Speech Emotion Recognition

Amri Myftija
Politecnico di Torino
Student id: s281585
amri.myftija@studenti.polito.it

Faezeh Kazemihatami
Politecnico di Torino
Student Id: s289238
faezeh.kazemihatami@studenti.polito.it

***Abstract*—We present in this paper a possible method to address the problem of speech emotion recognition for short snippets of audio. We map audio to a spectrogram feature space and and perform classification via SVM. We show that the proposed approach gives us satisfying results for the SER task.**

I. Problem overview

The objective of this project is to perform Speech Emotion Recognition (SER) on short snippets of audio. SER is the task of recognizing the emotional aspects of speech irrespective of the semantic contents [1]. Seven possible recognizable emotions are assigned to an audio file: happiness, sadness, anger, fear, disgust, surprise and neutral. The dataset provided consists of two different parts:

- A development set of 9,597 audio files and their label
- An evaluation set of 3,202 unlabeled audios.

In Fig. 1 we can observe the label distribution of the development set. Its is unbalanced. The label "Surprised" in particular is significantly underrepresented. This could pose a problem for the classification task. Data are said to suffer the Class Imbalance Problem when the class distributions are highly imbalanced. In this context, many classification learning algorithms have low predictive accuracy for the infrequent class. [2]

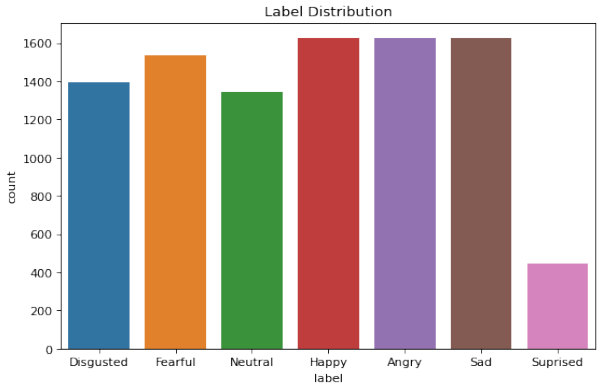


Fig. 1: Label distribution of the development set. The "Surprised" label is significantly underrepresented.

The audio files were sampled at a rate of 8 KHz and a using a 16 bit resolution. They have differing lengths as can be seen in Fig. 2.

Audio signals can be seen as a sum of sinusoidal waves. Any acoustic signal s(t) can be represented as a superposition of sinusoidal waves with different frequencies. Digitalized audio is obtained by sampling the analog audio in time and discretizing it's amplitude [3].

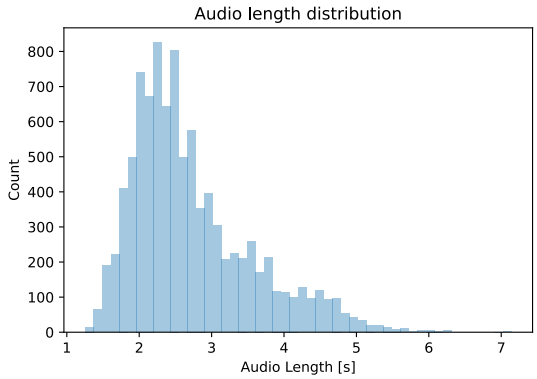


Fig. 2: Distribution of audio lengths.

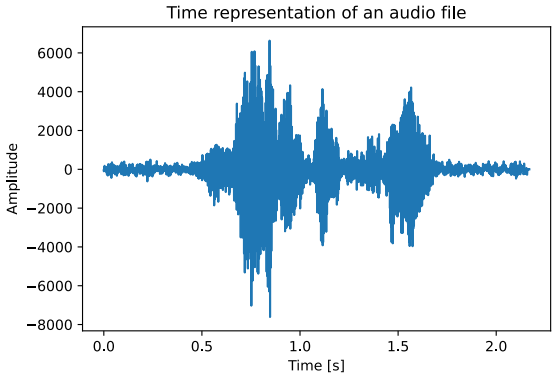
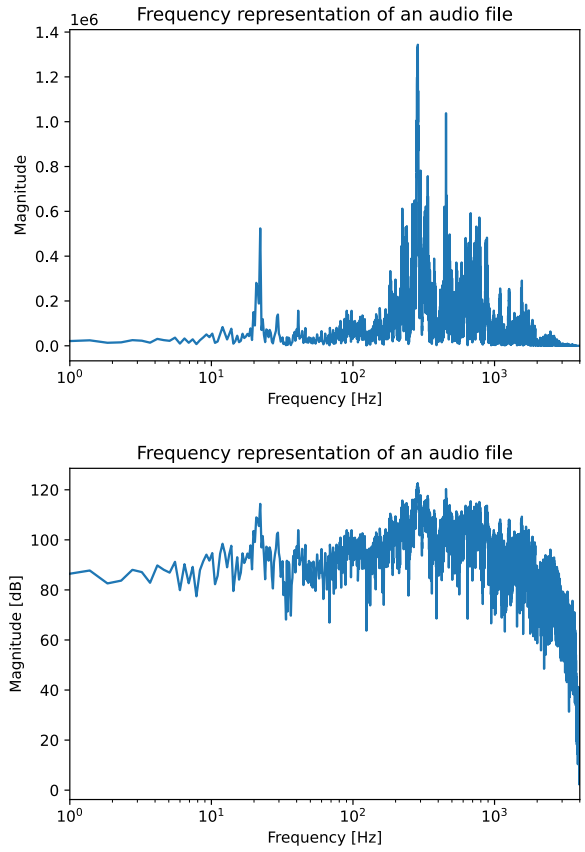


Fig. 3: Audio file "0.wav" representation in the time domain. We can observe the leading and the trailing silence by its low amplitude.

An example of digitalized audio can be seen in Fig. 3. Audio signals can be transformed through the Fourier transform. It allows us to view the different frequencies composing the signal. An example can be seen in Fig. 4.

Fig. 4: Audio file "0.wav" representation in the frequency domain through its magnitude spectrum. The frequency axis is shown in logscale as it better represents our perception of sounds. In Fig. 4b the amplitude is shown in dB. This scale is more representative than the linear one for the Human ears. [3]



II. Proposed approach

A. Data Exploration

We inspected the longer audio files manually. They contain sections of silence and longer utterances with respect to the mean. We kept them in our dataset.

B. Feature Extraction

The feature extraction follows these steps

- Obtain spectrogram $\mathbf{S_i}$ of each audio file $\mathbf{a_i}$. $\mathbf{S_i}$ is a (n_f, n_t) matrix
- Compress the spectrogram $\mathbf{S_i}$ of the audio to a fixed size. $\mathbf{S_i\_c}$ is a (n_f_bins, n_t_bins) matrix
- Build feature vector as by flattening $\mathbf{S_i\_c} \rightarrow \mathbf{x_i}$

C. Spectrogram Computing Details

The spectrogram is obtained by using a Short-time discrete fast Fourier Transform (SFT) on the on audio signal. The SFT parameters used are:

- time windows size: 256 [32 ms]

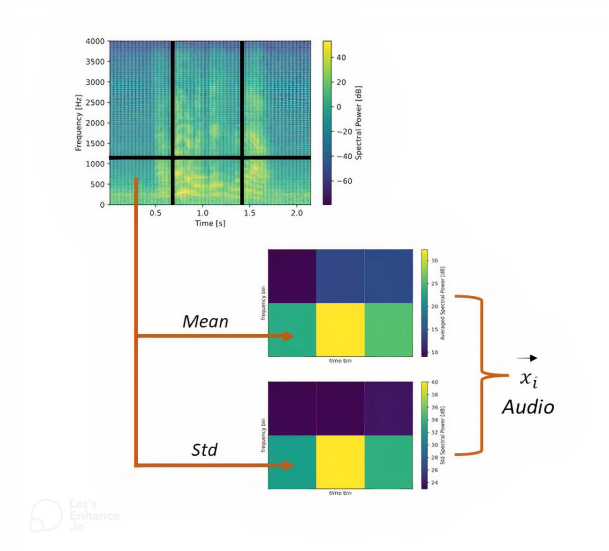


Fig. 5: Toy example of the feature extraction process. The audio's spectrogram is divided into a fixed number of rectangles. The frequency slices are equidistant on the mel scale. For each square, the mean and standard deviation is computed. These two matrices are computed, flattened and concatenated to produce the mapping of the audio in the feature space.

- window slide: 80 [10ms]

The pair (30ms, 10ms) has been used successfully for machine learning speech applications [3]. We use a 32 ms [256] window instead as it's faster to compute the SFT

The implementation of the DFFT will produce a $(129, f(audio\_length))$ matrix

D. Spectrogram Compression Details

The spectrogram is divided in equal time intervals but in unequal fequency intervals. This is motivated by the fact that human sensitivity to frequency is not linear. [3]. The frequency axis is split such that the divisions are equidistant on the mel scale. The mel scale is logarithmic scale which modeled on human frequence sensitivity [3]

An example of how it works is shown in Fig. 5.

E. Model selection

The following models are considered:

- Random Forest: A random forest is a predictor consisting of a collection of randomized base decision trees [4]. This is an algorithm which can perform well. It is fast to train as the decision trees can be trained in parallel. It is inherently multiclass.
- (Scaled) SVM is a classification technique which finds a separating hyperplane between the differently labelled classes [5]. It's a binary classification algorithm which can be extended to multi-label by building multiple classifiers. We use the default OVR setting for shorter training time. Training time is slower than random forest. However it has produced good results in audio classification in

TABLE I: Hyperparameter Configurations

Model	Parameter	Values
Preprocessing	n_f_bins	{2, 5, 15, 20}
	n_t_bins	$\{2, 5, 10, 15, 30\}$
Random Forest	$n_estimators$	$\{50, 100, 150\}$
	criterion	{gini, entropy}
	max_depth	{None, 10}
	class_weight	{None, balanced}
SVM (RBF)	C	{0.01, 0.1, 1, 10, 100}
	gamma	{"scale", 0.1, 1, 10, 100}
	class_weight	{None, balanced}

the past [1]. We use scaling because the algorithm uses euclidean distances. It is also faster to train the model when the features are scaled. We use z-score scaling as shown in Equation 1.

$$z = \frac{x - \mu}{\sigma} \tag{1}$$

F. Hyperparameters tuning

There are two sets of parameters to be tuned. One set is related to the feature extraction step. The other is related to the model tuning. For simplicity we assume that they are independent. Not doing so would significantly increase the search space and the time required to tune them. We will select the best feature extraction parameters using the default setting of our models.

The full range of the tested values is shown in Table 1.

III. Results

The best parameters for the feature extraction were the following. Using those features, the best scores (on the public leaderboard) for both models are shown in Table 3.

- $n_f\_bins = 20$
- $n_t\_bins = 10$

The best model is the SVM with a leaderboard score of 0.69.

TABLE II: Best Hyperparameter Configurations and Scores

Model	Hyperparameters	Score
RF	n_trees: 150, criterion: entropy, max_depth: None, class_weight: balanced	0.63
SVM	C: 10, gamma: scale , class_weight: None	0.69

IV. Discussion

The proposed model reached a satisfying result with respect to the baseline of 0.544. The approach could be improved by doing the following:

- Improve the preprocessing: The audio files have silences in them which may influence negatively in the mapping of the audios in the feature space. Approaches based on zero-crossing rate could be used to remove the long silences from the audios [3]
- More detailed grid search: With more time, we could run a more detailed grid search to better tune the hyperparameters

References

[1] B. C. Lech M, Stolar M and B. R, "Real-time speech emotion recognition using a pre-trained image classification network: Effects of bandwidth reduction and companding.," 2020.

[2] T. R. Shultz, S. E. Fahlman, S. Craw, and etal, "Class imbalance problem," in *Encyclopedia of Machine Learning*, pp. 171–171, Springer US, 2011.

[3] F. Camastra and A. Vinciarelli, *Machine Learning for Audio, Image and Video Analysis*. Springer.

[4] G. Biau, "Analysis of a random forests model," 2010.

[5] K. P. Bennett and C. Campbell, "Support vector machines," *ACM SIGKDD Explorations Newsletter*, vol. 2, pp. 1–13, Dec. 2000.