different from our test set.

3 Task 4

3.1 Introduce how do you improve the model. What's the performance? Why it works? If you have extra time and GPU resource, what's your next steps.

First, I divided the given dataset into train and validation set. Next, the pretrained model for audio processing, PaSST [2], was implemented and used as a feature extractor. PaSST is audio classification model with transformers. The final layer of the PaSST is used, and the extracted vector has 1295 dimensions.

If there is an extra time and GPU, we can fine tune the PaSST model with changing the last layer as the MLP layer with output size 15, which is the number of classes for our dataset.

3.2 What's the bottleneck of the current system?

In order to finetune the pretrained PaSST model, large memory and GPU is needed since the size of model is too big. Also, there was some limitations in improving the MLP classifier from scikit-learn package. Hence, we need to design our MLP network using PyTorch to increase the options such as batch-normalization or dropout layers.

References

[1] Yusuf Aytar, Carl Vondrick, and Antonio Torralba. Soundnet: Learning sound representations from unlabeled video. In *Advances in Neural Information Processing Systems*, 2016.

[2] Khaled Koutini, Jan Schlüter, Hamid Eghbal-zadeh, and Gerhard Widmer. Efficient training of audio transformers with patchout. *arXiv preprint arXiv:2110.05069*, 2021.