# *Intent Detection in Speech Recognition* Classification Problem

Andrea Pellegrino, Elisa Cascina
*Politecnico di Torino*
s309855, s318105
s309855@studenti.polito.it, s318105@studenti.polito.it

***Abstract*—In this report we introduce a possible approach for classification problem in the field of natural language processing and speech recognition. In particular, the proposed approach is based on the Mel-frequency Cepstral Coefficients (MFCC), a widely used technique for extracting features from the audio signal. Every mfcc is divided into blocks in the time domain, some summary statistics are computed and used as input for a Random Forest classification model. The presented method overcomes the benchmark set for the issue and delivers overall acceptable outcomes.**

## I. Problem overview

In general, figuring out the intended meaning or goal behind a spoken statement is crucial in creating effective voice assistants and similar audio-based programs because it enables the system to correctly interpret and respond to the user's requests and instructions.

The task of this project is a classification problem on a dataset composed by a collection of audio file in WAV format. Each recording is tagged with several attributes, such as the type of action and the object required through the intent and speaker's information. In particular, the competition goal is to correctly identify the intent of a given input audio sample by interpreting both the action requested and the object that is affected by the action.

The dataset is divided into two parts:

- a development set, a portion of audio files containing 9,854 labeled recordings
- an evaluation set, a portion of audio files containing 1,455 recordings without action and object columns

The subset of data known as the development set will be the one used to build a classification pipeline in order to determine as well as possible the intent expressed in each evaluation set's audio recording. In order to do that, we will create a model that can correctly assign labels to our files.

First of all, we can make some observations about the set used for development. Considering that an intent is given by the combination of action and object related to an audio, we can obtain from the dataset exactly 7 possible labels. These concatenated strings related to records occur in a not balanced way. For instance the "increasevolume" label is associated with 2614 records, while "deactivatelights" only appears 552 times.

Furthermore, we can observe that files do not have the same lengths: durations are very variable and range from 0.189 seconds to 12.64 seconds. Figure 1 shows the distribution of the durations of the recordings; we can calculate the standard deviation, a measure of the spread of a distribution, and the coefficient of variation, the ratio of the standard deviation to the mean multiplied by 100, that provides a normalized percentage of variability. The values are respectively equal to 0.47 and 42.56% meaning that data is spread out over a relatively wide range (high std and high CV) in comparison to the mean value. In general, we can notice that some of the signals have a length that is significantly different from the average length and after representing an example recording in the time domain (figure 2), we find that many signals have silence before and after speaking time. Since machine learning methods require comparable data, preprocessing is needed.

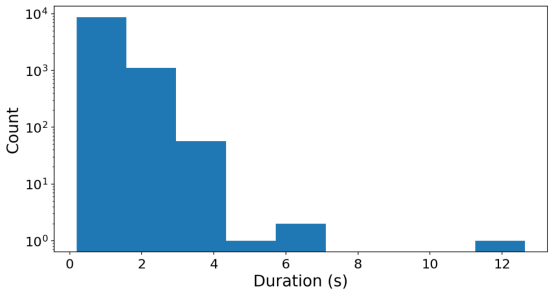



Fig. 1: Distribution of the durations of the recordings

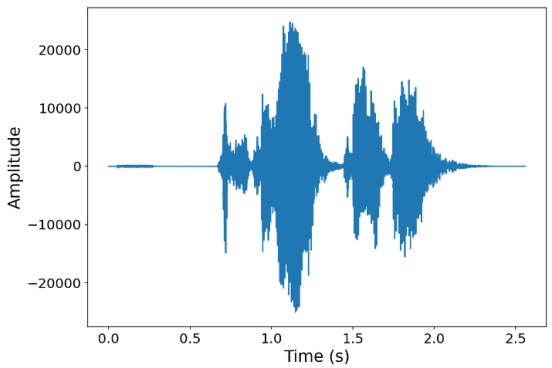



Fig. 2: Representation of an example recording in the time domain

Remarkable is the fact that also the sample rate of recordings is not the same for every signals: in particular 16000 Hz

occurs 9554 times while 300 recordings have been sampled at 22050 Hz.

As for the durations, there is a good variability in frequency term too. It can be useful to have a closer look at them using Fast Fourier Transform (FFT), that is the algorithm for converting waveforms into corresponding spectrum plots in order to perform frequency analysis. Thanks to Fourier transform, we can have a representation of the audio example in frequency domain (Figure 3).

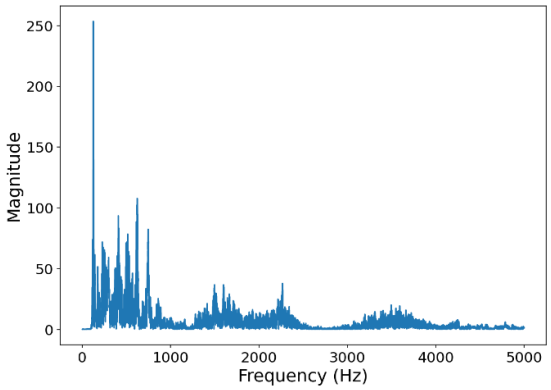



Fig. 3: Representation of an example recording in the frequency domain

## II. Proposed approach

### A. Preprocessing

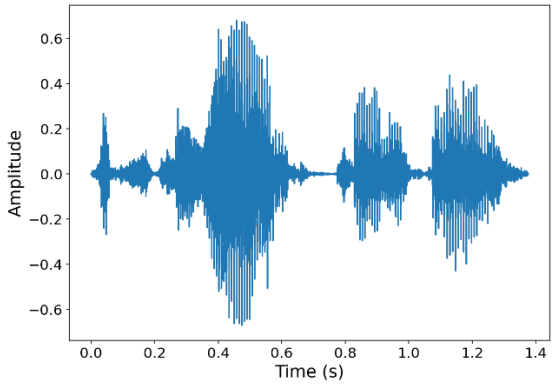
As seen before, the development set is composed by recordings that differ both in frequency and in time domain. Data preprocessing is used for solving the problem of heterogeneous data and includes the key steps we need to transform data in a way easily parsed by the machine. By ensuring consistency in data, it can increase accuracy and reliability.

First of all, we notice that audio are characterized by silence and noise. For one thing, we can trim leading and trailing silence from each signal. Secondly, we can solve noise issue by using a band-pass filter that passes frequencies within a range and attenuates frequencies outside that range. The band of frequency related to a voice recording, known as the speech frequency band or the voice frequency band, is typically in the range of 300 Hz to 4500 Hz.

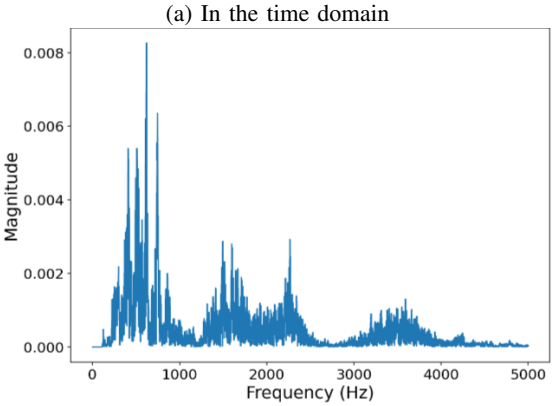
In addition, as said before, our audios differ from many points of view and that could be a problem considering that training models needs similar data to create an accurate model. In order to avoid any issues, we can normalize signals, scaling the amplitude of the signal in range [-1, 1]. This process is typically done to ensure that the level of the audio is consistent across different recordings.

The processed audio example is showed in Figure 4, both in the time (4a) and frequency domain (4b).

An important tool that shows the visual representation of spectrum of frequencies of a time series as it varies with time, is *spectogram*. In the case of audio is also known as voicegram and, as said before, it is useful to analyse signals in




(a) In the time domain




(b) In the frequency domain

Fig. 4: Representation of the processed example audio

the frequency domain. Figure 5 provides the spectogram of the processed example audio. It's worth noting that the amplitude (shown as colour intensity) is prominent due to the scale transformation from linear to logarithmic. It is remarkable too that the amplitude abides by typical range for a voice recording, as a probably result of an effective preprocessing.

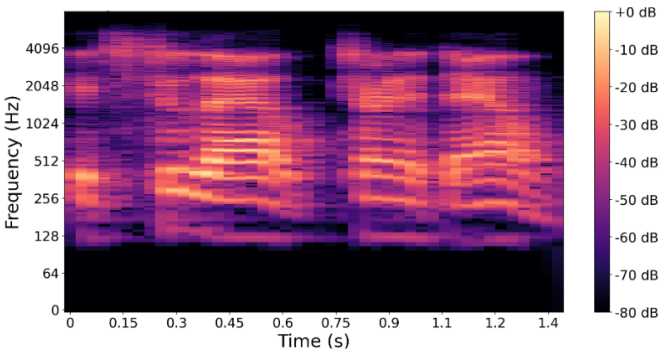



Fig. 5: Spectrogram of the example audio

The spectogram is one of the key steps for computing MFCCs of an audio signal, which will be the core point for extracting relevant features from audio recordings in this proposed solution [1].

In general, MFCCs are a compact representation of the spectrum of an audio signal which can capture its most relevant

information. When computed, they can be seen as a set of $N_{mfcc}$ coefficients that change value over time, as shown in Figure 6.

Given the spectogram of an audio signal, MFCCs can be computed as follows:

1. Mel Spectogram: Spectogram frequencies are converted using the mel scale (a unit of pitch such that equal distances in pitch sounded equally distant to the listener)
2. Log Power Spectrum: The power spectrum of the Mel-scale spectrogram is taken, and the logarithm of the power spectrum is calculated.
3. Discrete Cosine Transform (DCT): The log power spectrum is transformed into the frequency domain using the DCT. The results are a set of $N_{mfcc}$ MFCCs ($N_{mfcc}$ is a pre-selected value).

MFFCs, from the 1st to the last one considered, progressively lose the amount of information carried. It has been found that the most relevant coefficients are about the first 20 because more related to the shape of the vocal tract, that creates the sounds.

In the solution proposed, only the first $N_{mfcc}$ MFCCs are chosen, each one divided in $n_b$ blocks along the time dimension. This produces a total of $N_{mfcc} \cdot n_b$ blocks. For each of these blocks, two summary statistics are computed: the mean and the standard deviation. In this way each audio recording will be identified by a set of $2 \cdot N_{mfcc} \cdot n_b$ features, which will be used for the model training. The feature extraction process is shown in Figure 7. Another useful parameter is the hop length $h$ of the sliding window used for computing MFCCs. As $h$ decreases, it has the effect of increasing the resolution in time of MFCCs. An higher resolution can provide more accurate values of the statistics computed for each block.

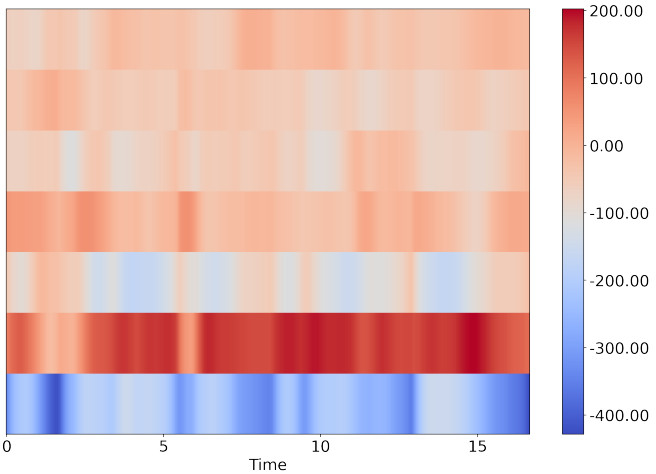



Fig. 6: MFCCs representation over time of the example recording ($N_{mfcc} = 7$)

### B. Model selection

The classification model chosen is the *Random Forest*: the algorithm constructs a multitude of decision trees, where each tree considers a different subset of features at training time. The Random Forest output class will be chosen by majority voting among the output class of each decision tree. One of the advantages of using this model is that it reduces the risk of overfitting, given that each tree is trained on slightly different subsets of the features. It has been shown that this model works really well for audio classification problems [2]. The best hyperparameters configuration has been chosen using a grid search as it will be better explained in the next section.

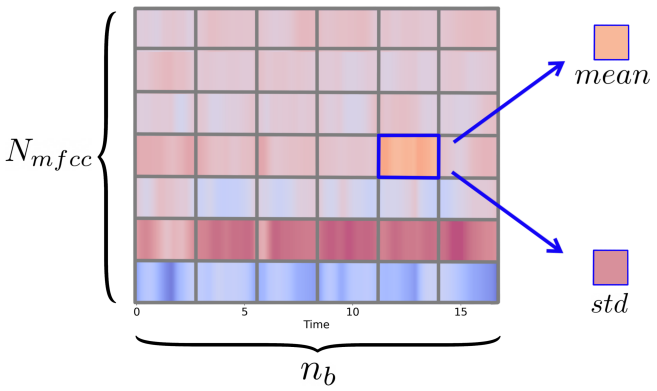



Fig. 7: Feature extraction example, $N_{mfcc} = 7$, $n_b = 6$

### C. Hyperparameters tuning

The main hyperparamters to be tuned are:

- $N_{mfcc}$, $n_b$, $h$ for the preprocessing phase
- random forest parameters

| | Parameters | Values |
|---|---|---|
| Preprocessing | $N_{mfcc}$ | $5 \rightarrow 10,\ 20$ |
| | $n_b$ | $10 \rightarrow 30,\ step = 5$ |
| | $h$ | $2^k,\ k = 5 \rightarrow 9$ |
| Random Forest | $n\_estimators$ | $\{100, 200, 500, 1000\}$ |
| | $max\_depth$ | $\{\text{None}, 5, 10, 20\}$ |
| | $criterion$ | {gini, entropy, log_loss} |

TABLE I: Hyperparameters considered

Parameters' values for the preprocessing phase has been chosen in order to avoid elevated complexity of the model and achieve acceptable results at the same time. Higher values of $N_{mfcc}$ can overcomplicate the model without adding relevant information, as explained in the previous section for MFCCs, while $n_b$ and $h$ shall be set such that the information contained along the time dimension can be summarized efficiently. The random forest parameters are chosen accordingly to the table I above, after the preprocessing parameters have been set.

## III. Results

The preprocessing parameters that have provided the best results are:

- $N_{mfcc} = 7$
- $n_b = 20$
- $h = 64$

This configuration provides a total of 280 features for each audio recording that will be used for training.

We use a grid search in order to achieve the best performance for the random forest model. The outcome configuration is $\{n\_estimators = 1000, criterion = log\_loss, max\_depth = None\}$. The best performing random forest has been trained on all development dataset and then used to label the evaluation set. The final configuration provided a public score of $0.852$. The same accuracy has been achieved with the following settings $n_b = 25$ and $h = 32$, but this configuration has been discarded since it would have increased the number of features without increasing the precision.

The result obtained on the public evaluation set are reasonable and we can assume that a similar score can be achieved on the private one.

## IV. Discussion

Overall, the discussed approach achieves our goal to overcome the naive baseline, thanks to the ability of MFFCs to catch an overall view of audios.

However, we could take into account some other elements for further improvement:

- Further classification strategies may be evaluated. First of all, one model that is largely used for audio signals classification is SVM (Support Vector Machine) [3]. It uses a kernel function to find a decision boundary that separates audio signals into different categories. An other promising approach may be Convolutional Neural Networks, very powerful and accurate when has to deal with great amount of data, like in our case.
- Other important factors are the features adopted for training model. Further options, other than evaluating the basic statistics, such as mean or standard deviation, are indices as Skewness (a measure of the distribution asymmetry) or Kurtosis (a measure of the distribution "tailedness"). Furthermore, we could consider intrinsic features of the audio signal such as peak amplitude, velocity of changes, derivatives or other features in time or frequency domain. However, in some cases, the combination of multiple features can provide better results.

This complex dataset certainly leaves room for improvements nevertheless the results already achieved are satisfactory.

## References


[1] T. D. Ganchev, N. Fakotakis, and G. K. Kokkinakis, "Comparative evaluation of various mfcc implementations on the speaker verification task," 2007.

[2] F. Saki, A. Sehgal, I. Panahi, and N. Kehtarnavaz, "Smartphone-based real-time classification of noise signals using subband features and random forest classifier," pp. 2204–2208, 03 2016.

[3] P. Dhanalakshmi, S. Palanivel, and V. Ramalingam, "Classification of audio signals using svm and rbfnn expert syst," *Expert Systems with Applications*, vol. 36, pp. 6069–6075, 04 2009.