Audio Intent Classification using MFCCs and Delta Features

Giorgia delle Grazie, Elisa Salvadori
Politecnico di Torino
Student id: s300879, s302630
s300879@studenti.polito.it, s302630@studenti.polito.it

***Abstract*—In this report we introduce a possible approach for the Audio Intent classification problem. In particular, it consists in extracting the Mel Frequency Cepstral Coefficients and the Delta features of each audio signal and testing different classification algorithms. The proposed approach has been evaluated in terms of accuracy.**

I. Problem overview

The competition is a classification problem on a collection of audio recordings, our goal is to predict the intention of the audio i.e. the action requested and the object that is affected by the action.

The provided dataset consists of two parts:

- a development set, containing 9854 recordings for which the label, i.e. the action and the object, is known;
- an evaluation set, containig 1455 recordings for which the label is unknown.

Each instance of the dataset is characterized by:

- *path*: the path of the audio file;
- *speakerId*: the Id of the speaker;
- *action*: the type of action required through the intent;
- *object*: the device involved by intent;
- *self-reported fluency level*: the speaking fluency of the speaker;
- *first Language spoken*: the first language spoken by the speaker;
- *current language used for work/school*: the main language spoken by the speaker during daily activities;
- *gender*: the gender of the speaker;
- *ageRange*: the age range of the speaker.

The attributes are the same for the Evaluation and the Development set, except for *action* and *object* which are not given for the first one.

In order to obtain the intent of the audio, a new attribute is added, replacing *action* and *object* by the string concatenation of them. The *Intent* is the target variable (and it is the unknown in the evaluation set).

II. Proposed approach

For speech recognition it is common to use spectral features. For our classification problem we combine speaker-dependent features with speaker-independent ones. Generally speaking we compute the Mel Frequency Cepstral Coefficients (MFCCs) of the audios for the first class of features and for the second one we take the first and the second order derivatives of the MFCCs [1].

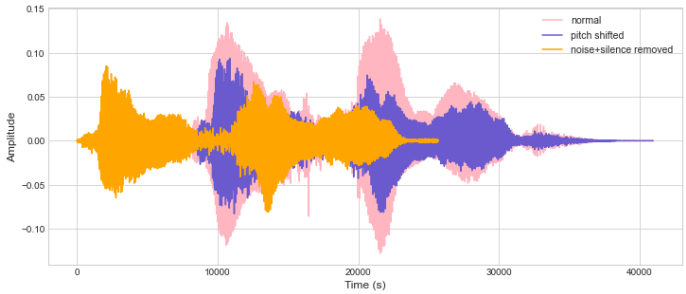

Fig. 1. Comparison of an audio of a female speaker before and after preprocessing technique.

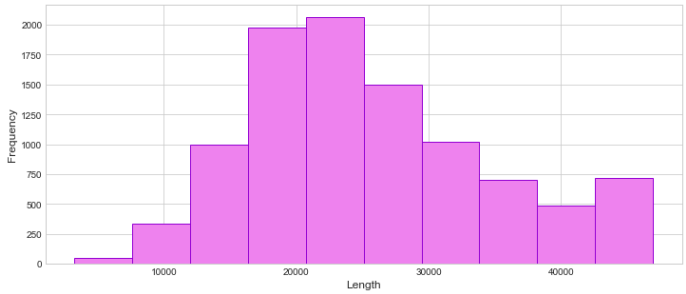

Fig. 2. Distribution of lengths in the Development set.

A. Preprocessing

First of all we want to obtain a robust method which is speaker-independent. In order to get it we decide to discard every attribute except for *path*, and the target variable for the Development set. We know that also the features extracted from the audio signal could be speaker dependent but we discuss it later on. The following steps are taken:

- Resampling: The sample rates of the dataset are not the same for each record, in particular there are 9554 records with a sampling frequency of 16000 Hz and 300 with a sample rate of 22050 Hz, we resample each record with the bigger of the two.
- Pitch shifting: As it is discussed in [2] there is a difference between the average fundamental frequency ($F0$) of men

and women, in fact *F0* is commonly used in gender identification problems. We try to overcome this speaker dependency by shifting the pitch of one semitone (-1) for audios of females (using **librosa.effetct.pitch_shift()** with $n\_steps = -1$.)

- Standardization: We compute the standardization of the recordings to get less sensitive to different recording conditions.
- Denoising: A filtering technique is applied to smooth the signal: the Savitzky-Golay filter is used. "The Savitzky-Golay filter can be used to reduce high frequency noise in a signal due to its smoothing properties and reduce low frequency signal (e.g., due to offsets and slopes) using differentiation." [3] (using **scipy.signal()** with $window\_length = 15$ and $polyorder = 5$).
- Removing silence: Some records contain silence at the beginning or at the end of them, so we remove the leading and trailing silence (using **librosa.effect.trim()** with $top\_db = 35$, $frame\_length = 256$ and $hop\_length = 64$).
- Trimming and Padding: The resulting signals have different lengths, we trim and zero-pad each signal up to a certain length (in particular up to the 95 percentile).

The Figure 1 shows an audio of a female speaker before the preprocessing, after computing the pitch shifting and after the denoising and silence removing phases. The Figure 2 shows the distribution of lengths in the Development set. Leveraging this information to get the 95 percentile we make an assumption about the Evaluation set: the distribution of the Evaluation set needs to be similar to the one of the Development.

Each audio signal is ready now for the features extraction process:

- MFCC: The MFCCs represent sound approximately as it is perceived by the human ear (the Mel scale is a logarithmic scale).
  We compute 13 MFCCs for each audio using the Librosa function **librosa.feature.mfcc()** with $n\_mfcc = 13$ and $n\_fft = 512$.
  We could summarize the process of the extraction in some steps as it is shown in the Figure 3. In particular, as it is said in [4], we have:
  - *Pre-emphasis*: This phase emphasizes higher frequencies using filters.
  - *Framing*: This phase divides the audio in overlapping frames of N samples.
  - *Windowing*: In this step an hamming window is applied.
  - *Fast Fourier Transform*: This phase converts each frame from time domain to frequency domain.
  - *Mel Filter Bank Processing*: This step performs a bank of Mel triangular filters.
    Given a linear frequency:

$$f_{Mel} = 2595 \times log_{10}\left(1 + \frac{f}{700}\right).$$

  - *Discrete Cosine Transform*: The result of this phase are the MFCCs, they are computed by converting, using the DCT, the log Mel spectrum into time domain.

  After the extraction the coefficients are normalized by the Mean and Variance Cepstral Normalization (CMVN).
- Delta and Delta-Delta MFCCs: These coefficients are dynamic features and are known as differential and acceleration coefficients [5].
  With these coefficients we keep in track the change in Cepstral features over time [4] (using **librosa.feature.delta()** with $order = 1$ and $order = 2$).

We keep all the MFCCs to exploit the temporal dynamic of signals.

The Figure 4 shows the plot of the MFCCs of an audio signal before and after the Mean and Variance Cepstral Normalization.

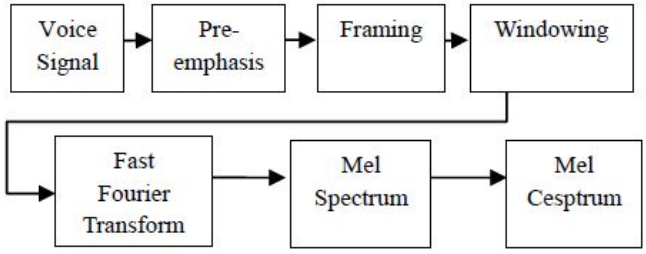

Fig. 3. Feature vector extraction steps [6].

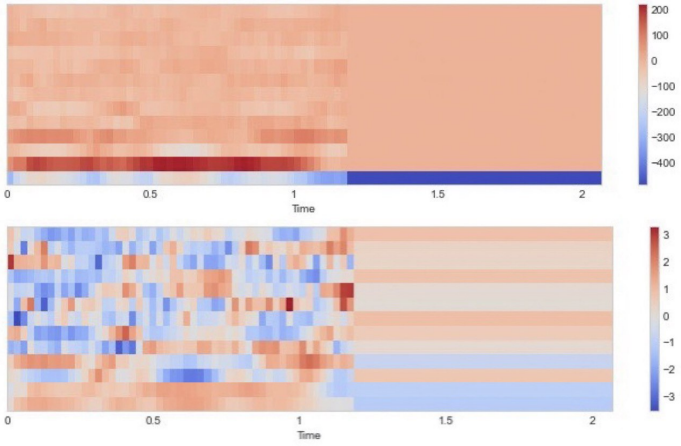

Fig. 4. MFCCs before and after the CMVM.

B. Model selection

The following algorithms have been tested:

- **Random Forest**: It is an ensemble learning technique. It uses N decision trees making their own prediction. It avoids overfitting by assigning the class label with majority voting.

- **K-Nearest Neighbor**: This algorithm makes classifications or predictions by using class labels of nearest neighbors (k closest points).
- **SVM**: It is an algorithm which separates two classes of data with an hyperplane or a decision boundary.

C. Hyperparameters tuning

For tuning our models and see which model performs better with our problem we decide to split the Development set as follows:

- Training set: 60% of the dataset, used to train model configurations;
- Validation set: 20% of the dataset, used to validate the configurations of hyperparameters;
- Test set: 20% of the dataset, used to asses the quality of the chosen model, after trainig the selected configuration with the union of the validation and trainig set.

Table I shows the different configurations of hyperparameters tested.

TABLE I
HYPERPARAMETERS CONFIGURATIONS

Model	Parameters	Values
Random Forest	n_estimators criterion max depth	{70, 100, 150} {gini, entropy} {None, 10, 20}
SVM	C kernel	{1, 5, 7, 10 } {rbf,linear, poly}
KNN	n_neighbors	{6, 10, 15}

III. RESULTS

The Table II shows the result of the best configuration of hyperparameters for each model trained with the union of validation and trainig set, we tested each configuration for feature vectors of $MFCC + \Delta$ and of $MFCC + \Delta + \Delta\Delta$ (the features are normalized for the SVM).

TABLE II
FINAL MODELS

Features	Model	Best Values	Accuracy
$MFCC + \Delta$	Random Forest	n_estimators: 100 criterion: entropy max depth: None	0.574
	SVM	C: 5 kernel: rbf	0.763
	KNN	n_neighbors: 6	0.682
$MFCC + \Delta + \Delta\Delta$	Random Forest	n_estimators: 100 criterion: entropy max depth: 20	0.579
	SVM	C : 7 kernel: rbf	0.744
	KNN	n_neighbors: 6	0.682

According to Table II the best model is the one with $MFCC + \Delta$ using SVM with $C = 5$ and $kernel = rbf$.

Now, comparing the results obtained with the different feature vectors, using the model and the best hyperparameters as discussed in the previous section, we could notice that:

- The accuracy of **Random Forest** is similar whether we consider $MFCC + \Delta$ or $MFCC + \Delta + \Delta\Delta$.
- The accuracy of **KNN** is the same for the two proposed vectors.
- For **SVM**, as previous said, the best accuracy is reached with $MFCC + \Delta$ (0.763). On the other hand also $MFCC + \Delta + \Delta\Delta$ reached quite satisfactory result.

For what it concerns the public score, we trained the model using all the Development set with SVM ($C = 5$ and $kernel = rbf$) and $MFCC + \Delta$ as feature vector, the result on the Leaderboard for the Evaluation set is 0.874 (the first baseline is 0.334).

IV. DISCUSSION

The proposed approach provides satisfactory results for all three model that we consider.

We have further considerations to make for improving our model:

- The proposed Development set is not well-balanced (see Figure 5). To deal with this kind of dataset we also could consider to oversample or undersample the dataset. Undersampling consists in decreasing the number of samples of the biggest class down to the size of the smallest one, while oversampling increases the size of the smallest one to the size of the the biggest one.
- Make a better hyperparameters research considering more parameters and configurations, because we only consider a small domain of possible hyperparameters.
- In this report we have discussed about our assumption for the length of the Evaluation set. To make the model more robust we could consider, instead of all the MFCCs, some statistical features like mean or standard deviation of each coefficient [7]. With this new approach there is no need of trimming and zero-padding but we could lose some information about the temporal dynamic.
- For reducing the number of features considered we could use PCA. To increase the accuracy the use of CNNN could be taken in consideration.

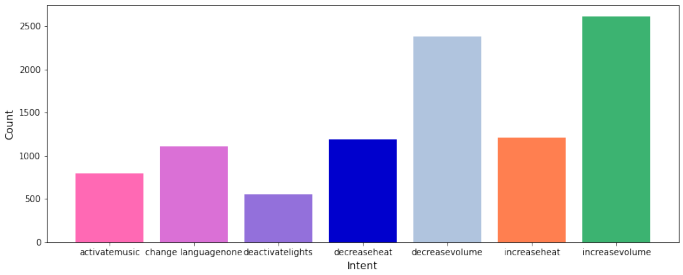

Fig. 5. Class-label counts

REFERENCES

[1] Z.-T. Liu, M. Wu, W.-H. Cao, J.-W. Mao, J.-P. Xu, and G.-Z. Tan, "Speech emotion recognition based on feature selection and extreme learning machine decision tree," *Neurocomputing*, vol. 273, pp. 271–280, 2018.

[2] S. Gaikwad and B. W. Gawali, “Gender identification using svm with combination of mfcc,” 2012.
[3] N. Gallagher, “Savitzky-golay smoothing and differentiation filter,” 01 2020.
[4] L. Muda, M. Begam, and I. Elamvazuthi, “Voice recognition algorithms using mel frequency cepstral coefficient (mfcc) and dynamic time warping (dtw) techniques,” *J Comput*, vol. 2, 03 2010.
[5] A. Bhavan, P. Chauhan, Hitkul, and R. R. Shah, “Bagged support vector machines for emotion recognition from speech,” *Knowledge-Based Systems*, vol. 184, p. 104886, 2019.
[6] S. C. Ileri, A. Karabina, and E. Kılıç, “Comparison of different normalization techniques on speakers’ gender detection,” 2018.
[7] H. Harvianto, L. Ashianti, J. Jupiter, and S. Junaedi, “Analysis and voice recognition in indonesian language using mfcc and svm method,” *ComTech: Computer, Mathematics and Engineering Applications*, vol. 7, p. 131, 06 2016.