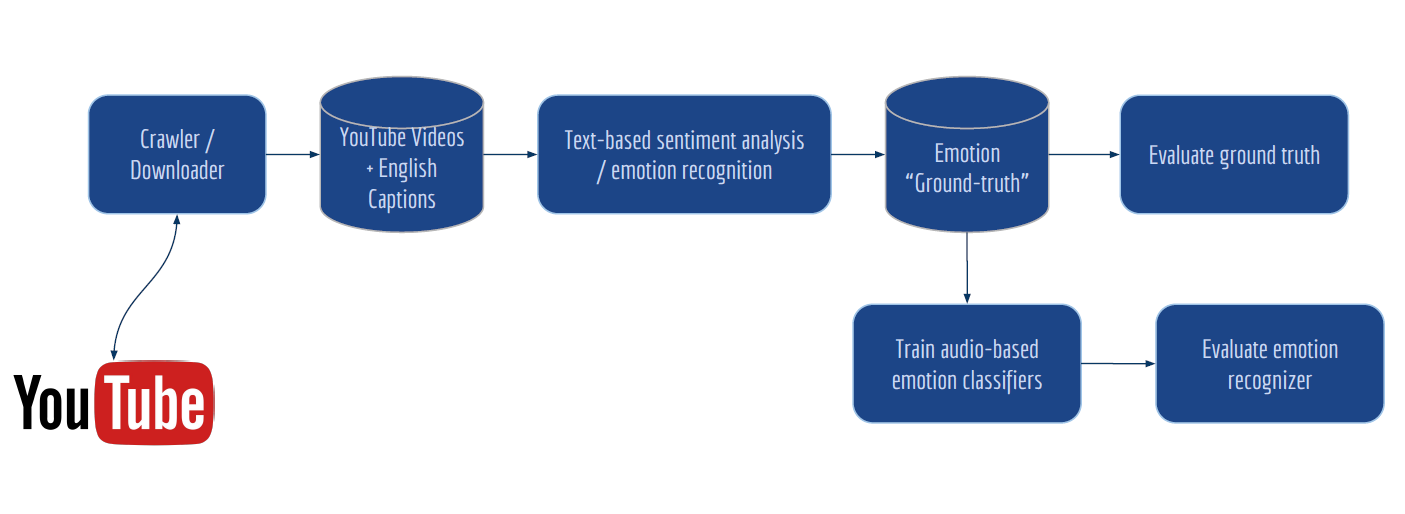
**Multimodal Information Processing and Analysis**

P3 Large – scale cross – modal speech emotion recognition



Athens 2019

Alevizopoulou Sofia 2022201704002

Avgeros Giannis 2022201704003

Tsiatsios George 2022201704024

Contents

[Contents 1](#_Toc532042332)

[Introduction 2](#_Toc532042333)

[Create dataset 2](#_Toc532042334)

[Source code 3](#_Toc532042335)

[Folder structure 3](#_Toc532042336)

[Crawl Youtube videos with English captions are available 3](#_Toc532042337)

[Sentiment classification model, recognize negative, neutral and positive sentiment 4](#_Toc532042338)

[Srt caption file 4](#_Toc532042339)

[Data Preprocessing 4](#_Toc532042340)

[Sentiment classifier 4](#_Toc532042341)

[Vader 4](#_Toc532042342)

[Text blob 5](#_Toc532042343)

[Pattern.en 5](#_Toc532042344)

[Final sentiment decision 5](#_Toc532042345)

[Train an audio-based emotion recognizer 6](#_Toc532042346)

[Evaluate the ground-truth 7](#_Toc532042347)

[Evaluate the performance of the audio-based emotion recognizer 7](#_Toc532042348)

[Repository of the project and management tool 7](#_Toc532042349)

[Repository 7](#_Toc532042350)

[Project management tool 7](#_Toc532042351)

[References 7](#_Toc532042352)

# Introduction

At this assignment we will create a model that used for modal speech emotion recognition on YouTube interviews. As first step we crawled videos that contain English dialogs from YouTube. After that, 3 sentiments classifiers used to recognize negative, neutral and positive sentiments. The most confident of these decisions (negative, neutral and positive sentiments) used to train an audio-based emotion recognizer. The final step, was the evaluation of ground truth. A subset of the audio classifier outcomes, specifically 100 samples per each class negative, neutral and positive were evaluated by listening them and put a sentiment tag. After these 2 different tags are compared.

# Create dataset

The dataset created by us, choosing 100 random interviews from YouTube. A csv file “dataset.csv” has been created and contains information about these videos.

The format of csv file is:

*Id,URL,Title,Audio,Captions,Polarity,Pickle*

For example: 1,https://www.youtube.com/watch?v=zkpelP3x0mw,Westley Allan Dodd - Last Interview before Execution,audio/1.wav,subtitles/1.srt,polarity\_csv/polarity\_1.csv,

pickle\_lists/polarity\_1.p

|  |  |
| --- | --- |
| Id | 1 |
| *URL* | *https://www.youtube.com/watch?v=zkpelP3x0mw* |
| *Title* | *Westley Allan Dodd - Last Interview before Execution* |
| *Audio* | *audio/1.wav* |
| *Captions* | *subtitles/1.srt* |
| *Polarity* | *polarity\_csv/polarity\_1.csv* |
| *Pickle* | *pickle\_lists/polarity\_1.p* |

More specifically, each video has a unique id. All the related files with that video will be saved with the name of videoId in the specific subfolders. At each entry of csv file is also defined the path of all related files for this video: path where audio file located, path for the subtitles, path for the results of captions’ sentiment analysis.

# Source code

There are 3 scripts:

*Downloader.py* 🡪download the captions and audio file for a video, convert audio to mp3 and captions from .vtt to .srt format

*Parser\_caption.py* 🡪sentiment analysis on segments based on captions context. Hold just the relative segments.

*Parser\_audio.py* 🡪Train an audio svm classifier with the outcomes of captions’ sentiment analysis. Split the audio in the related segments and create folders with positive, negative and neutral segments.

# Folder structure

Running the above scripts several folders will be created. Specifically, the folder structure of the project is:

There will be a folder “audio” that will contain all the mp3/wav files which downloaded from *Downloader.py*. Inside audio folder will also be created 3 subfolders, negative, positive, neutral that will contain audio segments for each sentiment sentiments. Also, there will be a folder “subtitles” at which will be contained all srt and vtt files (files with captions from the videos). A folder “polarity\_csv” will be created that will contain a csv file for each video. Each csv file will have as name the videoId and will contain the segments intervals and the result of the speech emotional sentiment classifier. This info will be also saved as pickle file at the subfolder “pickle\_lists”. These files used form the audio classifier.

# Crawl Youtube videos with English captions are available

For crawling YouTube videos, youtube-dl has been used. A python script, Downloader.py has written for that scope. On this script the captions and the audio fro all videos of the dataset are downloaded. Captions firstly downloaded as .vtt format and after converted to .srt format. Similarly, audio is first downloaded as mp3 and after is converted to wav with FFmpeg.

# Sentiment classification model, recognize negative, neutral and positive sentiment

## Srt caption file

The next step is to classify the captions of each video as negative, positive or neutral based on the text. The .srt file of each video has the below format. Firstly, is defined the segment’s duration and after the text.

*00:00:05,900 --> 00:00:07,999*

*This is a subtitle for the course multimodal Analysis*

*00:00:10,000 --> 00:00:14,000*

*This is another subtitle for the course multimodal Analysis*

*00:00:15,000 --> 00:00:15,100*

*[Laughing]*

*00:00:16,000 --> 00:00:17,000*

*[Laughing] And guess, this is another subtitle*

## Data Preprocessing

Captions usually contains tags like: cheering, laughing etc. We have excluded the segments that has no actual text and contain only tags. Tags that are included in sentences are not excluded but are taken into account on the sentiment analysis. Also, we have excluded segments whose duration is less than 2 secs. As our task it to recognize the sentiment of an audio, very small segments have no valuable information.

## Sentiment classifier

Since we have cleaned the data, the next step is to use some sentiment classifiers for classifying the captions into 3 sentiments (positive, negative, neutral).

Vader lexicon, text blob and pattern.en module have been used for that.

### Vader

VADER produces four sentiment metrics. The first three, positive, neutral and negative, represent the proportion of the text that falls into those categories. The final metric, the compound score, is the sum of all of the lexicon ratings which have been standardised to range between -1 and 1 and shows how strong is the recognized sentiment. Positive score means positive sentiment, negative means negative sentiment and zero means neutral sentiment.

### Text blob

The [sentiment](https://textblob.readthedocs.io/en/dev/api_reference.html#textblob.blob.TextBlob.sentiment) property of text blob returns a tuple of (polarity, subjectivity). The polarity score is a float within the range [-1.0, 1.0]. The subjectivity is a float within the range [0.0, 1.0] where 0.0 is very objective and 1.0 is very subjective. Positive polarity means positive sentiment, negative means negative sentiment and zero means neutral sentiment.

### Pattern.en

The pattern.en module contains a fast part-of-speech tagger for English sentiment analysis. Positive result means positive sentiment, negative means negative sentiment and zero means neutral sentiment.

### Final sentiment decision

We choose to create a model with high precision, for that reason we took in mind only the segments that we were “sure” about their sentiment (positive, negative, neutral). We have excluded segments at which the above sentiment classifiers had different results. After, as final score of sentiment we decided to use the average outcome of the 3 classifiers for each segment. Only these segments will be used at the next step of audio analysis, segment with different polarities based on 3 sentiment classifiers excluded.

The outcome of each video will be saved at a csv file located at /polarity\_csv/ with the name polarity\_videoId.csv. At this file will be written the several segments duration [strat-end] followed by the polarity score. This information will also be saved as a pickle file at the path /pickle\_lists/ with the name polarity\_videoId.p. These files will be used from the audio classifier at the next step.

For example, the context of the file polarity\_1.csv:

*"00:00:51,319",0.10695*

*"00:00:58,930",0.67985*

*"00:01:04,640",0.43745*

*"00:01:17,030",-0.1839838383838384*

*"00:01:20,690",0.0*

*"00:01:28,700",-0.446825*

*"00:01:32,240",0.0*

*"00:01:36,040",0.0*

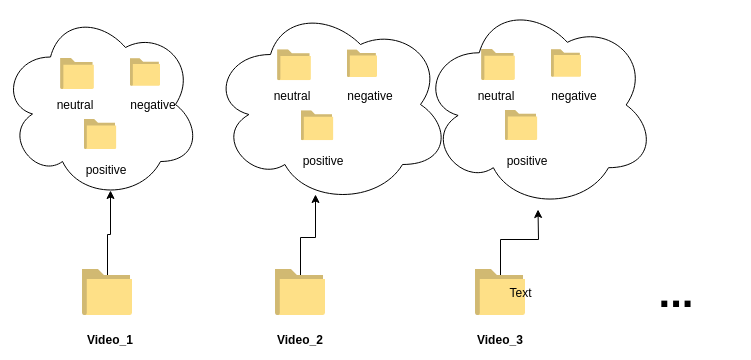
*"00:01:46,700",0.0*

So, from the video 1.srt that initially has 100 segments we hold just the 9 segments that are written above. The other 81 segments either their duration was less than 2 secs or the 3 sentiment classifiers didn’t have the same outcome. For these 9 segments, 3 are positive, 2 negative and 4 neutral.

# Train an audio-based emotion recognizer

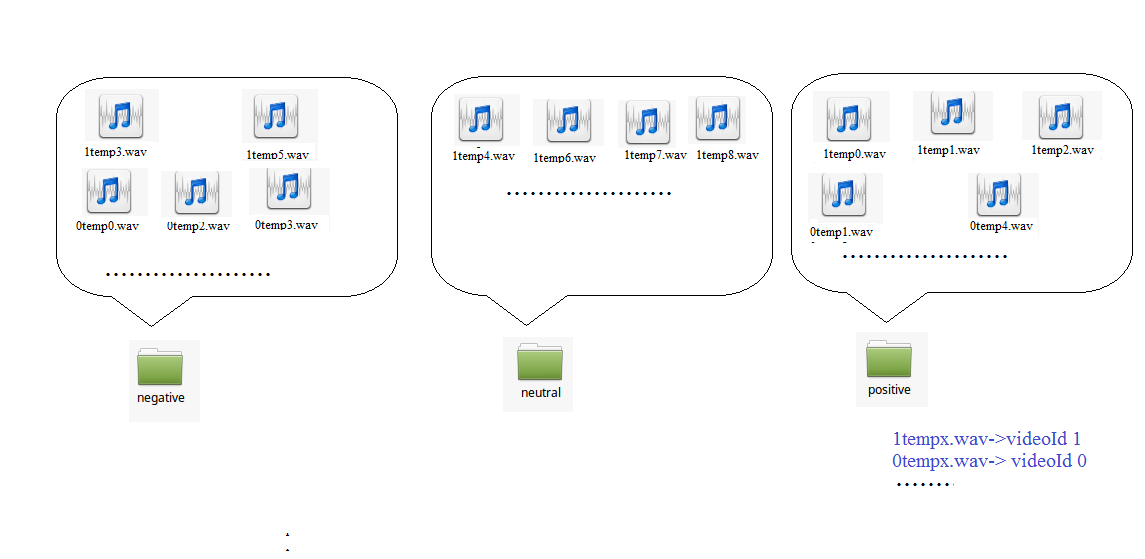
The next step is to train an audio-based emotion recognizer. For that scope we have used the pickle files created at the previous step. As said before, there is one pickle file for each video. Parsing each pickle file, we can take information about which are the segments of the video that we are interested in.

Each wav file is splitted in the segments that are referred at the pickle file of this video. Only the segments that are referred at the pickle file will be used. The name of th segment will be a substring of videoID and an increased integer eg: 0temp1.wav. After, a folder per VideoID is created. Each VideoID folder will contain the corresponding wavs per sentiment separated in folders (Positive, Neutral , Negative) this approach will help us during the K-Fold validation or Leave one out.



For each selected video ID, the segments based on the label are copied in the corresponding folder in the Train or Test folder as below. The modifications we did associated with the way of train and test dataset split. We would like to split the dataset based on videoId and not randomly on segments. We used the name of each wav file to do it, as all the segments that come from a specific video its name is a substring of the video Id. Generally, All the segments from one video should be totally on the train or on the test dataset.

The subfolder of positive, sensitive, neutral look like the below graph:



For each wav we extract the features. In order to deal with unbalance dataset that can occur due to the nature of the task, SMOTE library is used. SMOTE will balance the dataset for training and generate equal distribution of all classes.

Before Training the dataset will be also normalized to avoid any outliers.

After, we seek the best hyper parameter for the SVM classifier (other classifier can also be used) based on the best F1 score.

Finally, with the best parameter the model will be fitted in whole train dataset , which will be firstly normalized. The model will be saved as pickle file for future train or test purposes.

# Evaluate the ground-truth

# Evaluate the performance of the audio-based emotion recognizer

# Repository of the project and management tool

## Repository

The source code of the project is here: <https://github.com/salevizo/multimodal_audio.git>

## Project management tool

As project management tool we used Trello. The url for our board is:

https://trello.com/b/dTThH7Ep/project3

# References

[1] https://github.com/tyiannak/pyAudioAnalysis