Homework 1: Audio-based Multimedia Event Detection

11-775 Large-Scale Multimedia Analysis (Spring 2021)

Due on: Wednesday March 8, 2021 11:59 PM

1 Task

There are three homework assignments in this course. You will learn to build the whole multimedia event detection (MED) pipeline step-by-step. In homework 1, you will learn to process the audio part in videos to build audio-based MED pipeline that classifies videos into 10 classes. In homework 2, you will learn to build visual-based models. You will investigate multimodal fusion in homework 3. You will have two weeks for each homework. Please **START EARLY**! It will take time to understand different tools and modules in the pipeline. Also, the feature extraction process may be time-consuming.

2 MED Pipeline Overview

The overview of MED pipeline is depicted in Fig. 1. The task of homework 1 is to perform MED with audio features. In the first step, we extract features from the raw training data. In this homework, the audio features we will use are MFCC and the self-supervised SoundNet features. For MFCC, once the features are extracted, the parsing part further does some processing on the extracted raw features to pack them into the final representation, so that each video is represented by a single feature vector. Specifically, in this homework, we ask you to implement the bag-of-words representation with k-means clustering for MFCC. With the extracted representations and the labels, two typical approaches for training classifiers are Support Vector Machine (SVM) and Multi-Layer Perceptron (MLP) you learned in the class. The trained classification models are then to be used in the testing phase. For testing, we extract and pack video representations on the testing data with the same parameters/models used in the training phase. With the trained classifiers, we then score those videos accordingly and evaluate the MED system. In short, in the first homework, you will -

1. Learn to extract the MFCC and neural network features from the audio part of the videos. Train k-means (k-words), and represent videos as bag-of-audio-words.
2. Train and test classification models (SVM and MLP) for MED.

To help you get started, we provide you an exemplary framework. Please refer to **`spring2021/hw1/`** in the GitHub repository of this course: `https://github.com/11775website/11775-hws.git`. Please note that the provided code is just an example setup that helps you to understand the basic components in the MED system. Also, the original parameters may not perform well. Therefore, you are NOT required to follow the provided example pipeline. We encourage you to create your own pipeline and try different tools, features, and configurations to get better results. The only requirement is to write a **`README.md`** in your GitHub repository to explain how to run your pipeline.

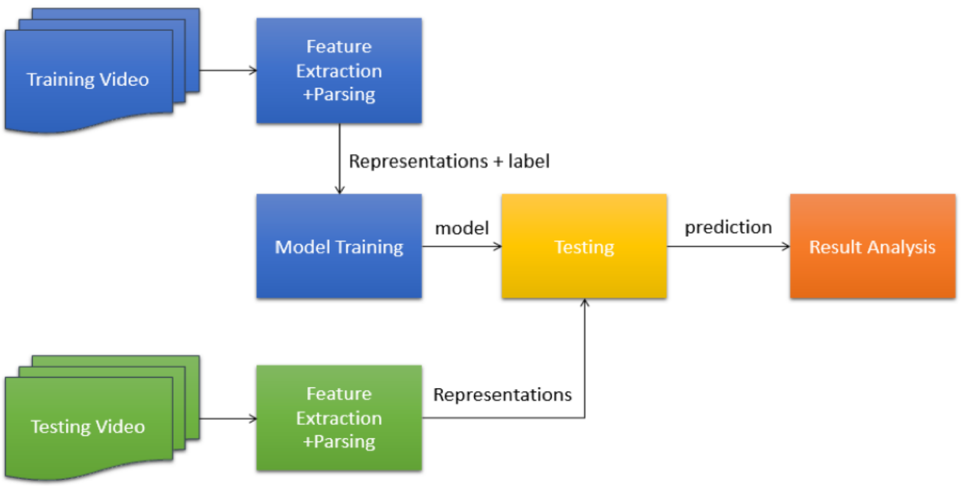

Figure 1: Overview of MED pipeline.

Figure 2: A visualization of events in the dataset: *dribbling basketball, playing guitar, shoveling snow, and mowing lawn*.

3 Dataset

For this course TAs collected a special audio-visual dataset that contains 7,942 10-sec videos (around 22 hours of video in total). There are 5,662 videos for training/validation and 2,280 videos for testing. The dataset will be the same for all three homework assignments. There are 10 classes/events in this dataset. Specifically, C00: *dribbling basketball*, C01: *mowing lawn*, C02: *playing guitar*, C03: *playing piano*, C04: *playing drums*, C05: *tapping pen*, C06: *blowing out candles*, C07: *singing*, C08: *tickling*, and C09: *shoveling snow*. The visualization can be found in Fig. 2. For simplicity, each video in the dataset contains only one event. The event-name to ID mapping is provided in **`cls_map.txt`**. Please read README.md for more details about the dataset.

For training and validation, the file **`trainval.label`** specifies the 5,662 videos and their ground-truth labels. The beginning of the label file is:

Example 1

```
Id,Category
HW00006645,6
HW00001236,6
HW00004004,6
```

It means that HW00006645.mp4, HW00001236.mp4, and HW00004004.mp4 belong to C06: *blowing out candles*. You should shuffle and split the label file into your own training and validation set (e.g., 5,200 videos for training and 462 videos for validation). You may then use the validation

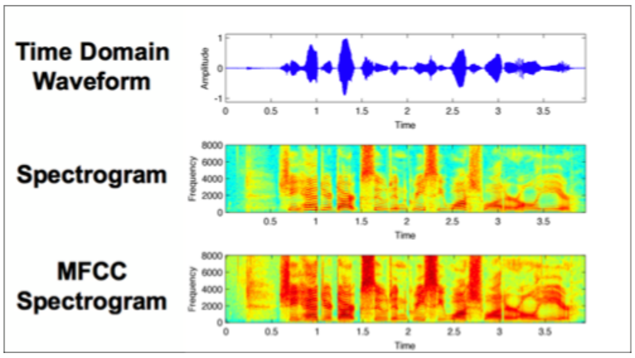

Figure 3: MFCC features

set to tune hyper-parameters, conduct ablation studies, and report your interesting findings in the report.

For testing, there are 2,280 testing videos specified in the **`test_for_student.label`**, in which their labels are excluded deliberately. For instance, the first three lines in **`test_for_student.label`** are:

Example 2

```
HW00002897.mp4
HW00001276.mp4
HW00000794.mp4
```

You are expected to use your MED system to classify every testing video into ten event types with the two audio features described in the next section.

4 Audio Features

4.1 MFCC Features

In the class you have learned some popular audio features. Mel-frequency cepstrum (MFC) is a representation of the short-term power spectrum of a sound, based on a linear cosine transform of a log power spectrum on a nonlinear mel-scale of frequency. Mel-frequency cepstral coefficients (MFCCs)[1] are coefficients that collectively make up an MFC.

One of the well-known tools to extract MFCCs is openSMILE. For each video, you may use ffmpeg to extract the audio part of the video and extract the MFCC feature as follows:

1. ffmpeg: extract mono audio track (.wav) from the video file.
   ```
   ffmpeg -y -i HW00006645.mp4 -ac 1 -f wav HW00006645.wav
   ```
2. openSMILE: extract MFCC feature
   ```
   SMILExtract -C config/MFCC12_0_D_A.conf
   -I HW00006645.wav -O HW00006645.mfcc.csv
   ```

We suggest you to tweak the configuration file in **`config/MFCC12_0_D_A.conf`** after you finish the pipeline to get a better sense of MFCC features and improve the MED performance.

[1]More information can be found here.

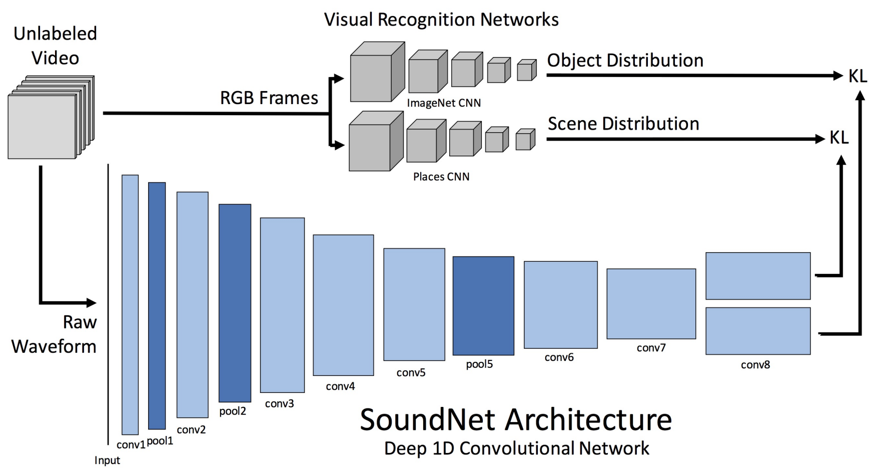

Figure 4: SoundNet architecture

4.1.1 Bag-of-Words Representation

As studied in class, representing a video using bag-of-(audio-)words is one of the popular approaches. It reduces high-dimensional features, learns shared concepts (words), and results in a more compact representation. In this part, we will train a k-means clustering model to cluster the MFCC vectors to represent a video using these cluster centers.

To speed up the clustering process, you can choose only a small portion of the MFCC vectors (e.g., randomly select 20% MFCCs from each video). It will reduce the size of data for k-means clustering and thus speed up your pipeline development. It is up to you whether to use this speed up strategy.

Please implement functions to train a k-means model or using other clustering algorithms to represent videos (this part is not included in our exemplary pipeline code). There are many possible ways to do so. For example, you may set $k = 200$ to train 200 cluster centers using Python sklearn package for k-means. Once you have the clustering, you then represent each video in a k-dim vector according to the histogram where each dimension indicates the counts of MFCCs to the “closest” cluster center. You may also try different distance measurements to find out which approach is more suitable for audio features. Please note that exception handling may be required for videos with slightly shorter or longer audio tracks.

4.2 SoundNet

SoundNet developed by the MIT CSAIL Lab learns rich natural sound representations by capitalizing on large amounts of unlabeled sound. It yields significant performance improvements over the state-of-the-art results on standard benchmarks for acoustic classification. You can extracted the raw features from different layers (conv3, conv4, conv6 and conv8) of SoundNet pre-trained model using an open-source Tensorflow implementation and use the provided **`SoundNet-tensorflow/extract_feat.py`**. You could load them in Python using numpy as follows (e.g. the conv3 feature of HW00006645):

Example 3

```
import numpy as np
raw_feat=np.load('HW00006645_conv3.npz')
```

```
raw_feat=raw_feat['arr_0']
...
```

The raw features are 4-dimensional arrays in (N, H, W, C) format, whereas N is the batch size of the feature, H and W is the size of feature maps and C is the number of filters. How to generate the vector-like representation for each video? You can refer to the implementation in the original paper. We expect you to try different SoundNet features (e.g., extract features from different layers in Soundnet) and employ various kinds of feature aggregation strategies and report your findings on the dataset.

5 Model Training and Testing

With the extracted audio feature, now we can train and test the classification model for MED. In this homework you learn how to train Support Vector Machine (SVM) and Multi-Layer Perceptron (MLP) models. Again, you don't have to re-invent the wheel. There are lots of tools available such as scikit-learn and libsvm. Your goal is to integrate these tools into your pipeline. You may also try writing functions to tune SVM parameters such as cost and kernels and number of hidden units or layers in MLP to get better results .

Evaluation on the validation set provides important information to determine the model performance. A model performs well in the validation set is likely to perform well in the testing set (assume the data are i.i.d.). To this end, you may want to define a criteria (e.g. top-1 accuracy) and select the best-performing model in the validation set and use it for testing.

For model training and testing, here are the steps you may follow:

- Train the classification model with the extracted MFCC and SoundNet features.
- Apply the trained classifier to the validation set you defined. For each video, your model predicts event happened in the video. Define a criteria (e.g., top-1 accuracy) and try different setups and find the best model in the validation set.
- Use the model to predict events of the testing videos in **`test_for_student.label`**. Please output a csv file to store such prediction. The order of the videos in the original testing list should be kept (see Example in **`hw1_sample_submission.csv`**. You can submit to Kaggle `https://www.kaggle.com/c/11775-hw1` to estimate about how well your model performs on the (partially-hidden) testing set.
- For the canvas submission, please name your best submission file **`best.csv`** and specify your method in the report.

To sum up, we expect you to submit **`{mfcc|soundnet|best}.csv`**. For your reference, we provide you a baseline implementation under **`spring2021/hw1/`**. A successful submission should at least aim at getting comparable results to the baseline TAs submit to Kaggle. Submissions which significantly exceed our baseline will get extra points.

Note that you can also add other submissions which can be a mix (fusion) of MFCC and SoundNet, or you can add any other new features. You are also encouraged to try different classification models as well.

6 What to submit

In Canvas: Please compress your submission into **`ANDREWID_HW1.zip`** and submit it through the turn-in link in Canvas. In order to evaluate your submission easier, the overall contents of your **`ANDREWID_HW1.zip`** should be organized as the following:

1. **report.pdf**: Your pdf report for homework 1.
2. **github_kaggle.txt**: A txt file with the link to your GitHub repository and your Kaggle account. Please be sure to add a README.md explaining how to run your code or script.
3. **mfcc.csv**, **soundnet.csv**, and **best.csv**: The classification results for each testing video. The submission csv format is:

Example 4

```
Id,Category
HW00002897,7
HW00001276,1
HW00000794,0
```

In the report, first you are required to describe the detailed steps and parameters in your MED pipeline with two types of features: MFCC and SoundNet. Secondly, please specify the size of your training and validation set and the evaluation metric you use (e.g., top-1 accuracy, top-5 accuracy, or average precision) for validation. Please report your model performance on the validation set. We ask you to also report the confusion matrix. Which class(es) is harder and with which it is confused? Lastly, we expect you to report the time your MED system takes (CPU time) for feature extraction and classification on the testing set. Please also tell us the amount of credits left on your AWS account after you finish homework 1.

In Kaggle: Please submit your **best.csv** to the Kaggle leaderboard on `https://www.kaggle.com/c/11775-hw1` and briefly explain your method. You can submit up to 5 times/day. The final performance on the whole testing set will be revealed on March 9.

7 AWS

In this course you will learn how to use Amazon Web Services (AWS). You may develop your MED pipeline on a t2.large instance with a storage around 100 GB. If you want to use GPU, we suggest you use p2.xlarge and reduce the batch size to fit the GPU memory.

Cost for t2.large is around \$0.09/hr and p2.xlarge is around 0.9/hr. We have provided you with a \$50 coupon, so you do not need to spend any of your own money for the homework. However, please note that AWS charges you when you start the instance whether it is idle or not. Therefore, please remember to **stop** your instance (**don't terminate it!**) when you don't need it, especially for the expensive GPU instance! You should first develop a stable pipeline on a CPU instance then dive into the neural models that may require a GPU instance.