Speaker Recognition via Nearest Neighbors Methods using Audio Features

Emmanouil Lykos
University of Piraeus
NCSR Demokritos
manolislykos97@hotmail.gr

June 21, 2022

Abstract

Speaker recognition is a pretty demanding task with many real-world applications, which has to do with the techniques used to identify a speaker from audio or a video. Those systems can be used in different assistive technologies or to just annotate movies or podcasts, and they even can be expanded to the field of verification. In this paper, we will present a technique that its method is pretty easy to understand but it gives to the reader a clear intuition on how it works. Also, because this method cannot work with raw samples we will present two Feature Extraction techniques that we will use in order to extract the feature vectors from each sample. Finally, we will present how those features perform against our method and we will explain the results and also cite some directions in order to achieve better predictive performance.

Keywords: Speaker Identification, k-Nearest Neighbors, Machine Learning, Feature Extraction

1 Introduction

Speaker recognition is the task of identifying who speaks, given an audio or a video segment. According to [1], the aforementioned task has many real-world applications. Initially, along with Automatic Speech Recognition, it can be used to add the extra information on shows' subtitles on who is speaking which will be pretty useful for deaf people that solely rely on their vision when watching some shows. Finally speaker recognition systems can be used for security and surveillance reasons.

The particular task falls into the area of supervised learning, and more specifically to classification, because in order to train a model that exceeds to this task we need to provide labelled data with discrete classes, i.e. assign a particular class(in our case a speaker) to each of the given instances. Also, this task can be formulated more properly as an online open-set classification one as it is done on[2], because when this type of systems will be used into production, it is natural that not all speakers in the world will be present to the train dataset, hence, it would be more appropriate for the system to not only be able to tell that the given speaker is not contained in the database but also work in an online fashion by updating its model to be able to recognize the new speaker in the future. However, I will not deal with the open-set methods in this paper.

In this project, I will deal, initially, with various audio feature extraction techniques that can be applied to our dataset. Afterwards, I will apply and evaluate them against various Machine Learning algorithms but my research on that will be based mostly on k-Nearest Neighbors[3]

method. The structure of the paper is the following. In Section 2, I will introduce some background knowledge needed and our proposed methodology. Afterwards in Section 3, I will show the experimental setting and the results that the proposed method yields. Finally, in Section 4, I will write my conclusions from the research done and also provide directions for further research.

2 Methodology

2.1 Audio Features

Before proceeding to the proposed method of the proposed system, we need to define the types of features that I will feed into the system in order to evaluate my method. The former ones, are derived from pyAudioAnalysis library[4] and are the short term ones, i.e. a pack of both time-domain and spectral features like energy, Spectral Centroid etc. They are called short term because they are extracted by splitting the signal into some windows and then we calculate each feature on each window. However, because our model cannot handle sequential data, like an LSTM for example, we will pass the average vector that was derived by calculating the average value of each feature through each window of a specific segment. The dimensionality of those vectors will be equal to 64. The latter ones, are derived by taking a pre-trained Wav2Vec2[5] model. The Wav2Vec2 model follows a virtually similar transformer architecture as BERT[6] with difference that instead of passing the word embedding along with positional embedding, I extract the features of each window by passing them from some Convolutional Neural Network. However, for Sequence Classification Tasks in Natural Language Processing only the first output representation is consider down the line in the model, I will not do the same but I will take the average of all the representations because that approach gave me better results by a wide margin. The dimensionality of those vectors is equal to 1024.

2.2 k-Nearest Neighbors Algorithm for Speaker Identification

Now that I stated the different types of features that I'll work on, I will present the system's method. The method that the proposed system uses, is the k-Nearest Neighbors[3] using Cosine Similarity as it seems that yields better results than Euclidean Distance. k-Nearest Neighbors classification algorithm is proposed because it may not be the best like Neural Network approaches, but it is more intuitive and easier to explain why some predictions were done. The algorithms is pretty straightforward and the prediction of the given sample consists of the following steps:

1. Find the k nearest neighbors of the given sample, according to the given distance metric.
2. The prediction is the more prevalent label among those k neighbors.

3 Experimental Evaluation

In order to perform the experiments I should prepare the given data. The only preprocessing that I do is to remove the constant features, i.e. features that given some instances they have very small standard deviation in the dataset and afterwards I normalize the dataset with sci-kit Learn[7] MinMaxScaler method. Afterwards, I evaluate the approach by using different number of neighbors each time via Cross Validation. The metric that I use is the accuracy one, i.e the percent of samples that my classifier predicted correctly.

The dataset used is the VoxCeleb1[8] one, which contains speech samples of 1251 celebrities. However due to computational and resource limitations, the experiments were performed by taking a subset of that dataset that contains all the samples of each celebrity, but for only 250 of those. The experimental results are the following:

Neighbors	Short-Term Features		Wav2Vec2 Features	
	train	**test**	**train**	**test**
1	**1.0000**	**0.5484**	1.0000	0.3659
5	0.6851	0.5173	0.5448	0.3027
10	0.6263	0.5068	0.4933	0.3010
25	0.5268	0.4634	0.3922	0.2717
50	0.4507	0.4132	0.3202	0.2433
100	0.3740	0.3547	0.2495	0.2000
200	0.3025	0.2910	0.1844	0.1632
350	0.2459	0.2440	0.1412	0.1323

Table 1: Train and Test Accuracies of the proposed System Identifier in respect to the number of neighbors and the audio features that I use.

As we can see from Table 1, the best k-Nearest Neighbors model is the one with $k = 1$ that uses Short-Term Features because it has 54% accuracy and it is better by approximately 3% from the second best. However, from the results I observed two things. Firstly, we can see that Short-Term features have around 20% better accuracy. To be honest, I did not expect that wide margin in the results however this might be the case because Wav2Vec2 was used mostly for Speech Recognition, thus, the extracted embeddings might not be appropriate for speaker identification because two instances that are near each other according to Wav2Vec2 might mean something different that the speakers of those instances is the same. Another reason can be that those kind of embeddings could not be show their potential using k-Nearest Neighbors and probably need to use another one like Support Vector Machines(SVMs)[9]. But, I tried to train an SVM classifier also, but still the Wav2Vec2 performed poorly. Secondly, I was expecting personally to get the best results from a middle value like 25 or 50, but instead I got it when I take into consideration the nearest neighbor. This might probably happen because there might be a case that for $k > 1$ we can have too many "ties" in the voting, thus, the classifier might do so many random choices that-in most times-they will be wrong.

4 Conclusion & Further Work

In this paper, we presented a Speaker Identification method that is based in essence in different Feature Extraction techniques from audio and k-Nearest Neighbors algorithm. The types of features that we tested our system was the short-term features that are extracted from pyAudioAnalysis library and Wav2Vec2 features. Although this approach seems to be pretty simple we saw from the Experimental Results, that with the correct features we can achieve a pretty decent performance because our best model has 54% accuracy on 250 speakers, where each speaker has like 100 samples. In terms of further work we can follow two paths. On the one hand, we can find better feature extraction techniques, like another pre-trained model or create an autoencoder in order to have variable sized vectors for each sample. On the other hand, we can focus on Neural Network approaches, where we can use LSTMs in order to pass the sequence of those features in each sample and then make predictions or Convolutional Neural Networks, where we can pass the spectrogram of each sample.

References

[1] Nilu Singh, Prof. Raees Khan, and Raj Shree Pandey. "Applications of Speaker Recognition". In: *Procedia Engineering* 38 (Dec. 2012), pp. 3122–3126. DOI: 10.1016/j.proeng.2012.06.363.

[2] Hemant Kekre and Vaishali Kulkarni. "Closed set and open set Speaker Identification using amplitude distribution of different Transforms". In: Jan. 2013, pp. 1–8. ISBN: 978-1-4673-5618-3. DOI: 10.1109/ICAdTE.2013.6524764.

[3] Pádraig Cunningham and Sarah Jane Delany. "k-Nearest Neighbour Classifiers - A Tutorial". In: *ACM Computing Surveys* 54.6 (July 2021), pp. 1–25. ISSN: 1557-7341. DOI: 10.1145/3459665. URL: http://dx.doi.org/10.1145/3459665.

[4] Theodoros Giannakopoulos. "pyAudioAnalysis: An Open-Source Python Library for Audio Signal Analysis". In: *PloS one* 10.12 (2015).

[5] Alexei Baevski et al. *wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations*. 2020. DOI: 10.48550/ARXIV.2006.11477. URL: https://arxiv.org/abs/2006.11477.

[6] Jacob Devlin et al. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. 2018. DOI: 10.48550/ARXIV.1810.04805. URL: https://arxiv.org/abs/1810.04805.

[7] Fabian Pedregosa et al. "Scikit-learn: Machine Learning in Python". In: (2012). DOI: 10.48550/ARXIV.1201.0490. URL: https://arxiv.org/abs/1201.0490.

[8] Arsha Nagrani, Joon Son Chung, and Andrew Zisserman. "VoxCeleb: A Large-Scale Speaker Identification Dataset". In: *Interspeech 2017*. ISCA, Aug. 2017. DOI: 10.21437/interspeech.2017-950. URL: https://doi.org/10.21437%2Finterspeech.2017-950.

[9] Bernhard Boser, Isabelle Guyon, and Vladimir Vapnik. "A Training Algorithm for Optimal Margin Classifier". In: *Proceedings of the Fifth Annual ACM Workshop on Computational Learning Theory* 5 (Aug. 1996). DOI: 10.1145/130385.130401.