Intent Detection on the Fluent Speech Commands Dataset

Iman Morovatian
Politecnico di Torino
Student id: s310262
s310262@studenti.polito.it

Hossein Khodadadi
Politecnico di Torino
Student id: s313884
s313884@studenti.polito.it

***Abstract*—In this report, a possible approach is proposed to the *Fluent Speech Commands Dataset* intention detection problem. Firstly, the proposed aproach converts the problem into a classification one. After that, performs preprocessing on data and computes Mel Frequency Cepstral Coefficients (MFCC) in order to extract some features. Finally, it applies Support Vector Machine (SVM) and Convolutional Neural Netwroks (CNN) as classification models on the preprocessed data. Approximately, the results of the proposed approach are satisfactory.**

I. Problem overview

The proposed problem is an intent detection on the *Fluent Speech Commands Dataset*. The dataset consists of audio recordings of different people and a CSV file showing some characteristics of the people and information about their intentions. The CSV file contains the following features:

- Id: id of each row
- path: path in which the corresponding audio recording is stored
- speakerId: id of the person saying the sentence in the audio recording
- action: action desired by the person
- object: object involved in the desired action
- Self-reported fluency level: fluency level of the spoken language reported by the person
- First Language spoken: the first language spoken by the person
- Current language used for work/school: the language used by the person currently at work or in school
- gender: gender of the person
- ageRange: the range in which the age of the person is

The goal is to correctly identify the intention of people by predicting *action* and *object* columns. In this way, the task is a classification problem. To simplify the problem, the concatenation of *action* and *object* columns is considered as the prediction target (class). The dataset is divided into two parts:

- development set containing 9855 records which will be used for model building and hyperparameters tunning
- evaluation set containing 1456 records without *action* and *object* columns. This set will be used for the final model's evaluation

According to the development set, some considerations can be made. First, the problem is not well-balanced. The relative frequency of each class is given in Table I. Second, all audio recordings have been sampled at a frequency of 20.05 kHz, while they differ in duration. Figure I shows the histogram of audio recordings duration. It seems that there are some audio recordings significantly varying in duration. More specifically there are two audio recordings exceeding 6 seconds. Upon manual inspection, one of them is just silence and in the other one, a person repeats a command.

TABLE I
Distribution of Classes in Development Set

Class	Relative Frequency
increase volume	0.26%
decrease volume	0.24%
increase heat	0.12%
decrease heat	0.12%
change language	0.11%
activate music	0.08%
deactivate lights	0.05%

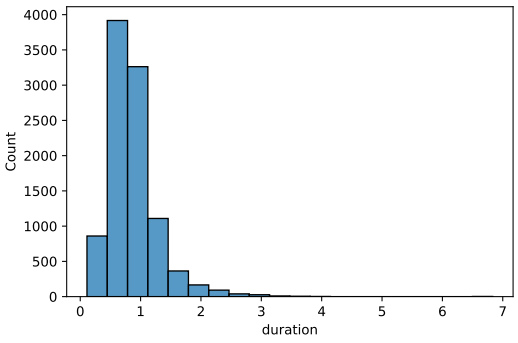

Fig. 1. Histogram of audio recordings' duration

II. Proposed approach

Following the common framework for data science problems, the proposed approach contains preprocessing, model selection, and hyperparameters tuning phases. Overall, it makes data clean and extracts some features from that using the Mel Frequency Cepstral Coefficients (MFCC). Then, applies two

classification algorithms including Support Vector Machine (SVM) and Convolutional Neural Networks (CNN) on it.

A. Preprocessing

1) Feature Selection: Among all available features, only audio recording is selected and other features showing characteristics of the speaker persons in the audio recordings are removed. The *speakerId* column is ignored because of its high cardinality (87 unique values). Meanwhile, *Self-reported fluency level*, *First Language spoken*, and *Current language used for work/school* columns contain dominant values and do not provide much information about the class of data. Dominant values and the corresponding relative frequencies of them for the mentioned columns are shown in Table II. Moreover, in *gender* and *ageRange* columns, values follow

TABLE II
RELATIVE FREQUENCY OF DOMINANT VALUE IN SOME FEATURES

Feature	Dominant Value	Relative Frequency of Dominant Value
Self-reported fluency level	native	95%
First Language spoken	English (United States)	97%
Current language used for work/school	English (United States)	97%

the same pattern in different classes, so knowing their values cannot help the model predict the class. In Figures 2 and 3, the distributions of values of these two features in different classes are illustrated.

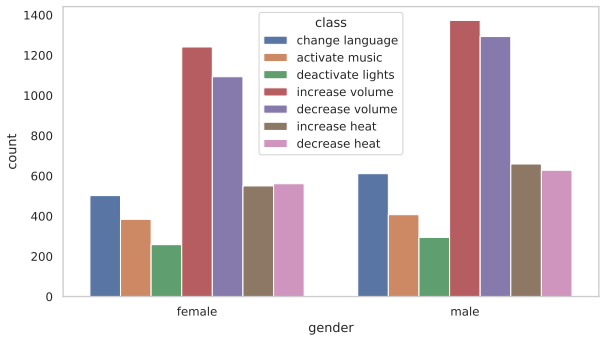

Fig. 2. Frquency of Values of *gender* column in different classes

2) Trim: There is silence at the beginning and end of some audio recordings. This silence is redundant and does not provide information. Thus, all audio recordings are trimmed.

3) Remove Outliers: As was mentioned in section I, some audio files have considerably long duration compared to others. They can affect the performance of the model and other

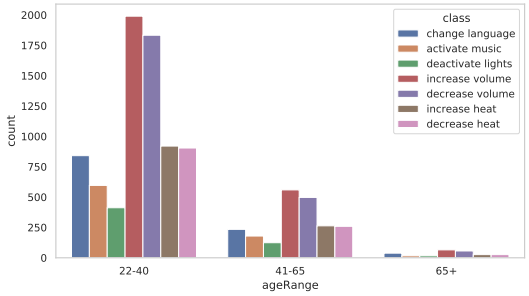

Fig. 3. Frequency of Values of *ageRange* column in different classes

preprocess phases. Hence, they are considered outliers and are removed.

4) Add Padding: Feature extraction on audio recordings with different duration culminates in a different number of features, while most classification models expect a fixed number of features as input. To address this problem, the maximum duration of audio recordings is determined and all audio recordings are padded with zeros in order to have the same duration.

5) Feature Extraction: Information extracted from signals in the time domain can be used to train the model still, to achieve high accuracy useful features should be extracted as much as possible. This can be done by transforming the signals to frequency domain. MFCC (Mel-Frequency Cepstral Coefficients) is a method for representing audio signals in a compact form. It involves transforming the audio signal into a representation that captures the relative power of different frequency bands, by converting it into the Mel frequency scale, which is applying a Fourier transform, and taking the logarithm of the magnitude spectrum. The resulting coefficients are then further processed to obtain a compact and discriminative representation of the audio signal [1]. Figure 4 depicts an example of MFCC conversion of a sample audio.

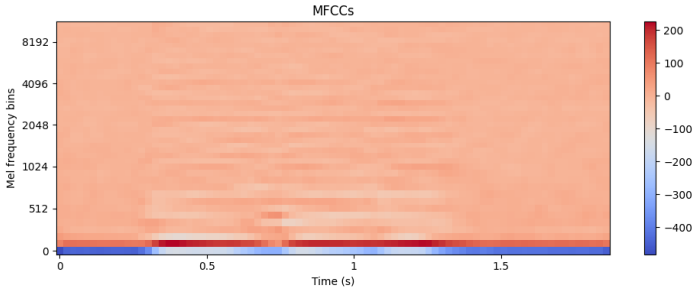

Fig. 4. MFCC Transformation

6) Apply PCA: The MFCC for each audio recording is a two-dimensional array. While some algorithms like CNN can be applied directly to such data, a group of algorithms such as SVM is not able to handle it. To pass a two-dimensional array to this group of algorithms, a possible approach is to make the array flatten and convert it into a one-dimensional

array. One disadvantage of this approach is that the number of features introduced by the one-dimensional array may be high, so there is a need for dimension reduction. Principal Component Analysis (PCA) is adopted as the dimension reduction algorithm.

7) Standardization: Standardization is performed on data in order to scale its distribution to distribution of the Standard Normal Gaussian.

B. Model Selection

1) Deep Convolutional Neural Networks: they are a specialized kind of neural network for processing data that has a known grid-like topology. CNNs are neural networks that use convolution in place of general matrix multiplication in at least one of their layers [2].They are particularly designed for image classification tasks by recognizing hierarchical patterns in images such as edges, textures and shapes. Since the output of the MFCC method can be interpreted as a 2D image this method was adopted to handle this task. However this method is hardly interpretable and computationally expensive.We chose to use CNNs as they have been shown to perform well on audio signal classification problems [3].

2) SVM: This model tries to determine a hyperplane with the maximum margin separating classes. Good performance and robustness to noise and outliers are advantages of SVM, while its model building requires significant parameter tuning. Another disadvantage is that SVM is not interpretable. It is shown in [4] that SVM works well on audio signals.

C. Hyperparameters Tuning

There are three main sets of hyperparameters to be tuned:

- Number of components in PCA for the preprocessing
- SVM hyperparameters
- CNN hyperparameters

Firstly, the number of components for the PCA algorithm is determined. To reach this goal, 80% and 20% of the development set are considered as the train and validation data, respectively. According to Table III, different values are tested for the number of components. For each value, the PCA algorithm is trained on the train data and then will be used for the transformation of train and validation data. Following that, SVM with default hyperparameters is trained on train data, and the performance of this model in terms of accuracy is evaluated by validation data. The value leading to the best performance of the SVM is considered as the number of components for the PCA algorithm in the rest of the hyperparameter tuning phase.

To tune hyperparameters related to SVM, a grid search is performed according to Table III.

Performing a grid search on CNN parameters is excessively time-consuming and inaccessible with the available computational resources, therefore a limited number of different hyperparameters were manually tested. The best hyperparameters found for the CNN model can be found in table IV.

TABLE III
HYPERPARAMETERS CONSIDERED FOR THE PCA AND SVM

Model	hyperparameter	Value
PCA	Number of Components	100, 80, 50, 40, 20, 10, 5, 1
SVM	Kernel C	rbf, poly 1, 2, 4, 8, 10

TABLE IV
THE BEST HYPERPARAMETERS FOUND FOR THE CNN

Layer Type	Filters/ Units/ Values	Output Shape	Filter Size
Conv2D	24	(19, 149, 24)	(2,2)
Max Pooling2D	-	(9, 74, 24)	(2,2)
Conv2D	48	(8, 73, 48)	(2,2)
Max Pooling2D	-	(2, 36, 48)	(2,2)
Conv2D	64	(1, 35, 64)	(2,2)
Global Max Pooling2D	-	(64)	-
Dense	64	(64)	-
Dropout	0.2	(64)	-
Dense	16	(16)	-
Dropout	0.2	(16)	-
Dense	7	(7)	-
Activation (softmax)	-	(7)	-

III. RESULTS

Figure 6 shows the tuning of the hyperparameter *number of components* for the PCA algorithm. Based on this figure, the PCA algorithm can reduce the number of features without affecting SVM's performance until 50, but after this point, SVM's performance decreases. Thus, 50 is chosen as the number of components for the PCA algorithm. The best configuration for the SVM was found for *kernel=rbf* and *C=4* through the use of grid search. After that, the best SVM was trained on the whole development set and used to predict evaluation set. The public score obtained by the model was 83%.

Figure 7 and 8 depict the CNN training accuracy and loss

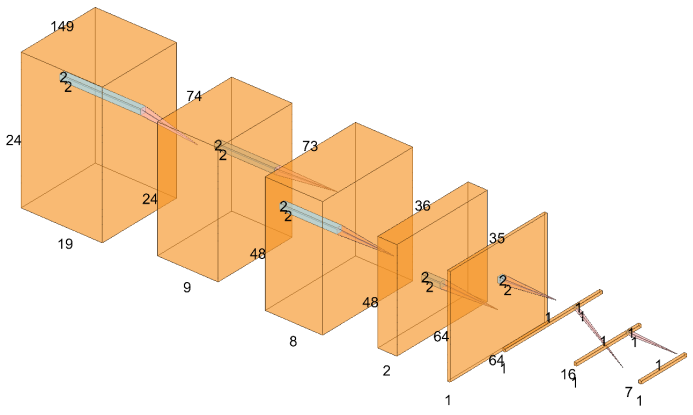

Fig. 5. Detailed Structure of the CNN

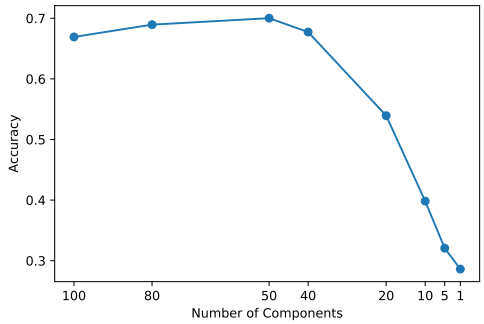

Fig. 6. Tuning the Number of Components for the PCA Algorithm

for the best-found configuration. The details of the best CNN is shown in Figure 5. The public score achieved by the CNN was 87%.

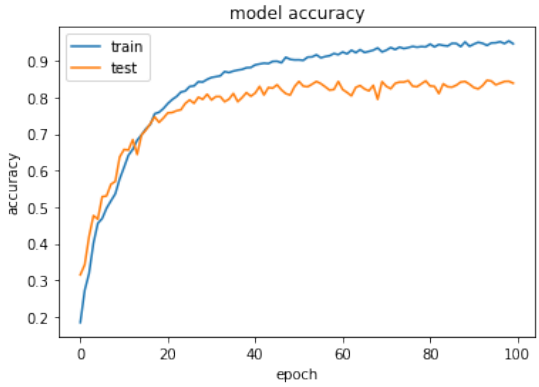

Fig. 7. CNN Accuracy on Development Data

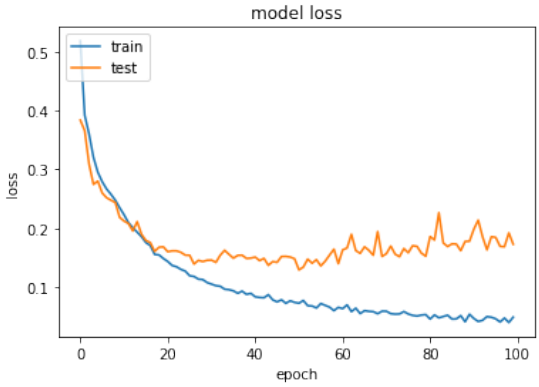

Fig. 8. CNN loss on Development Data

IV. Discussion

There are some aspects that might improve the obtained results

- In the proposed approach, features showing characteristics of the people are removed, but applying feature extraction methods or some transformations to them might make the results better
- Applying a denoising algorithm in preprocessing phase can contribute to the improvement of the results.
- Using Recurrent Neural Networks (RNN) and Long Short-Term Memory Networks (LSTM) may be considered since they are models specialized for processing sequential data. Also, they can keep a state evolving over time.
- Tokenizing is the process of dividing speech into smaller units, called tokens, for more efficient and accurate processing. Tokenization is applied to the audio signal in order to divide it into segments, typically phonemes or subwords, for more effective analysis. The goal of tokenization is to divide the input speech into meaningful units that can be used for feature extraction and modeling. This process helps reduce the complexity of the speech signal and facilitates the implementation of speech recognition algorithms.

References

[1] K. S. Rao and K. Manjunath, *Speech recognition using articulatory and excitation source features*. Springer, 2017.
[2] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.
[3] K. Palanisamy, D. Singhania, and A. Yao, "Rethinking cnn models for audio classification," *arXiv preprint arXiv:2007.11154*, 2020.
[4] P. Dhanalakshmi, S. Palanivel, and V. Ramalingam, "Classification of audio signals using svm and rbfnn," *Expert systems with applications*, vol. 36, no. 3, pp. 6069–6075, 2009.