Spoken language intent classification, a log-Mel spectrogram based approach

Riccardo Moroni
Politecnico di Torino
Student id: s259169
s259169@studenti.polito.it

Jacopo Bracci
Politecnico di Torino
Student id: s314674
s314674@studenti.polito.it

***Abstract*—In this report we introduce a possible approach to the Spoken Intent classification problem that was proposed us. In particular we intend to demonstrate how classifying over features extracted from the Mel-Spectrograms of the recordings produces satisfying results for the task at hand, comparing the performances of 3 different classifiers.**

I. Problem overview

The proposed competition is a classification problem over spoken intents. The goal is to correctly identify the action asked to perform and the object on which it has to be performed.

The label we want our model to correctly assign the evaluation dataset's recordings with is the concatenation of the two strings action + object.

Example: if the recording says "I can't hear anything, turn up the volume" the correct label would be: increasevolume.

We were provided with:

- a *Development* dataset: of 9854 recordings characterized by their speaker ID, action, object, speaker self reported fluency level, first language spoken, current language used for work/school and gender and age of the speaker. For the action and object attributes there are 5 possible values each, but one value for the object is "none" as can be seen in Fig. 1.
- an *Evaluation* dataset: of 1453 recordings with the same structure as the development one but the action and object columns.

We first proceeded inspecting the development and the evaluation datasets looking for anomalies.

No duplicates or missing values were found in the development dataset, so we didn't make use of any kind of padding for the already provided features.

The evaluation dataset, though, has significally different distributions of values for some features. Both "Self-reported fluency level" and "Current language used for work/school" only assume one value each, so, it would have been useless learning on those features too.

As shown in Fig 1 only few combinations of all possible couples of action and object are present in the development dataset, but since the evaluation dataset is supposed to have the same distribution as the development one, not considering all the other possible couples should't have affected our performances.

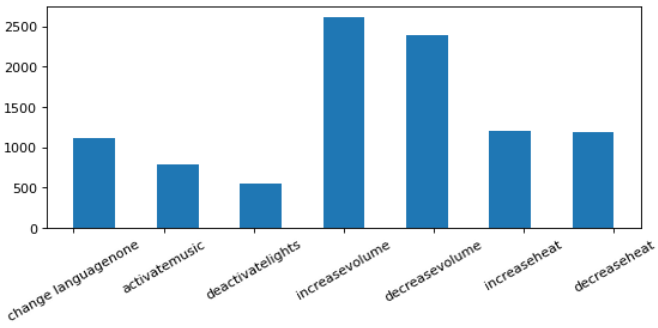

Fig. 1: Distribution of the intents

In Fig. 2 it's shown how the recordings are distributed as for their durations. We can see there's a set of 20 seconds long recordings, they are 300 and by inspecting some of them we noticed that they tend have a lot of trailing and/or tailing silence.

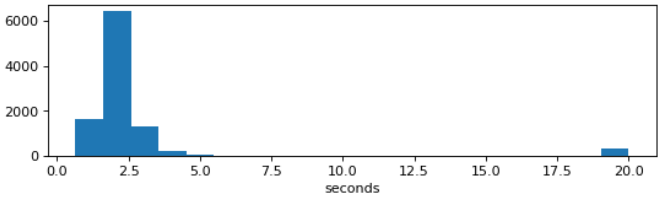

Fig. 2: Distribution of recording lengths

Speaking about the sample frequency of the recordings, they do not share all the same one (the 20 seconds long recordings has a sample frequency of 20.5 kHz while the others of 16 kHz), however, the usable voice-frequency band ranges from approximately 300 Hz to 3400 Hz, so, according to Nyquist-Shannon sampling theorem, all the recordigs are properly sampled.

Once these considerations were done, the problem seemed us well balanced at most, but an important preprocessing was still needed.

II. Proposed approach

A. Preprocessing

The first preprocessing step is the trimming of unwanted silence frames or abnormal and rapid peaks (such as mouse clicks) present at the beginning or end of the audio signals. For each audio the cumulative energy is computed. In the presence of a sudden and rapid peak or a silence frame the cumulative energy function doesn't increase greatly and always stays below a certain threshold (found to be at: the maximum amplitude$-$55dB). We can now trim the part of the signal where the cumulative function doesn't surpass the mentioned threshold. The same idea can be applied to the end of the signal using as function the maximum amplitude minus the cumulative energy (so that we can have a value close to zero at the end of the signal). After performing this step the audio length is much more uniform having now a smaller mean length and a much lower standard deviation (as summarized in Table 1). This will greatly impact the feature extraction steps since we now have recordings with similar length.

TABLE I: Before and after silence removal.

	Before	After
Mean [s]	2.64	**1.13**
Std [s]	3.13	**0.51**

Despite the proven efficiency of the silence and peaks removal we made, we however chose to delete remaning outliers, which we identified in those having duration $<$0.3 sec or $>$4 sec.

Since a spoken audio signal may have frequency components that fall off at high frequencies, we applied a pre-emphasis to the recordings with a first order differencing filter [1] in order to avoid overlooking them. The filter is given by the input-output relationship in the time domain (1) where $x_p(t)$ is the pre-ephasized sample, $x(t)$ is raw signal sample at time t and a is the pre-emphasis coefficient which lies in the interval [0.95, 0.98].

$$x_p = x(t) - a \cdot x(t-1) \tag{1}$$

The second preprocessing step is data augmentation. Data augmentation is a technique that consists in increasing the amount of training data by modifying existing data, and consequently increasing model reliability and generalization. The main technique used is Pitch Shifting combined with the gender attribute of the dataset. Its aim is to make pitch changes and consequently raise or lower the original tone of the voice of a male or a female speaker respectively. In the Librosa implementation, it's important to notice that the process used to shift the pitch doesn't affect the length of the original sound clip. Being just a data augmentation step we didn't focus particularly on how pitch and gender are related togheter taking a more straightforwand approach using the results of [2].

Since an audio can be described with complementary features which belong to time and freqeuncy domain, we needed a representation which brought the information of both domains: spectrograms. Spectrograms can represent a signal as it's frequency spectrum varying in time leveraging a time-frequency visualization in combination with a colored scale. To obtain the spectrogram it is necessary to apply a Short-Term Fourier Transform (STFT). At first each audio signal is divided into several frames whose length is 32ms with an overlap of 8ms (both values refer to a 16kHz audio). To each frame a Window function is now applied in order to reduce the spectral leakage as much as possible. The window process is effectively weighting the original frame according to the shape of the window function favoring samples that lie towards the center of the window. The most commonly used functions are: Hamming, Hanning and rectangular and in this analysis the Hanning function is used. As shown in [3] the analysis carried out with windows of comparable size is usually sufficient to provide good spectral resolution of the audio, and at the same time short enough to resolve significant temporal characteristics Each windowed frame is converted into magnitude spectrum by applying DFT (2).

$$X(k) = \sum_{n=0}^{N-1} x(n) \cdot e^{\frac{-j2\pi nk}{N}}; \qquad 0 \leqslant k \leqslant N-1 \tag{2}$$

Human perception of sound isn't linear but logarithmic both in amplitude and in frequency. For example high frequencies changes aren't detected as clearly as low frequencies one. This introduces the need of a new scale along the frequencies that better represent the human auditory system. The Mel scale is approximately a linear frequency spacing below 1 kHz and a logarithmic spacing above 1 kHz according to (3)

$$\text{Mel}(f) = 2595 \cdot \log_{10}(1 - \frac{f}{700}) \tag{3}$$

The Mel spectrum is now computed by passing the Fourier transformed signal through a set of band-pass filters known as Mel-filter bank. These filters have a triangular shape and their characteristic is not having the center frequencies evenly distibuted as in normal filters, but they follow the Mel scale according to (3). The Mel spectrum of the magnitude spectrum $X(k)$ is computed by multiplying the magnitude spectrum by each of the of the triangular Mel weighting filters as in (4).

$$s(m) = \sum_{k=0}^{N-1} [|X(k)^2| \cdot H_m(k)|]; \qquad 0 \leqslant m \leqslant M-1 \tag{4}$$

where M is total number of triangular Mel weighting filters. $H_m(k)$ is the weight given to the k^{th} energy spectrum bin contributing to the m^{th} output band.

Once the mel-spectrograms were computed we leveraged the fact that each mel spectrogram is a 2-dimensional matrix virtually "sliciable" into the same number of small non-overlapping blocks. For each block its mean value was pooled and concatenated to the overall feature vector. It's important to

notice that the size of the block was variable for each spectrum in order to guarantee the same number of features for each spectrogram.

Along with these extracted features, we trained our models with "gender" and "age" already provided ones. Being categorical features we needed to encode them as numerical thru the use of a One Hot Encoder.

B. Model selection

For the classification task these algorithms have been tested:

- *Random Forest* : a tree-based ensable method, it leverages multiple decision trees training each of them on a different subset of features and dataset portions. It's really powerful but very expensive in terms of building and training time, making it not well scalable with a high number of features. One of the main characteristics of this algorithm is the possibility of computing the feature importance after the training step, giving the chance to better understand which subset of features has more relevance in the classification. Another pro of random forests is that they do not need any normalization on the features they are fed with (as the simple trees they rely on).
- *K-Neighbors* : this algorithm relies on distance for classification, if the features represent different physical units or come in vastly different scales then normalizing the training data can improve its accuracy dramatically. We must be aware of the curse of dimensionality that may affect this algorithm: if the number of features is too large then the data will become too sparse making the concept of distance basically useless. With the right tune of paramethers this can be avoided.
- *Support Vector Machine* : a supervised learning algorithm whose aim is to draw the hyperplane which best separate points with different labels in the N-dimensional space by solving a convex optimization problem. The hyperplane not only has to separate different kinds of points but also has to maximize the margin to points in either class. The points that fall exactly on the margin are called the supporting vectors. Since this method relies much on the concept of the distance, the same considerations about normalizing can be made as in the KNeighbors algorithm.

C. Hyperparameters tuning

In this section we fine-tune the previously said models. In order to find the best parameters for each configuration we did many different gridsearches.

In addition to all the model-specific paramethers, we ran gridsearches on some preprocessing paramethers too, such as the dimensions of the sliding window from which the mean and std were pooled (n_f and n_t), the number of mel-bands generated (n_mels), the amount of decibels below the maximum amplitude from which to trim the audio when surpassed (trim_top_db) and the number of short overlapping windows on which to compute the discrete Fourier transform when passing to the frequency domain (n_fft). The firsts, in particular, are essential, since they will determine the tradeoff between granularity and a non excessive number features.

The preprocessing paramethers and the model-specific hyperparamethers we tuned with their candidate values are all resumed in Table II.

The gridsearches were structured in a way that, for every possible configuration of model-specific hyperparamethers, the feature extraction was performed $\#n_f \cdot \#n_t$ times (where $\#x$ is the number of candidates for x), iterating on the values of n_f and n_t.

TABLE II: All the parameters we evaluated each model for.

Preprocessing	
n_f	2, 4, 8
n_t	10, 14, 20, 25
n_mels	64, 128
trim_top_db	15, 18, 21
n_fft	256, 512
Random Forest	
n_ estimators	500, 750, 1000, 1500
max_features	sqrt, log2
criterion	gini, entropy
min_samples_leaf	1, 2, 4
min_samples_split	2, 4, 5
SVC	
kernel	poly, rbf
max_iter	5000, 10000
gamma	scale, auto
tol	$1e^{-4}, 1e^{-3}, 1e^{-2}$
C	10, 100, 200, 500
K-NN	
n_neighbors	5, 7, 9, 11
p	1, 2
weights	uniform, distance

The pipeline was built in such

III. Results

The results of the three different model are shown in Fig.3. The scores shown are the result of a 5-fold cross validation on just a subset of the training data in order to lower the computational cost. The results clearly shows how the SVM has outperformed the other two ....

In Fig. 4 we can see the models learning curve based on the number of training examples provided. The scores are again obtained through a 5-fold cross validation for the possible number of training examples. The mean value and the standard deviation of the scores are shown as the full line and colored region respectively. From the plot we can undersand how the models have logarithmic growth which tends to stabilize at higher values of training examples.

The best public score we achieved is **0.938** and was reached by a Support Vector Classifier, setted with the parameters in Table 3.

IV. Discussion

The proposed pipeline, through the simple means of data analysis and machine learning, is able to reach competitive results. But there is room for improvement:

TABLE III: Paramethers of the SVC for our best public performance

kernel	rbf
max_iter	10000
gamma	scale
tol	0.001
C	300

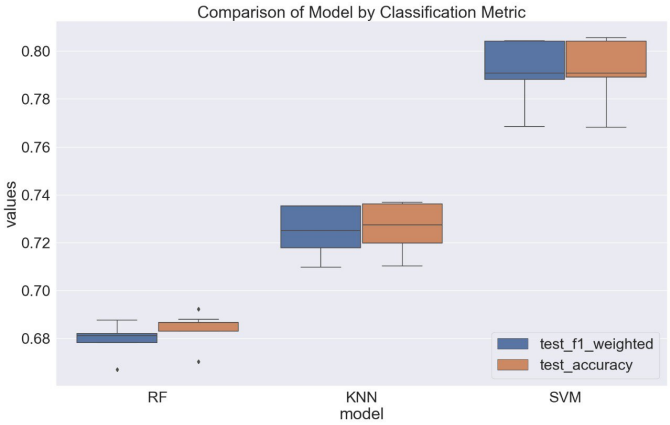

Fig. 3: Models comparison based on f1 score and accuracy

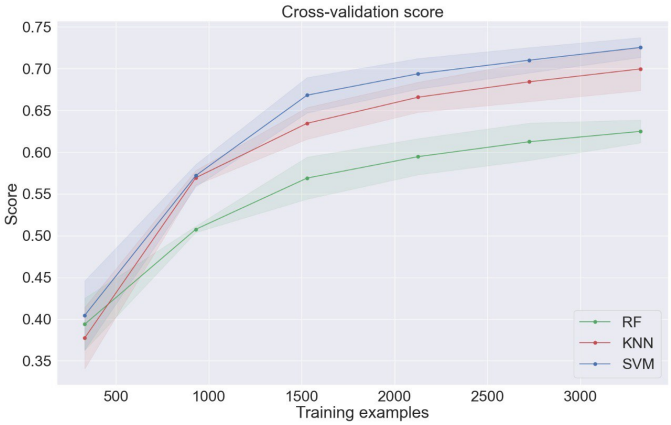

Fig. 4: Models learning curves.

- develop a more complex feature extraction algorithm that is not just based on the mean value of sub-blocks of the original spectrogram but on some other relevant features.
- Use the Mel Frequencies Cepstral Coefficients as features. Even though this method was running a more thorough grid search process for the proposed models. This process allows to achieve better results, but it necessarily requires a higher computational time.
- reducing the dimensionality and the sparsity of the dataset through the use of specific algorithm such as PCA or SVD.
- using a more complex model, such as a neural network (e.g. MLP Regressor). Nevertheless we remark the fact that it is not trivial to manage a deep learning model and we avoided this approach precisely because it was out of the scope of this project.

REFERENCES

[1] J. Picone, "Signal modeling techniques in speech recognition.," (1993).

[2] D. Rendall, S. Kollias, and C. Ney, "Pitch (f0) and formant profiles of human vowels and vowel-like baboon grunts: The role of vocalizer body size and voice-acoustic allometry," *Journal of Acoustical Society of America*, p. 944–955, 2005.

[3] K. E. M. K. Sreenivasa Rao, *Speech Recognition Using Articulatory and Excitation Source Features*. SpringerBriefs in Speech Technology, Springer, 2017.