Speaker Recognition Using Sparse Representation via Superimposed Features

Yashesh Gaur, Maulik C. Madhavi, and Hemant A. Patil

Dhirubhai Ambani Institute of Information and Communication Technology (DA-IICT), Gandhinagar, Gujarat, India
{yashesh_gaur,madhavi_maulik,hemant_patil}@daiict.ac.in

Abstract. In this paper, we demonstrate the effectiveness of *superimposed features* for the purpose of template matching-based speaker recognition using sparse representations. The principle behind our hypothesis is, if the test template approximately lies in the *linear span* of the training templates of the genuine class, then so does any linear combination of test templates. In this paper, we introduce the notion of *superimposed features* for the first time. Using our initial trials on the TIMIT database, we have shown that *superimposed features* can result in reducing the complexity cost by *80* % with a very minor decrease in identification rate by *0.67* % and a minor increase in EER by *0.85* %.

Keywords: Superimposed features, sparse representations, orthogonal matching pursuit, template matching, speaker recognition.

1 Introduction

Speaker recognition is the task to recognize a person from his or her voice, with the help of machines. Depending on the way feature matching is done, the systems can be classified into template matching systems and probabilistic modelling systems [1]. Probabilistic modelling systems involve modelling feature vectors with *probability density functions* (pdf). The probability of a test utterance, given the speaker model, is evaluated to get the *confidence scores* [1]-[2]. The template matching techniques, on the other hand, do not involve any probabilistic measures. The features from the test utterances are considered as some variation of the training features [1]. The template matching-based techniques are usually faster as no probabilistic modelling is required prior to matching. A sparse representation for the purpose of pattern classification has been used in [3]-[5]. A sparse representation model for the speaker recognition was used in [6]. This technique was probabilistic in nature as they used Gaussian Mixture Model (GMM) mean super vectors to model speaker characteristics. Sparse representations were invoked after the speaker characteristics were modelled using GMM. Recently, sparse representations were used using template matching technique in [7]. The benefit of this kind of technique is the reduction in complexity as sparse representations can directly be used on the features, without any prior modelling. The work also uses Orthogonal Matching Pursuit (OMP)[8] for the sparse recovery of the weight vector.

However, due to large amount of features in a test or training utterance, this approach sometimes becomes computationally ineffective as sparse recovery has to be performed for each and every test feature vector. In this paper, we propose a new method for speaker recognition using sparse representations, via *superimposed features*, which substantially reduces the computational complexity of such a system. The main assumption behind our approach is, if the testing template lies in the linear span of the training templates, then so does any linear combination of the testing templates. Thus, superimposed features can be considered as test features. This simple but powerful insight can help us in making real-time template matching systems for speaker recognition which are computationally much less expensive, with a very little degradation in performance.

The rest of the paper is organized as follows: The sparse representation framework, sparse recovery using OMP, evaluation of confidence scores and superimposed features are discussed in Section *2*. In Section *3* and Section *4*, we discuss the experimental setup and results, respectively. Finally, Section *5* concludes the work along with future research directions.

2 Sparse Representation Framework and Superimposed Features

2.1 Sparse Representation Framework

Suppose that each speaker is being evaluated against K speakers and each speaker has a set of N training features, which are m-dimensional. Let us define

$$A_k = [\mathbf{a}_{k1}, \mathbf{a}_{k2}, .., \mathbf{a}_{kN}] \in \mathbb{R}^{m \times N}, \tag{1}$$

as a $m \times N$ matrix of all the training features of k^{th} speaker concatenated. Here, $\mathbf{a}_{kn}$ represents the feature vector extracted from n^{th} frame of the k^{th} speaker. A universal dictionary can thus be created by concatenating such matrices from all the K speakers. That is,

$$A = [A_1, A_2, .., A_K] \in \mathbb{R}^{m \times K \cdot N}. \tag{2}$$

Now, consider a feature vector $\mathbf{y}$, extracted from a test utterance of some speaker. One can express this feature vector as an approximate linear combination of columns of matrix $\mathbf{A}$ as

$$\mathbf{y} \approx \sum_{k=1}^{K} \sum_{n=1}^{N} x_{kn} \mathbf{a}_{kn}, \tag{3}$$

where x_{kn} is the weight associated with the column a_{kn}. This can be written in a more compact form as

$$\mathbf{y} = A\mathbf{x} + \mathbf{n}, \tag{4}$$

where the vector $\mathbf{x}$ contains the weights corresponding to the columns of matrix A, $\mathbf{n}$ is the *noise vector* which accounts for the approximation in (3) and also the noise present in the measurements. If $\mathbf{y}$ belongs to a particular class, say the k^{th} class, then it can be said that it will approximately lie in the linear span of the training vectors of the k^{th} class [6],[7]. In other words, the test vector of the k^{th} class can be represented as a linear combination of the training vector of the k^{th} class. In that case, the weight vector should exhibit high sparsity since most of the weights corresponding to vectors of other classes will be zero and only vectors from the k^{th} class will have non-zero weights [6],[7].

To find the weights corresponding to the columns of A, we need to solve the system of linear equations $\mathbf{y} = A\mathbf{x}$. Since $m \ll K \cdot N$, this is an underdetermined set of equations and there are infinitely many solutions to this system. However, we know that the weight vector needs to be highly sparse, therefore, we need to choose the sparsest solution among these infinitely many solutions. In other words,

$$\underset{\mathbf{x}}{min} \, \|\mathbf{x}\|_0 \text{ subject to } \mathbf{y} = A\mathbf{x}. \tag{5}$$

This optimization problem is an NP hard problem [9]. However, several greedy algorithms like matching pursuit [8], Orthogonal Matching Pursuit (OMP) [8], etc. have been proposed to solve this system. In this work, we will be using OMP for the sparse recovery of the weight vector $\mathbf{x}$.

2.2 Sparse Recovery Using Orthogonal Matching Pursuit (OMP)

A brief description of the OMP algorithm is given in Box B.1. The algorithm with its performance guarantees and details can be found in [8].

Box B.1. OMP Algorithm

```
Initialize: $\hat{\boldsymbol{x}}_0 = 0$, $\boldsymbol{r}_0 = \boldsymbol{y}$ and $\Lambda_0 = \Phi$
for $i = 1$; $i := i + 1$ until stopping criterion is met,
Do
Form signal estimate from residual: $\boldsymbol{g}_i \longleftarrow A^T \boldsymbol{r}_{i-1}$
Add largest residual entry to support: $\Lambda_i \longleftarrow \Lambda_{i-1} \cup supp(max(\boldsymbol{g}_i))$
Update signal estimate using least squares: $\hat{\boldsymbol{x}}|_{\Lambda_i} = A^{\dagger}_{\Lambda_i} \boldsymbol{y}|_{\Lambda_i^c \longleftarrow 0}$
Update measurement residual: $\boldsymbol{r}_i = \boldsymbol{y} - A\hat{\boldsymbol{x}}_i$
End for
```

This algorithm takes the test vector $\mathbf{y}$, the matrix A and returns the approximated weight vector $\hat{\mathbf{x}}$. In each iteration, it selects the column in A which has the highest correlation with the residual, adds the column index to the set Λ, gets the signal estimate by applying least squares to the columns of A indexed in the set Λ, and updates the residual for the next iteration. The stopping criterion used here is,

$$\|\mathbf{r}_i\|_2 < \lambda \|\mathbf{y}\|_2. \tag{6}$$

Therefore, whenever the norm of the residue goes below a certain fraction of $\|\mathbf{y}\|_2$, the algorithm stops. Heuristically, we found that $\lambda = 0.1$ best suits the requirements of this sparse recovery. It should be noted that this value will differ for different scenarios. The value of λ must be chosen carefully, since a high value will not be able to capture the contributions of enough vectors, whereas a very small value may result into OMP running unnecessarily for a longer duration.

2.3 Confidence Scores

Suppose, we are given a frame extracted from a testing utterance. We can find the weight vector $\hat{\mathbf{x}}$, using OMP algorithm as described in the previous section. This vector will be highly sparse as only a few vectors from the genuine class will contribute to this test frame. The contribution of each speaker to the test frame can be calculated using class-based residual [3],[6],[7]. The class-based residual for k^{th} speaker can be computed by retaining the weights corresponding to the k^{th} class, and putting all other weights to zero. This can be done by defining a function $\delta_k(\hat{\mathbf{x}})$as:

$$\delta_k(\hat{\mathbf{x}}) = [0 \dots 0 | 0 \dots 0 | \dots | \hat{x}_{k1} \dots \hat{x}_{kN} | \dots | 0 \dots 0] . \tag{7}$$

The normalized class-based residual error can be computed as:

$$\mathbf{r}_k(y) = \frac{\|\mathbf{y} - A\delta_k(\hat{\mathbf{x}})\|_2}{\|\mathbf{y}\|_2} . \tag{8}$$

The normalized class based residual error always lies between *0* and *1*. A residual error closer to *0* indicates a close match. The confidence score for k^{th} speaker for test frame $\mathbf{y}$ can be computed as:

$$c_k(\mathbf{y}) = exp\left(-r_k(\mathbf{y})\right) . \tag{9}$$

This is the confidence score of the k^{th} speaker, for a single frame $\mathbf{y}$, extracted from the test utterance. Suppose there are Z frames in the testing utterance. Confidence scores are calculated for all the Z frames and a mean confidence score is generated for the k^{th} speaker. That is,

$$C_k = \frac{1}{Z} \sum_{i=1}^{Z} c_k\left(\mathbf{y}_i\right) . \tag{10}$$

This process can be repeated for all the K speakers to get speaker-specific confidence scores for the given test utterance.

2.4 Superimposed Features

If we have a frame from a test utterance belonging to the k^{th} class, it approximately lies in the linear span of the training vectors of that class [6],[7]. Let $\mathbf{y}_1$ and $\mathbf{y}_2$ be two frames which come from a testing utterance of k^{th} speaker. Both $\mathbf{y}_1$ and $\mathbf{y}_2$ can be approximately written as a linear combination of training vectors of the k^{th} class.

$$\mathbf{y}_1 \approx \alpha_{11}\mathbf{a}_{k1} + \alpha_{12}\mathbf{a}_{k2} + \dots + \alpha_{1N}\mathbf{a}_{kN} , \tag{11}$$

$$\mathbf{y}_2 \approx \alpha_{21}\mathbf{a}_{k1} + \alpha_{22}\mathbf{a}_{k2} + \cdots + \alpha_{2N}\mathbf{a}_{kN}, \tag{12}$$

$$\mathbf{y}_1 + \mathbf{y}_2 \approx (\alpha_{11} + \alpha_{21})\,\mathbf{a}_{k1} + (\alpha_{12} + \alpha_{22})\,\mathbf{a}_{k2} + \cdots + (\alpha_{1N} + \alpha_{2N})\,\mathbf{a}_{kN}. \tag{13}$$

It follows that, any combination of $\mathbf{y}_1$ and $\mathbf{y}_2$ should also approximately lie in the linear span of training vectors of the k^{th} class. Let us call the combination of $\mathbf{y}_1$ and $\mathbf{y}_2$ as the '*superimposed vector*'. If this '*superimposed vector*' is used to calculate the weights, the weight vector $\hat{\mathbf{x}}$ will still be *sparse*, as ideally, only the vectors from the genuine class should contribute to the test frame (13). This weight vector after class-based residual should give identical results, as in the case of $\mathbf{y}_1$ or $\mathbf{y}_2$ alone. The entire argument can be extended for superposition of more than two frames as well. Let us call any such combination of multiple test frames as the '*superimposed vector*' and define '*level of superposition*' as the number of feature vectors that are superimposed.

One can see that, using a superimposed feature, with level of superposition *'L'*, implies that we are clubbing L features together and feeding it to OMP. Consequently, the number of times OMP is invoked is reduced by L times. This in turn reduces the computational complexity of the system, and can help us in making such systems real time. It should be noted that, though the weight vector is still sparse, its sparsity will change after superimposed test frames are given. OMP can be similarly used for the sparse recovery of the weight vector, however, the stopping criterion on the residual may need to be changed. For our experiments, we have used the same threshold for stopping OMP.

3 Experimental Setup

3.1 Speech Corpus

The proposed approach is applied on all the *630* speakers in the TIMIT corpus. In the corpus, each speaker has *10* waveforms, which are divided into '*sx*', '*sa*' and '*si*' sections, containing *5*, *2* and *3* files per speaker, respectively [10]. The features, from the waveforms in the section '*sx*' and '*sa*' are taken for training purpose, whereas the features from the waveforms in '*si*' section are taken for testing purposes. Thus, there are *630* test trials. Each test is evaluated against *20* speakers, which are randomly selected. However, the correct speaker is always present in the *20* claimants.

3.2 Feature Extraction

Mel-Frequency Cepstral Coefficients (MFCC) [11] were used as features to represent speaker-specific characteristics. The speech signal is pre-emphasized and then divided into frames of *20* ms with an overlap of *10* ms. Each frame is first multiplied by a Hamming window and then *26* Mel-scaled triangular filters are used to extract a *19*-dimensional MFCC vector. This vector is appended by delta cepstral coefficients which are computed over a span of ±*2* frames, thus producing a *38*-dimensional feature vector. Voice Activity Detector (VAD) is also used to discard frames with low energy. Cepstral Mean Subtraction (CMS) and normalization were also performed on the feature vectors to cancel the channel effects.

3.3 Performance Measures

The performance of the proposed technique can be tested using various parameters. We have used Equal Error Rate (EER) [12], % Identification rate (%ID) and computational complexity as the standard measures for performance analysis.

3.3.1. % Equal Error Rate (EER)

The confidence score obtained by evaluating test speaker with genuine speaker is called as *true* (genuine) score and all other confidence scores are called *impostor* scores. For speaker verification, an operating threshold score can be set for making decisions. In the decision making process, errors are encountered and the system may face *2* types of errors, *viz.*, false acceptance (false alarm) and false rejection (miss detection). For one particular operating threshold, both the errors are equal and the corresponding error value (false alarm or miss detection) is called as Equal Error Rate (EER). % EER is very useful evaluation metric in the speaker verification task [12].

3.3.2. % Identification Rate

Let N_C be the total number of correct hits (i.e., number of correctly identified speakers) using the algorithm and N_T be total number of test trials (i.e., total number of speakers). % Identification (%ID) rate is defined as following.

$$\%ID = \frac{N_C}{N_T} \times 100. \tag{14}$$

3.3.3. Computational Complexity

Computational complexity is one of the most important constraints when real-time systems are deployed. The major complexity step in the mentioned scheme is the OMP step. Therefore, the computational complexity can either be measured by number of OMP invocations or the average time it takes to process one query. All experiments were conducted on *Intel Xeon (R), CPU E5-2420 @ 1.90 GHz* machine.

4 Experimental Results

Table 1 gives a detailed description of the effect of various levels of superposition on the performance of system.

It is evident from Table 1 that with increase in level of superposition, there is a very minor deterioration in the values of % EER and % ID. Comparing to the

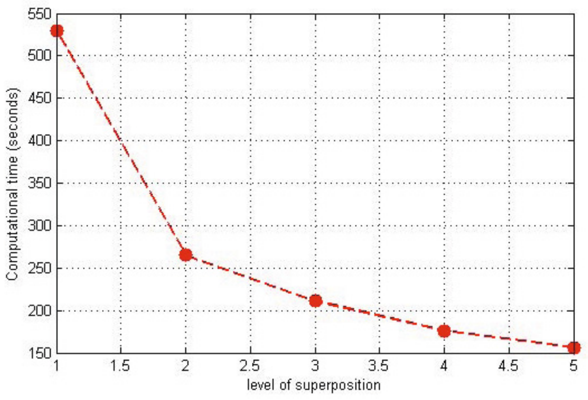

Fig. 1. Average execution time per query, in seconds, for different level of super position

Table 1. % EER, % ID and execution time for different levels of superposition on TIMIT database

Level of superposition	% EER	% ID	Execution time (sec)
1	0.45	99.84	530
2	0.45	99.84	265
3	0.7	99.67	211
4	0.97	99.51	176
5	1.3	99.17	156

baseline results, superposition level *5* results in a minor EER increase of *0.85* % and the % ID decreases by just *0.67* %. On the other hand, the numbers of OMP invocations are reduced to one-fifth of the original value. Consequently, the execution time per query is also reduced, which is also depicted by Figure 1. Hence, one can significantly reduce the computational cost of the system, with a little deterioration in EER and identification rate. This experiment can be conducted for higher level of superposition as well. It is natural that the % ID rate and EER performance will be deteriorated as level of superposition increases. This is because, the approximation depicted in (13) grows more and more inaccurate with increasing level of superposition. Therefore, a trade-off exists between computational complexity and EER and % ID rate performance. If one goes on increasing the level of superposition, one will reach a point when the EER and % ID performance is no longer acceptable. That should be the optimal level of superposition for the system. The optimal level of superposition will differ from system to system and has to be experimentally determined. In this paper, we have considered $\%ID > 99$ % as acceptable performance. Thus, we have limited ourselves to superposition level of *5*.

5 Summary and Conclusions

In this paper, we have proposed a new technique for speaker recognition, using sparse representations via superimposed features, which greatly reduce the complexity cost with a minor deterioration in EER and % ID rate performance. However, the proposed method does not take into account the increased sparsity of the weight vector when superimposed vectors are fed to OMP. Presently, we are running OMP using the same stopping criterion every time. OMP might need to run a few more iterations, as the sparsity has been decreased. Work can be done which adaptively determines the number of iterations needed for a particular level of superposition. The robustness of this scheme can also be checked against noise with varying SNR.

Acknowledgments. The authors would like to thank Department of Electronics and Information Technology (DeiTy), Govt. of India for their support to carry out this work.

References

1. Campbell Jr., J.P.: Speaker recognition: a tutorial. Proc. of the IEEE 85(9), 1437–1462 (1997)
2. Hazen, T., et al.: Multi-modal Face and Speaker Identification on a Handheld Device. In: Proc. Wkshp. Multimodal User Authentication, pp. 120–132 (2003)
3. Wright, J., et al.: Robust face recognition via sparse representation. IEEE Trans. on Pattern Analysis and Machine Intelligence 31(2), 210–227 (2009)
4. Pillai, J.K., et al.: Secure and Robust Iris Recognition Using Random Projections and Sparse Representations. IEEE Trans. on Pattern Analysis and Machine Intelligence 33(9), 1877–1893 (2011)
5. Yang, A.Y., et al.: Distributed recognition of human actions using wearable motion sensor networks. J. of Ambient Intelligence and Smart Environments 1(2), 103–115 (2009)
6. Naseem, I., Togneri, R., Bennamoun, M.: Sparse Representation for Speaker Identification. In: 20th Int. Conf. on Pattern Reco. (ICPR), pp. 4460–4463 (2010)
7. Boominathan, V., Sri Rama Murty, K.: Speaker recognition via sparse representations using orthogonal matching pursuit. In: Int. Conf. on Acoustics, Speech and Signal Process. (ICASSP), pp. 4381–4384 (2012)
8. Elad, M.: Sparse and Redundant Representations. Springer, New York (2009)
9. Zucker, S.W., Leclerc, Y.G., Mohammed, J.L.: Continuous Relaxation and Local Maxima Selection: Conditions for Equivalence. IEEE Trans. on Pattern Analysis and Machine Intelligence, PAMI 3(2), 117–127 (1981)
10. Garofolo, J.S.: Getting started with the DARPA TIMIT CD-ROM: An acoustic phonetic continuous speech database. National Institute of Standards and Technology (NIST), Gaithersburgh, MD (1988)
11. Davis, S., Mermelstein, P.: Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. IEEE Transactions on Acoustics, Speech and Signal Process. 28(4), 357–366 (1980)
12. Martin, A., et al.: The DET Curve in Assessment of Detection Task Performance. In: Proc. Eurospeech 1997, vol. 4, pp. 1899–1903 (1997)