Audio Emotion Classification

Summer 2022 project of Data Science Lab

Lorenzo Scarciglia
Politecnico di Torino
Turin, Italy
lorenzo.scarciglia@studenti.polito.it

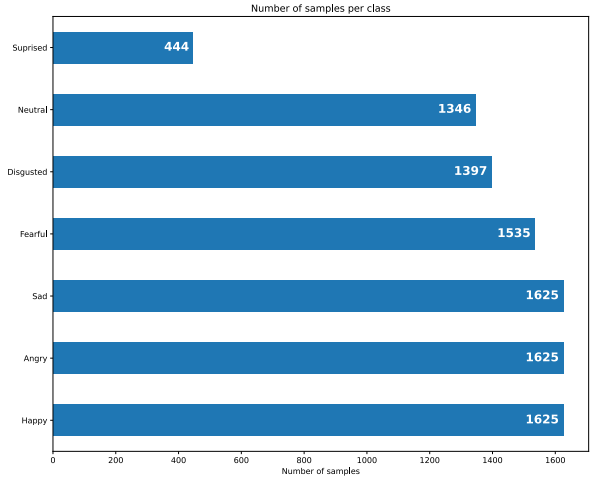

Fig. 1. Number of samples per class.

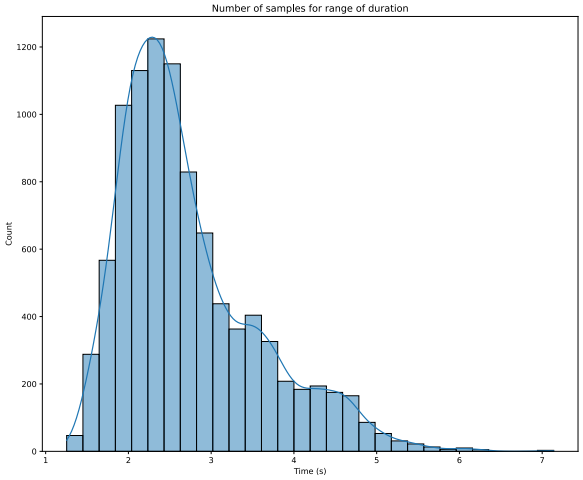

Fig. 2. Distribution of time lengths

***Abstract*—This report has the aim to present possible approaches to *audio emotion classification*.**
Such approaches involve extraction of audio features from time domain and time-frequency domain. Three classifiers were used and all of them outperformed a naive baseline defined for the classification problem.

***Index Terms*—Audio, MFCC, RMS, ZCR, SVM, Random Forests, PCA, SMOTE**

I. Problem overview

The competition concern a classification problem on a *audio dataset*. It is a collection of audios containing sentences spoken by different people in a particular mood. The goal is to classify, in terms of F1 macro, the beneath emotion between the following:

- *Surprised*
- *Neutral*
- *Disgusted*
- *Fearful*
- *Sad*
- *Angry*
- *Happy*

The dataset is composed by two parts:

- *development set*, which contains 9597 labeled samples as shown in Fig. 1
- *evaluation set*, which is made of 3201 unlabeled samples.

We will use the development set to build our classification model to correctly label the evaluation set.
By investigating the development set, we can make some considerations. First of all, the problem is not well-balanced as shown in Fig. 1. Hence, we have to take care of it. Second, all recordings have been sampled at a frequency of 8kHz with sample width of 16 bits. This is a sufficient sampling frequency that satisfy the Nyquist-Shannon sampling theorem.

If we inspect the time length of recordings (see Fig.2), we can see that there are some instances that are far from the mean value (around 2.3 seconds). Most of them belongs to the classes *sad* and *happy*. By manual inspection, we don't have any silence.
Audio recordings can be visualized in time domain as well as frequency domain. Fig. 3 and Fig. 4 are examples of such representations. Both kind of representations carry useful information, we thus will exploit both of them when we will extract features.

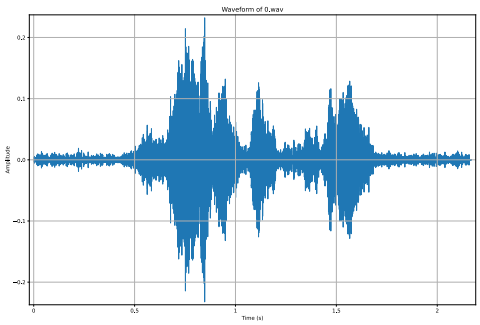

Fig. 3. Time representation.

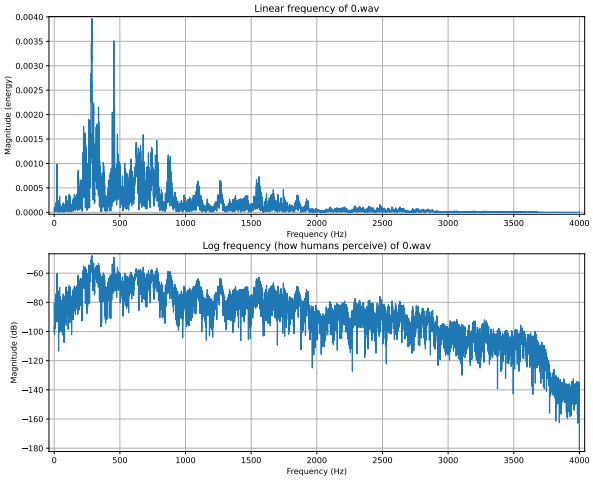

Fig. 4. Linear (up) and log (down) frequency representation. The logarithmic scale is preferred since humans perceive sounds in a logarithmic scale [1]

II. Proposed approach

A. Preprocessing

To deal with audio signals, we used `librosa` library. We red all signals and checked whether if there are silent signals and removed them. We discovered that 1 sample belonging to the class *sad* was just silence. Once we have done that, we encoded each *emotion* to a numerical value. (i.e. 'Disgusted': 0, 'Fearful': 1, 'Neutral': 2, 'Happy': 3, 'Angry': 4, 'Sad': 5, 'Suprised': 6).

We then extracted a bunch of features coming from time-frequency domain and time domain:

1) Time-Frequency domain features:

- Mel-Frequency Cepstral Coefficients (MFCCs for short): these coefficients the ones that make up the mel-frequency cepstrum[1]. Such coefficients are extensively used in many audio classification problems. Each coefficient is computed for each frame. The MFCC feature vector describes the power spectral envelope of a single frame.
- First and second order deltas of MFCCs[2]: Since speech signals are time-variant signals and in a constant flux, we need some coefficients that describe how MFCCs change through time. Such coefficients are the deltas of first and second order. A trivial interpretation of such coefficients is that they approximate first and second derivatives of the signal.
- Chromagram and its argmax and argmin: chroma is related to the twelve pitch classes. It is composed by twelve element (one for each pitch class) for each frame. It's used in music classification. Since the pitch it's the human perception of a sound wave, we thought that some kind of emotions could have different values of pitch class (typically anger has a higher mean value of pitch, others lower or something like that.) On the same thread we took the argmax and the argmin, to extract (we think) most relevant pitch classes.

2) Time domain features:

- Duration: it's the well known time length of a recording expressed in seconds.
- Zero Crossing Rate (ZCR): it's the number of times a signal crosses the zero line. For a frame t, we have:

$$ZCR_t = \frac{1}{2} \sum_{k=t\cdot K}^{(t+1)\cdot K-1} |sign(s(k)) - sign(s(k+1))|$$

where K is the frame size and $s(k)$ is the value of the k-th sample.

- Root Mean Square Energy: it's the mean energy of a frame. We sum all the amplitudes of a frame (squared), we divide by the frame size and then we take the root of such result.
  RMS of a frame t:

$$RMS_t = \sqrt{\frac{1}{K} \sum_{k=t\cdot K}^{(t+1)\cdot K-1} s(k)^2}$$

where $s(k)^2$ is the energy of the k-th sample (it's the value of the sample squared), K is the number of samples in a frame (frame size).
In our study, we took the average of all the frames contained in the audio. In such a manner we end up with a single value.

All the statistics where computed over the time axis to obtain the same number of features from each recording, even if the differs in time length.

The workflow of data preprocessing is shown in Fig. 5.

Once we have extracted such features (around 200 features for each sample), we now have to take care of the imbalanced class problem. We could use a classifier that allows to set a

[1]A cepstrum is the IFT (inverse Fourier transform) of the logarithm of the spectrum

[2]differential and acceleration coefficients respectively

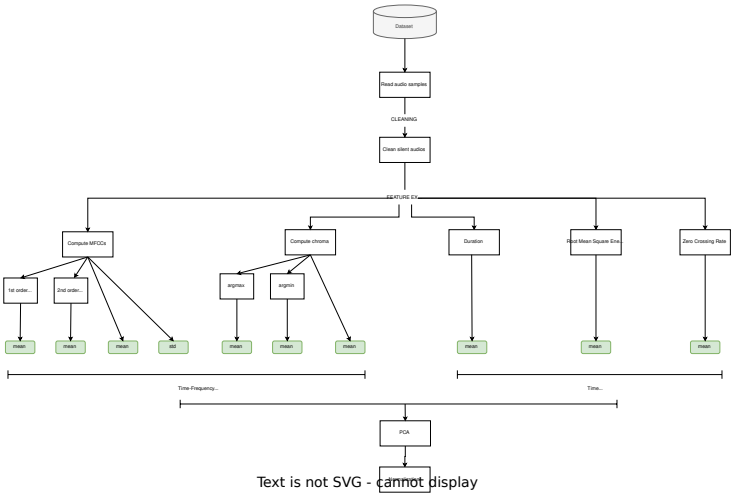

Fig. 5. Preprocessing workflow.

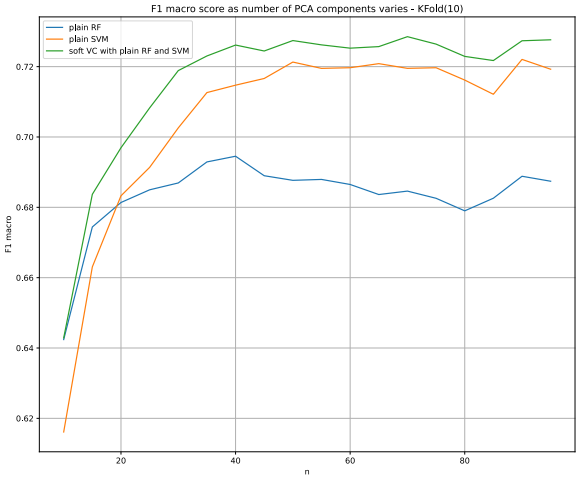

Fig. 6. Tuning of PCA as n varies.

`class_weight` parameter (like SVM), but since we chose to use Random Forest, SVM and a voting classifier using both, we have chosen the oversampling approach.
We used SMOTE [2]. It selects an instance of the imbalanced class, choose one of its k nearest neighbors and create a synthetic instance made of the convex combination of the 2 instances.
In such a manner, it is more general than just count twice or more one instance of the low-quantity class. Because such oversampling technique doesn't give any new information about mean and standard deviation.

We have a lot of features and not all of them are important. We applied PCA [3] to select most relevant ones.

Finally, we applied Min-Max normalization to our features. We did that even if Random Forests didn't need a normalization step. We did it to save time since we wanted to use a voting classifier using random forest and SVM. In such a manner, we ended up with a single kind of dataset. Hence, we applied normalization to all classifiers.

B. Model selection

In audio classification, many classifiers have proven to work well. To name a few of them, we have Gaussian Mixture Models (GMM), ANN, SVM and Random Forests[[4], [5], [7], [8]]. We have chosen to use SVM and Random Forests. In addition, we tried to exploit both of them in a voting classifier method. We propose the latter approach to exploit both RF and SVM, unfortunately results were not higher than SVM.

- *Random Forest (RF)*: ensemble method that uses a given number of *decision trees* each of which is trained on a random subset of features. Moreover, each tree is trained on a random subsample (with replacement) of the initial dataset.
  Such conditions allow RF to be more robust to overfitting (w.r.t. decision trees). Moreover, the parameters to tune are the ones concerning decision trees. RF works on one feature at a time, hence it's not mandatory to normalize data.
- *Support Vector Machine (SVM)*: this technique needs normalized data. It apply transformations (linear or not) to data and aims to find the maximum-margin hyperplane[3] that separates classes. It has been proven to work well in audio classification.
- *Voting Classifier*[4]: This method fit a given number of unfitted estimators and use them to classify an instance. The choice of such classifier can be weighted. It uses soft voting or majority voting.

C. Hyperparameters tuning

To tune hyperparameters, experiments were conducted on development dataset divided into: train (60%), validation (20%) and, test (20%) set. Classifiers were trained using K-fold cross-validation with $K = 10$[5].

We trained 3 different classifiers. To select the best number of features we ran PCA as the varying number of components as shown in Fig. 6 and took the best n for each kind of classifiers. Results are the following:

Classifier	**#features**
Random Forest	40
SVM	90
Voting Classifier	70

Once we have find those numbers, we performed the tuning of the classifiers (with the respective number of features chosen to use in PCA) as shown in Fig.7. All classifiers were trained using 10-fold cross-validation. The best one on

[3]supported by some training points, called support vector
[4]scikit-learn implementation available here.
[5]Also the best number of features in PCA was chosen running cross-validation

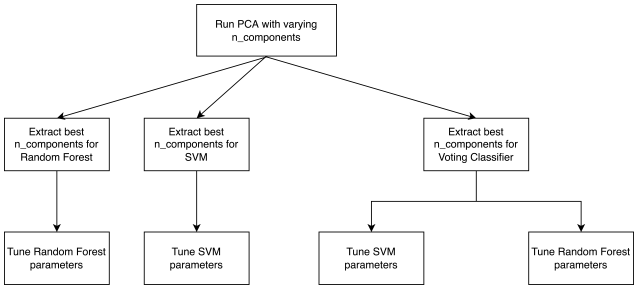

Fig. 7. Hyperparameters tuning workflow

validation set was selected, trained on the overall development set. Then, it was used on the evaluation set to see the results.

1) Random Forest tuning:

Classifier	Parameters
RF	n_estimators: 150, 300 criterion: gini, entropy max_depth: 20, 30, None

Best configuration with 0.6302:

- criterion: 'gini',
- max_depth: None,
- n_estimators: 300

2) SVM tuning:

Classifier	Parameters
SVM	C: 1, 3, 5, 7, 10, 15, 20, 30, 50, 100, 500

Best configuration with 0.7425: C = 3.

3) VC tuning:

Classifier	Parameters
RF	n_estimators: 150, 300 criterion: gini, entropy max_depth: 20, 30, None
SVM	C: 1, 3, 5, 7, 10, 15, 20, 30, 50, 100, 500
VC	voting: hard, soft

Best configuration with 0.7305 (voting: soft):

- RF with 0.6961 :
  - criterion: 'entropy',
  - max_depth: 20,
  - n_estimators: 300,
- SVM with 0.7417: C = 5

III. Results

Each of this classifier was trained on the overall development dataset and tested on the evaluation set. We obtained the following results:

Classifier	F1 score
RF	0.668
SVM	**0.707**
VC	0.701

As we can see, the most performing between the three is SVM. It's worth noting that every proposed classifier outperform the naive baseline of 0.544.

IV. Discussion

We have shown 3 different classifiers that outperform the given baseline. The most performing one is SVM. We exploited both time domain and time-frequency domain features. Many experiments were conducted, in particular we tried to implement the same approach without cross-validation since the n for PCA tooks around 2 hours to complete. Instead, without cross-validation, it tooks only 20 minutes. Albeit the overall lower time, resulting classifiers overfitted (scores on the validation set were higher) and didn't ended up as the ones found with cross-validation.
We found:

- 0.666 for RF,
- 0.695 for SVM and VC

We can appreciate the difference in SVM.

There is room for improvement in the model. First of all, a better tuning of hyperparameters and the tuning of other hyperparameters. Moreover, we could preprocess data in a different way. We could apply accurate denoising technique[6]. for example, we could use different features coming from the time-domain frequency like the spectral centroid and the spectral flux, or LPCC coefficients are another possibility [6], [7]. Unfortunately, due to the lack of domain knowledge and the short amount of time, it's possible that relevant features were not considered. Even if we don't consider other features, we could use other approaches like GMM and ANN that works really well. Another possibility is to use ANN in combination with audio features extracted with VGGish [9], such approach avoid us to extract features manually. Another possible approach is to augment data.
There are too many feasible scenario that couldn't be faced due to time constraints. However, our basic approach performed quite well and obtained satisfactory results in terms of F1 macro.

References

[1] E. B. Goldstein, Sensation and perception (3rd ed.). Wadsworth/Thomson Learning, 1989.
[2] N. V. Chawla, K. W. Bowyer, L. O. Hall, W. P. Kegelmeyer, SMOTE: Synthetic Minority Over-sampling Technique, Journal Of Artificial Intelligence Research, Volume 16, pages 321-357, 2002
[3] Andrzej Maćkiewicz, Waldemar Ratajczak, Principal components analysis (PCA), Computers Geosciences, Volume 19, Issue 3, 1993, Pages 303-342
[4] M. E. Ayadi, M. S. Kamel, F. Karray, "Survey on Speech Emotion Recognition: Features, Classification Schemes, and Databases", Pattern Recognition 44, PP.572-587, 2011.
[5] S. Emerich, E. Lupu, A. Apatean, "Emotions Recognitions by Speech and Facial Expressions Analysis", 17th European Signal Processing Conference, 2009.
[6] A. Nogueiras, A. Moreno, A. Bonafonte, Jose B. Marino, "Speech Emotion Recognition Using Hidden Markov Model", Eurospeech, 2001.

[6]A denoising approach was conducted but results were worse, maybe due to a superficial study of denoising.

[7] P.Shen, Z. Changjun, X. Chen, “Automatic Speech Emotion Recognition Using Support Vector Machine”, International Conference On Electronic And Mechanical Engineering And Information Technology, 2011.
[8] Ashish B. Ingale, D. S. Chaudhari, Speech Emotion Recognition, International Journal of Soft Computing and Engineering (IJSCE) ISSN: 2231-2307, Volume-2, Issue-1, March 2012
[9] Hershey et al., CNN architectures for large-scale audio classification, 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)