Audio intent detection

Rostislav Timuta
Politecnico di Torino
Student id: s280238
s2802381@studenti.polito.it

***Abstract*—In this report we introduce a possible approach to an intent detection problem. Precisely, the method we propose extracts the MFCCs, the first and second order derivatives of the original MFCCs of each audio signal. These features are used as input for two classification models. The proposed method outperforms a naive baseline defined for the task and achieves overall acceptable results.**

I. Problem overview

Intent detection is a vital aspect in NLP and speech recognition as it detects the purpose or the intention behind a spoken or written statement. In audio intent detection, the system must accurately understand the user's spoken words and determine their intended meaning. In particular, given an input audio sample, the goal is to predict both the action and the object that is affected by the action. The dataset is divided into two parts:

- a *development* set, containing 9854 recordings.
- an *evaluation* set, consisting of 1455 recordings

The development set will serve as the basis for constructing a classification model that accurately categorizes the points in the evaluation set.

In addition, the development set offers an opportunity for us to make some observations. First, the problem is unbalanced:

Class	Count
Increase volume	2614
Decrease volume	2386
Increase heat	1209
Decrease heat	1189
Change language	1113
Activate music	791
Deactivate lights	552

TABLE I: Class distribution

Second, the sample width is of 2 bytes for all files. Third, all recordings have been sampled at a frequency of 22kHz. The Nyquist-Shannon sampling theorem states that in order to accurately represent a signal, the sample rate must be at least twice the highest frequency in the signal. Based on this theorem, 22kHz can be considered a reasonable sample rate for signals with frequencies up to 11kHz. [1]

Although all signals have been recorded using the same sample rate, the duration of each recording varies. The distribution of such durations is shown in Figure 1.

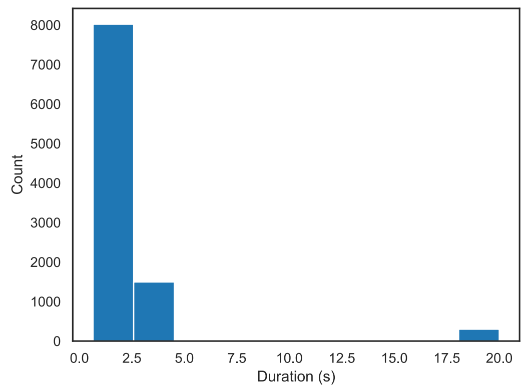

Fig. 1: Distribution of the durations of the recordings

There are some outliers that have a length that drastically deviates from the average duration (2.64 seconds). Manual inspection reveals that those signals contain silence right after the speech. Removing the silence from these recordings solves the problem. Since most classification models call for a specific number of features, we needed to come up with a method to extract the same number of features from each audio signal despite the different lengths of the entries.

We can visually examine various signals in the time and frequency domains to gain a better idea of the type of data at hand. Each signal is given useful and complimentary information by these two domains. Figure 2 shows one signal in the time domain. Figure 3 shows instead a trimmed version of the same signal. We can observe right away that sometimes there are pauses before and/or after the statement. In the trimmed version we remove both silence and what can be considered noise by accurately selecting the parameter that sets the threshold for trimming the audio signals.

II. Proposed approach

A. *Preprocessing*

Our method relies on the **MFCCs** (Mel-Frequency Cepstral Coefficients) **extraction** from each audio file, a widely used technique in speech and audio processing. [2]–[4]

The idea behind MFCCs is to capture the spectral envelope of a speech or audio signal in a compact and robust form. The process involves several steps:

- Mel-scale Spectrogram: The audio signal is transformed into a Mel-scale spectrogram, which emphasizes the frequency regions most relevant to human hearing.

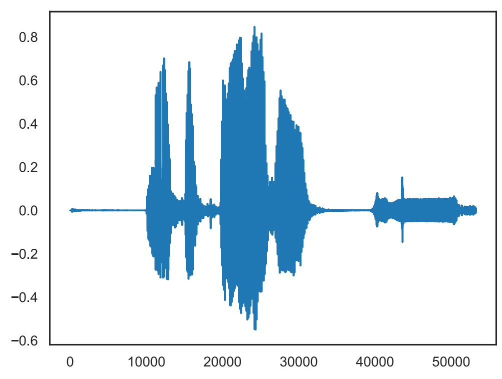

Fig. 2: Time domain signal rappresentation

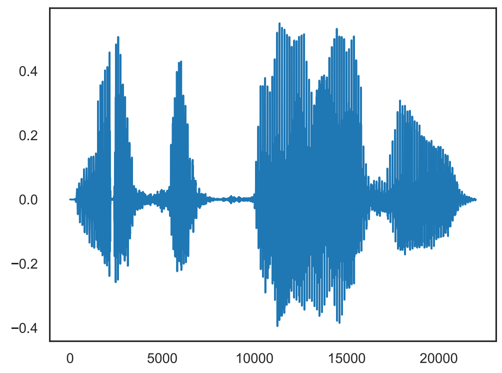

Fig. 3: Trimmed audio signal

- Logarithmic Compression: The Mel-scale spectrogram is logarithmically compressed to reduce the dynamic range of the data.
- Discrete Cosine Transform (DCT): A DCT is applied to the logarithmic spectrogram to produce a set of coefficients that represent the spectral envelope. These coefficients are called MFCCs.

We used MFCCs because they capture the relevant spectral information in a compact and robust form, while being relatively insensitive to noise and variability in the audio signal. In order to obtain a more comprehensive representation of the spectral and temporal information in the audio signal, we also make use of Delta MFCCs and Delta2 MFCCs which are variations of MFCCs that capture the temporal evolution of the spectral envelope over time.

- **Delta MFCCs** represent the first-order derivative of the MFCCs with respect to time, which capture the rate of change of the spectral envelope over time. In our case Delta MFCCs are used to capture the temporal dynamics of speech and audio signals, such as the rate of change of pitch and energy.
- **Delta2 MFCCs** represent the second-order derivative of the MFCCs with respect to time, which capture the acceleration of the spectral envelope over time. Delta2 MFCCs are used to capture additional temporal information, such as the rate of change of the rate of change of the spectral envelope over time.
  In Figure 4 we can have a representation of the MFCCs and the corrispondig deltas.

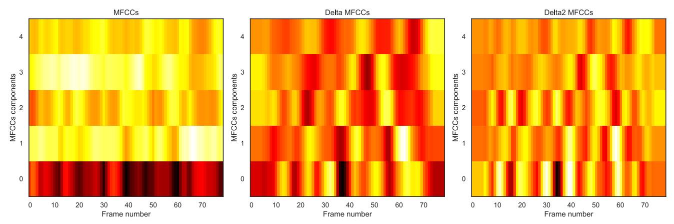

Fig. 4: Time domain signal rappresentation

To address the imbalance in our dataset we first experimented some common undersampling and oversampling techniques then we continued our study by applying several data augmentation technique in order to address this problem. Specifically, we randomly apply one of following techniques to each audio of the minority classes:

- *Speed Augmentation* involves changing the speed of the speech signal, which can help the model generalize to different speaking rates
- *Pitch Scaling* changes the pitch of the speech signal, which can help the model generalize to different speaking styles.
- *Noise Factor* involves adding noise to the speech signal, which can help the model generalize to noisy environments.
- *Random Gain* involves randomly adjusting the volume of the speech signal, which can help the model generalize to different volume levels.

Since our dataset is composed of intents spoken by different speakers with distinct accent, age, tone, speech rate, etc, applying augmentation techniques help to artificially increase the size of the training dataset and improve the robustness and generalization of the model.

B. Model selection

We tested two algorithms:

- *Random forest* is well-suited for speech intent recognition tasks due to its high accuracy and robustness. Random Forest is an ensemble method that combines multiple decision trees to make predictions. Each tree in the forest is trained on a different subset of the data and features, which helps to overcome overfitting issues and improves the robustness of the model. Additionally, Random Forest is relatively easy to implement and does not require extensive feature engineering or fine-tuning of hyperparameters. These characteristics make Random Forest a good choice for speech intent recognition problems where

accuracy is a top priority and the data is complex and diverse. [5], [6]
- *SVM* (Support Vector Machine) is another powerful ML algorithm that is well-suited for speech intent recognition problems. SVM is known for producing accurate results and handling high-dimensional data well, which is a common characteristic of speech intent recognition problems. SVM is also versatile since it can handle both linear and non-linear problems, and it can be fine-tuned to handle complex input data, making it versatile and adaptable to different audio speech recognition problems. [7], [8]

C. Hyperparameters tuning

There are 3 main sets of hyperparameters to be tuned:

- n mfcc - It specifies the number of MFCC coefficients to keep from the MFCC computation. Typically, the first few coefficients capture the main spectral shape of the signal, while the remaining coefficients capture fine spectral details. Since its value determine the dimentionality of the resulting MFCC features, from our experiments, we consider nmfcc=5 is an acceptable compromise.
- augmentation parameters - In the absence of real data, one of the main goal must be obtaining credible augmented data. This can be obtained by carefully selecting the augmented factors. In our case we considered :

Technique	**Factor**	**Range**
Add noise	noise	uniform(0.1, 0.3)
Pitch scaling	semitones	randint(-5, 5)
Apply random gain	gain	uniform(2,4)
Change volume	volume	uniform(low=0.5, high=1.5)
Speed augmentation	speed	uniform(0.8, 1.4)

TABLE II: Augmentation factors

- models hyperparameters - We performed a Grid Search with the following parameters for the Random Forest:

Grid params	**values**
n estimators	[10, 20, 30]
max depth	[None, 5, 10]
min samples split	[2, 4, 6]
min samples leaf	[1, 2, 3]

TABLE III: Grid search parameters

We maintained SVM with its default hyperparameters.

III. Results

Applying data augmentation with the tuned parameters allow us to surpass the naive baseline defined. The obtained results for with the Random Forest can be considered to have a decent performance with an accuracy of 0.841. SVM has promising characteristics for this type of problems and we obtained an accuracy of 0.8 with the default hyperparameters.

Sometimes Grid Search can decrease the accuracy of a model, insted of improving it. And this is our case. This can happen when the Grid Search is choosing the hyperparameters that are not optimal for the data and the model being used. This can lead to a poor performance on unseen data.

For time constraints we propose our best results obtained with the default hyperparameters on the Random Forest and on the SVM.

The public scores obtained are 0.848 for the Random Forest and 0.849 for the SVM.

IV. Discussion

The proposed approach used to build the classification pipeline to predict the intent expressed in each audio recording, obtains resonable results that outperform the naive baseline defined. It does so by cleaning the original audio data and by apply agumentation techqniques to balance the dataset as well as by extracting a combination of different MFCCs from each audio files. Further improvements can be obtained by:

- Exploring other feature extraction approaches. One of them can be relying on Deep Learning techniques after extracting the spectrogram of each audio file.
- Use data augmentation wisely. We randomly applied to each audio file from the minority classes an augmentation technique. We have available other few information for each audio file. For example we could separate the audio files based on the gender and apply pitch scaling with the appropriate number of semitones for men and women. Another example can be the age; We could apply speed augmentation with a factor greater then one on the people with age between 41-65 supposing older people speak slowly. Similar consideration can be applied also to the different accents. With these examples we want to highlight not only the importance of selecting the most adequate parameters but also the selection of the most appropriate technique for our data.
- Run a deeper grid search process. Studying the behavior of more hyperparameters can be beneficial for improving the accuracy.

The results obtained should be further increased. The reason is that if the system accurately classifies the user's intent, it can provide more relevant and accurate responses, leading to a better user experience.

References

[1] H. Nyquist, "Certain factors affecting telegraph speed," 1928.

[2] X. Serra and A. Alwan, *Speech and Audio Signal Processing: Processing and Perception of Speech and Music*. Wiley, 2007.

[3] A. Paliwal, *Speech processing in modern communication systems*. Springer, 2009.

[4] G. Tzanetakis and P. Cook, "Mel-frequency cepstral coefficients for music modeling," *IEEE Transactions on Speech and Audio Processing*, vol. 8, no. 1, pp. 37–48, 2000.

[5] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

[6] A. Cutler and L. Breiman, *Random Forests*. Springer New York, 2004.

[7] C. Cortes and V. N. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, 1995.

[8] C. J. C. Burges, "A tutorial on support vector machines for pattern recognition," *Data Mining and Knowledge Discovery*, vol. 2, no. 2, pp. 121–167, 1998.