Intent detection for English audio samples

Jacopo Lungo Vaschetti
Politecnico di Torino
Student id: s306074
s306074@studenti.polito.it

Mohammad Sadegh Radmehr
Politecnico di Torino
Student id: s301623
s301623@studenti.polito.it

***Abstract*—This report presents a model for classifying the intent of a given short audio or voice command. Our model will exploit statistics obtained from the Mel spectrogram, the mfccs and some speaker's attributes. It utilizes SVM and Random Forest as classification models. We obtained satisfying results with respect to the baseline.**

I. Problem overview

Voice intent detection is the process of determining the objective or purpose of an input sample (audio) and classifying it to accomplish the intent. Detecting the high variability and ambiguity in spoken language is, of course, a challenge that extends beyond purely technical issues. Throughout manual exploration, we discovered that speakers pronounce words differently in the given dataset, and sentences with different words are associated with the same label. In addition, background noise and other acoustic distractions make it challenging to assign the right label. The supplied dataset contains two distinct sections:

- A development set of 9854 labelled recordings, associated with some information about the speakers.
- An evaluation set of 1455 data points with the same structure as the development except without labels.

Using the development set, we will construct a classification model to label evaluation set points.

In the development set, there are seven labels. Even if their distribution is not well balanced (Fig. 1) we assume that the evaluation set has the same distribution.

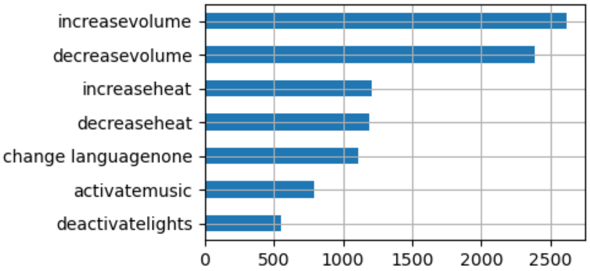

Fig. 1: Distribution of different voice command intentions

Data about speakers include the following features:

- *speakerId*: the id of the speaker.
- *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.

Analyzing them we discovered that the vast majority of the people in the dataset are reported to speak English as their first language and have native fluency. The age range is unbalanced toward the range 22-40. Analyzing audios in more detail, we found out that the sample rate is 16kHz for most of the samples except for 200 of them which are sampled at 22.05kHz. Furthermore, audio has different lengths (Fig. 2). Those that differ substantially by the mean duration (2.6 sec.) contain various seconds of silence. However, even shorter ones contain silence for the majority of the length, as we can see in figure 3. We have to take care of that in the preprocessing phase.

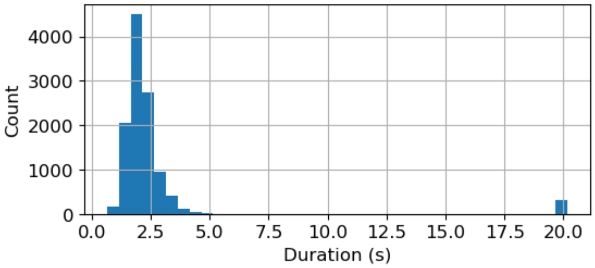

Fig. 2: Distribution of durations of the recordings

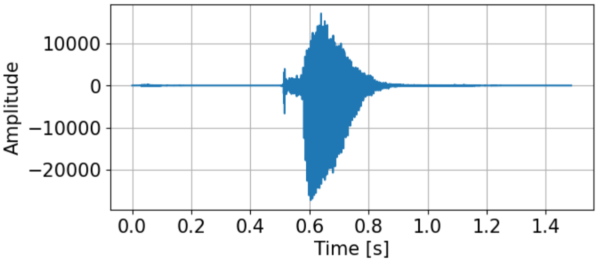

Fig. 3: Wave form of a sample not trimmed and not normalized.

II. Proposed approach

A. Preprocessing

We have to preprocess both audio signals and speakers' related features. Starting from the latter, we convert *ageRange* and *Self-reported fluency level*, which are discrete ordinal attributes, into ordinal numbers and we use one-hot encoding to deal with *gender* attribute since it has not a hierarchical order. Attributes about *speakerId* and language are discarded because no better results were obtained when included. This could be since using one-hot encoding on the *speakerId* adds 97 features that probably bring noise to our model.

To align the recording volume, audio data are normalized so that all the data points are between -1 and 1. When reading the data, we decide also to store the sample rates since they may differ between audios as analyzed previously. Then, we remove silence at the beginning and at the end of each audio since it does not contain useful information. After that, for each audio we extract the length in seconds, the log Mel spectrogram and the Mel-Frequency Cepstral Coefficients (MFCC):

- *Log Mel spectrogram*: it is similar to a standard spectrogram but with different units of measurement. The frequencies are converted from Hz to the mel scale, the amplitude is converted to decibels. These units of measure are based on the human perception of audio signals [1]. As we can see (Fig. 4), values on the x and y axis are not continuous but bins. Specifically, on the y axis there are 40 bins (the mel bands) and the Hz scale for comparison. Then, we divide it into n rows and n columns in order to obtain n^2 submatrices and compute for each of them the mean and the standard deviation. Generally speaking, any signal could be analysed in the time domain or in the frequency domain. This process exploits both. Also, we do not have to take into account that audios have different lengths since this method will extract $2 \cdot n^2$ features from each of them.

- *MFCCs*: they are obtained from the Log Mel Spectrogram by applying the discrete cosine transform. A representation is in figure 5. They are often used for audio classification alongside Random Forest and SVM [2]. MFCCs retrieved with the python library Librosa are matrices where each row is associated with an MFCC. Since the different lengths of audios, all these matrices will have different numbers of columns but the same number of rows. It is convenient to take the mean of each row as features.

Even if in the literature we did not find the usage of both Log Mel spectrogram and MFCCS for speech classification, after trying to use them individually, we achieved better results exploiting both.

B. Model selection

As anticipated, for this task we decided to use SVM and random forest.

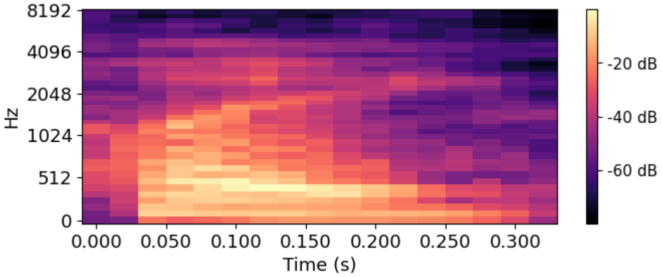

Fig. 4: Mel spectrogram of an audio track.

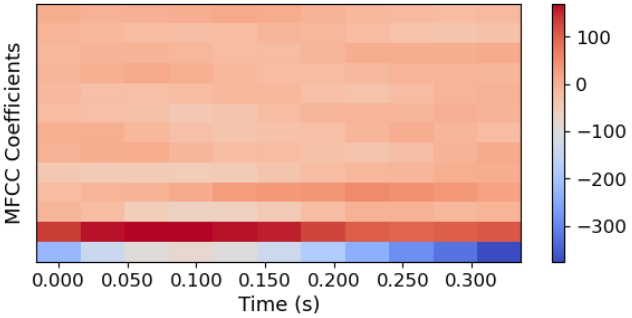

Fig. 5: MFCCs of an audio track.

- *SVM*: it is one of the best performers, and even though is not interpretable is a very fast classificator. It is suitable for cases where the number of features is high. It creates a hyperplane that separates the data into classes. This model benefits from feature normalization. So, we use a standard scaler before the model.
- *Random Forest*: using a set of decision trees it assigns a class by majority voting. These trees are decorrelated because the subset of features they use are sampled randomly. This model is fast to build and the features do not require normalization since they are evaluated individually.

C. Hyperparameters tuning

There are a lot of hyperparameters both in the preprocessing step and for the models. However, for simplification, we keep some of them fixed. Examples are the number of mel bands in the creation of the mel spectrogram and the number of coefficients in the MFCC extraction. For both of them, we decided to use values found in literature [1]. The hyperparameters we evaluate are:

- n: the number of rows and the number of columns in which we divide the mel spectrogram.
- *SVM parameters*: the kernel and C
- *Random Forest paramters*: max depth and criterion.

To reduce the number of possible combinations we decided to try different numbers for n with the default parameters for SVM and random forest. Then we can search for the best hyperparameters also for them. For both preprocess and model hyperparameter tuning we decide to use cross-validation with 5 folds since it provides a better understanding of model performance instead of a single train/test split. We also ensure that each fold is stratified[1] since we saw that classes are not evenly represented. The metric to be evaluated is *accuracy*. For the tuning of *n* we calculate it as the mean of the accuracies of the five folds[2]. After that, having *n* fixed, we use grid search to evaluate the two models testing the hyperparameters in table I.

Model	**Parameters**	**Values**
Preprocessing	n	$2 \rightarrow 20$, step 2
Random forest	*max_depth*	{None, 2, 5, 10, 50}
	n_estimators	300
	criterion	{gini, entropy}
SVM	*C*	{0.1, 1, 5, 10, 50, 100, 500, 1000}
	kernel	{rbf, linear}

TABLE I: Tested hyperparameters

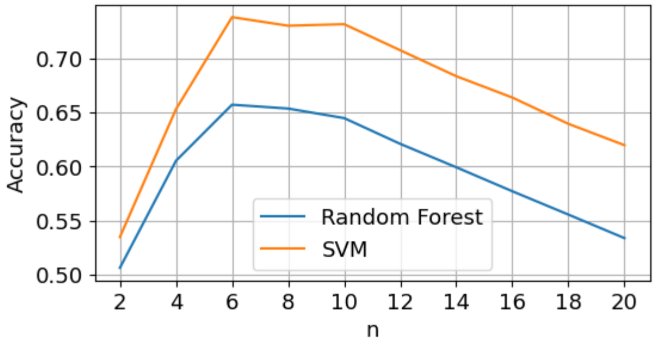

Fig. 6: Random Forest and SVM's accuracy for different values of *n*.

III. Results

As in figure 6 the accuracy is not very stable and seems deeply influenced by *n*. The best results are achieved for $n = 6$ for Random Forest and for $n = [6, 10]$ for SVM. The best accuracy achieved was about 0.66 for random forest and about 0.74 for SVM. We choose $n = 6$ and run the grid search for the two models. The best configurations were:

- SVM: {*C* = 10, *kernel* = *rbf*} (*accuracy* approx. 0.80)
- Random Forest: {*criterion* = entropy, *max_depth* = 50, *n_estimators*: 300} (*accuracy* approx. 0.68)

The random forest performs worst than the SVM. This could be because random forest randomly selects features for its trees and having a lot of features (if some of them are irrelevant) will introduce noise and reduce performance. For this reason, we decide to use just SVM for classifying the evaluating set. The score obtained on the leaderboard is 0.859. Since it is higher than the score obtained on the development set, we can say that overfit did not happen. Furthermore, since we did not change parameters to maximize the public score, we believe that the private score will not be much different from the public one.

IV. Discussion

We are satisfied with the result since it is much higher than the baseline. Also, this classification problem does not seem trivial for the reasons explained at the beginning of the report (e.g. speakers tell different sentences that are associated with the same label). To improve the results more hyperparameters could be considered and tested. For example, we could have searched for the optimal number of mel bands or the optimal number of MFCCs. Another option could be using more advanced techniques like artificial neural networks. The research has shown great results in classifying audio samples by feeding their spectrogram's images into a Convolutional Neural Network [3].

References

[1] H. M. Fayek, "Speech processing for machine learning: Filter banks, mel-frequency cepstral coefficients (mfccs) and what's in-between," 2016.

[2] B. Vimal, M. Surya, Darshan, V. Sridhar, and A. Ashok, "Mfcc based audio classification using machine learning," in *2021 12th International Conference on Computing Communication and Networking Technologies (ICCCNT)*, pp. 1–4, 2021.

[3] M. Lim, D. Lee, H. Park, Y. Kang, J. Oh, J.-S. Park, G.-J. Jang, and J.-H. Kim, "Convolutional neural network based audio event classification," *KSII Transactions on Internet and Information Systems (TIIS)*, vol. 12, no. 6, pp. 2748–2760, 2018.

[1] A fold is said to be stratified if it contains the same percentages of each class as it is in the full dataset

[2] We accept the approximation of using just the mean and not the mean weighted on the number of samples in each fold.