# Cosine Distance Metric Learning for Speaker Verification Using Large Margin Nearest Neighbor Method

Waquar Ahmad, Harish Karnick, and Rajesh M. Hegde

Department of Electrical Engineering,
Indian Institute of Technology Kanpur
rhegde@iitk.ac.in
http://202.3.77.107/mips/

**Abstract.** In this paper, a novel cosine similarity metric learning based on large margin nearest neighborhood (LMNN) is proposed for an i-vector based speaker verification system. Generally, in an i-vector based speaker verification system, the decision is based on the cosine distance between the test i-vector and target i-vector. Metric learning methods are employed to reduce the within class variation and maximize the between class variation. In this proposed method, cosine similarity large margin nearest neighborhood (CSLMNN) metric is learned from the development data. The test and target i-vectors are linearly transformed using the learned metric. The objective of learning the metric is to ensure that the k-nearest neighbors that belong to the same speaker are clustered together, while impostors are moved away by a large margin. Experiments conducted on the NIST-2008 and YOHO databases show improved performance compared to speaker verification system, where no learned metric is used.

## 1 Introduction

The task of speaker verification is to verify that the input utterance is by the claimed speaker. This requires checking whether the input utterance belongs to the claimant speaker or not [1]. I-vector, which stands for identity vector, represents an utterance by a fixed, low dimensional vector [2]. I-vectors are compared by calculating the dot product of the i-vectors (cosine similarity), which gives the similarity score between the i-vectors. Based on this score a verification decision is made for the input utterances. This method outperforms the one based on the Gaussian Mixture Model - Universal Background Model (GMM-UBM) [3]. Distance metric learning methods are used in machine learning to improve a system's performance by learning a metric from example data [4,5]. The learned metric transform the input feature space into a new feature space, where the separability between the data is modified such that data belong to same class are moved closer and data belong to different classes are moved away. We use the cosine similarity large margin nearest neighborhood (CSLMNN) metric that is learned from the development data. The metric calculation constrains the k

W.T. Ooi et al. (Eds.): PCM 2014, LNCS 8879, pp. 294–303, 2014.

nearest neighbors to belong to the same speaker, while other speakers are separated by a large margin [6]. The original LMNN algorithm [6] is learned by calculating the Euclidean distance between the data. As cosine similarity gives competitive results in state of art speaker verification system, we use cosine similarity between the data to learn the metric. The problem of metric learning is formulated as an instance of a semidefinite programming problem to efficiently compute the global minima.

The rest of the paper is organized as follows. Section 2 introduces the general framework used for cosine distance metric learning for i-vector based speaker verification. In that section, i-vector extraction and classification is discussed which is followed by the discussion of cosine similarity large margin nearest neighborhood metric learning. Section 3 describes the experiments conducted on NIST 2008 and YOHO databases using the CSLMNN metric in an i-vector framework. The performance of the system is evaluated using the detection error trade-off (DET) curves, equal error rate (EER) and minimum decision cost function (DCF) points. Section 4 concludes with a discussion of the results obtained and some thoughts on distance metric learning in i-vector based speaker verification.

## 2 Cosine Distance Metric Learning for I-Vector Based Speaker Verification

In this section, we discuss the cosine distance metric learning based on large margin nearest neighborhood method. As cosine distance gives very competitive performance in i-vector based speaker verification system, we choose cosine similarity as a distance measure in the proposed metric learning method. The learned metric transform the input i-vectors into a new feature space, where the i-vectors that belong to same speaker are moved closer and impostors are moved away. Scoring on this new transformed space of i-vector gives better performance than the previous i-vector space. Following subsection discusses the i-vector extraction and classification method used in current state of art speaker verification system with a brief discussion of linear discriminant analysis as inter session compensation and dimensional reduction technique.

### 2.1 I-Vector Extraction and Scoring

In this subsection, we briefly discuss the i-vector extraction and scoring for speaker verification system. I-Vector is a compact form of speech utterance that has been extracted using the total variability subspace [2]. It involves formation of GMM supervector by concatenating the *means* of the MAP adapted speaker model from the UBM model. The supervector is assumed to have the following structure.

$$s = m + Tw \tag{1}$$

Where $m$ is the speaker independent UBM supervector, $T$ is the total variability matrix and $w$ the total variability factor that is termed as the i-vector. Matrix

$T$ is computed from the training data in exactly the same way as the eigenvoice matrix in the $JFA$ system with the slight difference that the speech for training belongs to different speakers. Matrix $T$ is a low rank matrix. For the given matrix $T$, the i-vector $w$ is obtained for the given utterance by:

$$w = \left(I + T^t \Sigma^{-1} N T\right)^{-1} T^t \Sigma^{-1} F \tag{2}$$

where $I$ is an identity matrix and $N$ is a diagonal matrix of dimension $CF \times CF$, its diagonal block are $N_c I$, $(c = 1, 2, ..C)$ is the Gaussian index and $F$ is the supervector formed by concatenating all the centralized first order statistics. $\Sigma$ is a diagonal covariance matrix of dimension $CF \times CF$. The block diagram Figure 1 illustrates the procedure for extracting the i-vector.

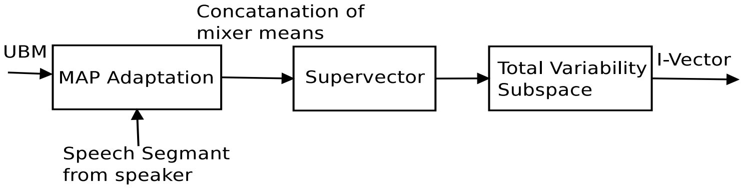
Concatanation of mixer means
UBM
MAP Adaptation
Supervector
Total Variability Subspace
I-Vector
Speech Segmant from speaker

**Fig. 1.** Diagram illustrating the i-vector extraction from a speech dataset

**Linear Discriminant Analysis.** Linear Discriminant Analysis (LDA) is a popular technique to compensate for inter-session variability in the speech data. The main objective of LDA is to find new orthogonal axes such that variation between classes is maximized and within classes is minimized. The LDA transformation matrix $A_{lda}$ consists of the eigenvectors having the largest eigenvalues for the eigenvalue problem $S_B v = \lambda S_W v$, where the between and within speaker scatter matrices, $S_B$ and $S_W$ respectively are calculated using:

$$S_B = \sum_{s=1}^{S} N_s \left(\mu_s - \mu\right) \left(\mu_s - \mu\right)^t \tag{3}$$

$$S_W = \sum_{s=1}^{S} \sum_{i=1}^{N_s} \left({w_i}^s - \mu_s\right) \left({w_i}^s - \mu_s\right)^t \tag{4}$$

nist2002 In the above formula $\mu_s$ is the mean i-vector of each speaker, $S$ denotes the total number of speaker under consideration and $N_s$ stands for the total number of utterances for speaker $S$. The matrix $A_{lda}$ is calculated as follows:

$$A_{lda} = \arg\max_{A} \frac{|A^T S_B A|}{|A^T S_W A|} \tag{5}$$

Following this the corresponding i-vector $w$ is transformed into the vector $w'$ as follows:

$$w' = w * A_{lda} \tag{6}$$

**I-Vector Scoring.** The cosine score between the test $w'_{test}$ and target $w'_{target}$ i-vectors is given by the dot product between these two i-vectors,

$$Score(w'_{target}, w'_{test}) = \frac{\langle w'_{target}, w'_{test} \rangle}{\|w'_{target}\| \|w'_{test}\|} \tag{7}$$

The score obtained is then compared with a threshold value and an accept decision is taken if the score is above the threshold else it is rejected. Both the target and test i-vectors are estimated exactly in the same manner with the same UBM and the same total variability matrix $T$.

### 2.2 Cosine Similarity Large Margin Nearest Neighbor Metric Learning

Distance metric learning methods are used in classification problems to improve the performance of the system by learning a metric from example data [7,8]. This leads to better discriminability between the input data and helps to improve the performance of the classification system. Let us consider the training set of $n$ examples of dimensionality $d$, $\{(w_i, w_j)\}^n{}_{i=1}$, where $w_i = R^d$ and $w_j \in \{1, 2, 3....S\}$. Here $S$ is the total number of classes (speakers). In general, the similarity score between two inputs $w_i$ and $w_j$ is given by the equation.

$$Score(w'_{target}, w'_{test}) = \frac{\langle Mw'_{target}, Mw'_{test} \rangle}{\|Mw'_{target}\| \|Mw'_{test}\|} \tag{8}$$

Where $M$ is a symmetric positive definite matrix, $M \succeq 0$. This metric is learned by the CSLMNN algorithm from the data.

The non differentiable leave-one-out and non-continuous classification error of the k nearest neighbor classifier is imitated by the CSLMNN algorithm with a convex loss function [6] by calculating the cosine similarity between the input data. The purpose of the loss function is to ensure that the local nearest neighbors around every input target i-vector having the same class label are moved closer together and inputs with different class labels are pushed further apart. One of the advantages of the CSLMNN algorithm is that the metric (global) is optimized locally. To achieve this, the CSLMNN algorithm needs prior information of nearest neighbors of the target class. We do this by measuring the Cosine distance between the i-vectors. Let $j \rightsquigarrow i$ indicate $w_j$ is a target neighbor of $w_i$. CSLMNN learns the Mahalanobis metric $M$ such that it keeps each input $w_i$ close to its target neighbors while input vectors of different classes (impostors) are separated by a large margin. Here learning a metric is equivalent to learning a linear transformation that maps input vectors to a transformed space where the above property holds. For the input $w_i$, having the target $w_j$ and impostor $w_k$ the relation is expressed as the following equation with respect to the cosine similarity metric.

$$\frac{\langle Mw'_i, Mw'_j \rangle}{\|Mw'_i\| \|Mw'_j\|} - \frac{\langle Mw'_i, Mw'_k \rangle}{\|Mw'_i\| \|Mw'_k\|} \geq 1 \tag{9}$$

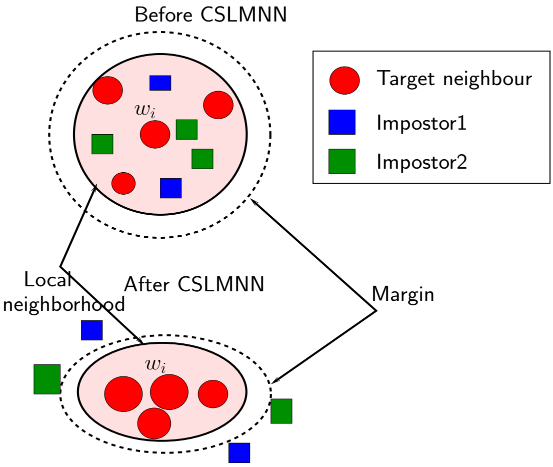

**Fig. 2.** Illustrates transfer of the i-vector before and after CSLMNN Metric learning. The small circles represent i-vectors with the same class label, while small rectangles represent i-vectors with different class labels.

In Figure 2 the i-vectors with the same class label are shown as small circles, while small rectangles represent i-vectors with different class labels (impostors). In the input space all points on a circle are equidistant from $w_i$. After learning the CSLMNN metric it can be seen that in the new transformed space the i-vectors of same speaker are moved closer, while impostors are moved far.

The semidefinite program (SDP) proposed in [6] moves the target neighbors closer by minimizing cosine distance $\sum_{j \rightsquigarrow i} \left(1 - \frac{\langle Mw'_i, Mw'_j \rangle}{\|Mw'_i\| \|Mw'_j\|}\right)$, while penalizing the criteria of separating the data of different class by a large margin. This problem is further solved by introducing an additive slack variable $\xi_{ijk} \geq 0$ for different class label, so that the SDP can be formulated as shown in the following Table 1. The triplet variable is $S = \{(i, j, k) : j \rightsquigarrow i, y_k \neq y_i\}$, where $y_k$ and $y_i$ are input data labels.

**Table 1.** Formulation of the convex optimization problem for the CSLMNN method

$$\min_M \sum_{j \rightsquigarrow i} \left(1 - \frac{\langle Mw'_i, Mw'_j \rangle}{\|Mw'_i\| \|Mw'_j\|}\right) + \mu \sum_{(i,j,k) \in S} \xi_{ijk}$$

**Subject to** : $(i.j, k) \in S$ :

**(1)** $\frac{\langle Mw'_i, Mw'_j \rangle}{\|Mw'_i\| \|Mw'_j\|} - \frac{\langle Mw'_i, Mw'_k \rangle}{\|Mw'_i\| \|Mw'_k\|} \geq 1 - \xi_{i,j,k}$

**(2)** $\xi_{i,j,k} \geq 0$

**(3)** $M \succeq 0$

Once the CSLMNN metric $M$ is learned from the example data, scores are obtained using the following equation for the i-vectors.

$$Score_{LMNN}(w'_{target}, w'_{test}) = \frac{\langle Mw'_{target}, Mw'_{test}\rangle}{\|Mw'_{target}\|\|Mw'_{test}\|} \tag{10}$$

The overall block diagram of the speaker verification incorporating the CSLMNN algorithm is shown in Figure 3.

## 3 Performance Evaluation

In this Section we describe how the new algorithm was evaluated and compare its performance with competing approaches. Experiments were done on the NIST-2008 and YOHO databases. DET curves and EER were used to evaluate the performance of the CSLMNN metric. We compare both the raw cosine and LDA transformed i-vectors and discuss the significance of the improvements obtained in terms of the DET curves and EER using the proposed method.

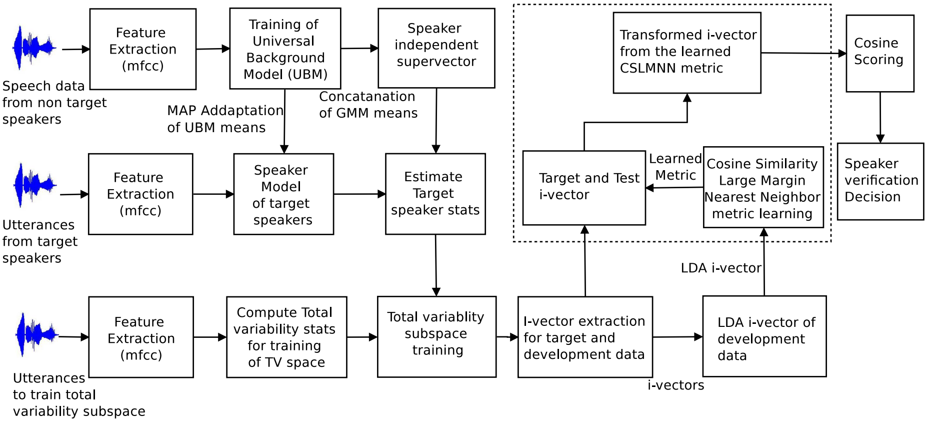
Speech data from non target speakers
Feature Extraction (mfcc)
Training of Universal Background Model (UBM)
Speaker independent supervector
MAP Addaptation of UBM means
Concatanation of GMM means
Utterances from target speakers
Feature Extraction (mfcc)
Speaker Model of target speakers
Estimate Target speaker stats
Utterances to train total variability subspace
Feature Extraction (mfcc)
Compute Total variability stats for training of TV space
Total variablity subspace training
I-vector extraction for target and development data
i-vectors
LDA i-vector of development data
LDA i-vector
Target and Test i-vector
Learned Metric
Cosine Similarity Large Margin Nearest Neighbor metric learning
Transformed i-vector from the learned CSLMNN metric
Cosine Scoring
Speaker verification Decision

**Fig. 3.** Block diagram illustrate the proposed speaker verification using the learned metric. The cosine distnace metric learning is shown in dotted box.

### 3.1 Organization of the Various Data Sets

Two data sets NIST-2008 and YOHO were used to evaluate the proposed method. NIST 2002/2004 was used to train the background models. For development i-vectors, NIST-2005, NIST-2006 and NIST-2007 databases were used.

- **NIST 2008 Database**: The short2-short3 condition of NIST-2008 database was used here to evaluate the proposed method. The database consists of 648 male and 1140 female speakers. Short 2 condition was used for training and short 3 for testing. Short2 data contains both telephone speech as well as interview speech. The interview speech samples used in the short2 condition are of 3 minute duration extracted from excerpts of the longer interview speech. The short3 condition testing data consists of the type of speech in short2 data as well as telephone speech recorded with an auxiliary microphone [9] [10].

- **YOHO Database**: The YOHO databse consists of 108 male and 30 female speakers. The database was collected during testing of ITT'S speaker verfication system in an office environment. Speaker variation spanned a wide range over attributes like age, job description and educational background. Most of the speakers are from the New York city area with some non-native English speakers. The data was collected using a high quality telephone handset (Shure XTH-383), but did not pass through the telephone channel [11].

### 3.2 Experimental Conditions

A 39 dimensional MFCC (mel frequency cepstral coefficient) (13 static, 13 $\Delta$ and 13 $\Delta\Delta$) was obtained as a feature vector for i-vector extraction from the speech signal at a frame rate of 10 ms with 20 ms Hamming window. Silence was removed using VAD (Voice activity detection) and MFCC features were normalized with standard cepstral mean subtraction and variance. The UBM of 512 mixture components with diagonal covariance matrices was trained using data from non-target speakers. Maximum likelihood criteria were used for the training of the UBM model. For the i-vector system the total variability space $T$ was trained using non-targeted speakers. I-vectors of 400 dimension were extracted from speech segments for the training and testing phases.

### 3.3 Experimental Results

Experiments on the NIST 2008 and YOHO database were done using the LMNN algorithm.

- Raw cosine scoring method: In this method cosine scoring is used to obtain the scores and there was no metric learning. This is the Baseline system in the i-vector framework.
- Raw cosine + Euclidean LMNN method: In this method the scoring is done on the transformed vector obtained after the LMNN matrix $M$ is learned from the development data by using Euclidean distance between the i-vectors. .
- Raw Cosine + CSLMNN Method : In this method the matrix $M$ is learned from the development data by measuring the cosine distance between the i-vectors.

- Raw cosine + LDA method: Here the LDA transformation matrix $A_{lda}$ is used to reduce the dimension of the i-vector. The matrix $A_{lda}$ is estimated using the development data. The transformed LDA i-vector is used for final cosine scoring.
- Raw cosine + LDA + Euclidean LMNN: This method involved applying LDA on the raw i-vector followed by Euclidean LMNN on the reduced LDA i-vectors.
- Raw cosine + LDA + CSLMNN: This method involved applying the CSLMNN metric scoring on LDA i-vectors. This method gives the best result compared to all above methods.

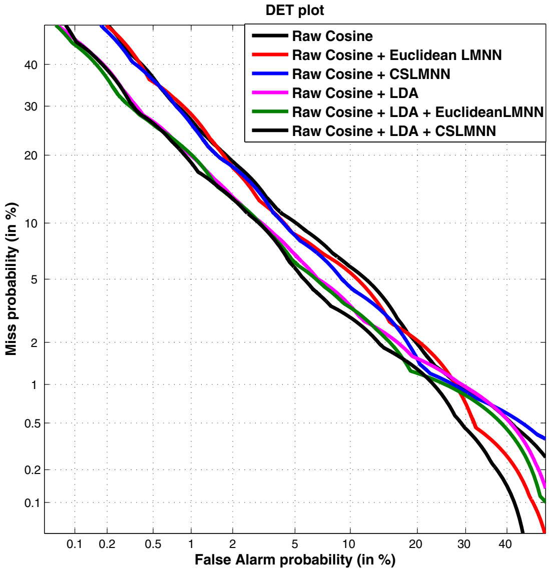

**Fig. 4.** DET plots of speaker verification for the NIST 2008 Database

The experimental results obtained using the proposed method have been presented in the form of DET [12] curves where a lower curve is interpreted as better performance. The corresponding EER values from both databases are also presented here. The DET plots for the NIST-2008 and YOHO databases using the LMNN metric are shown in Figure 4 and 5 respectively. The minimum DCF values are also evaluated for both the databases. Table 2 and Table 3 show the DCF value and EER value for NIST-2008 and YOHO databases. The DET curve is plotted using the Bosaris toolkit [13].

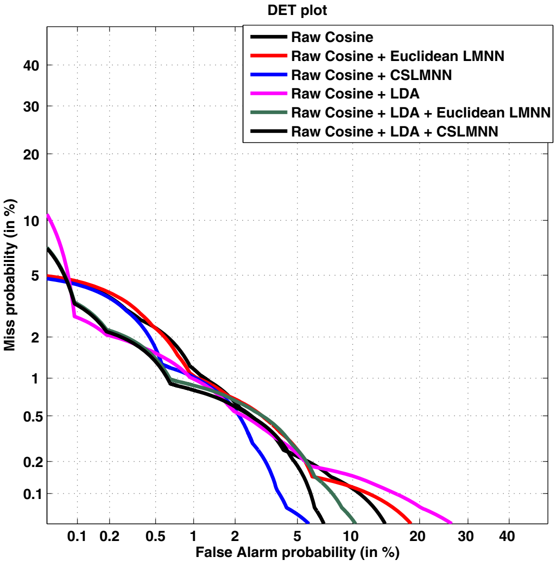

**Fig. 5.** DET plots of speaker verification for the YOHO Database

**Table 2.** DCF and EER values for NIST-2008 data

| Scoring Method | EER | min DCF |
|---|---|---|
| Raw Cosine | 7.53 | 0.2077 |
| Raw Cosine + Euclidean LMNN | 7.06 | 0.1985 |
| Raw Cosine + CSLMNN | 6.86 | 0.1980 |
| Raw Cosine + LDA | 5.88 | 0.1712 |
| Raw Cosine + LDA + Euclidean LMNN | 5.66 | 0.1702 |
| Raw Cosine + LDA + CSLMNN | **5.34** | **0.1698** |

**Table 3.** DCF and EER values for YOHO data

| Scoring Method | EER | min DCF |
|---|---|---|
| Raw Cosine | 1.09 | 0.0342 |
| Raw Cosine + Euclidean LMNN | 1.02 | 0.0324 |
| Raw Cosine + CSLMNN | 1.01 | 0.0258 |
| Raw Cosine + LDA | 0.99 | 0.0250 |
| Raw Cosine + LDA + Euclidean LMNN | 0.90 | 0.0251 |
| Raw Cosine + LDA + CSLMNN | **0.85** | **0.0244** |

## 4 Conclusions

The proposed Cosine similarity large margin nearest neighborhood metric learning method transform the input i-vectors into a new feature space. More specifically the method proposed herein learns the metric in such a way that the intra speaker variability is kept minimum by keeping the k-nearest neighbors closer to the target i-vector. Additionally, it also ensures that impostors are kept at a large margin from the target speaker. We have used cosine distance as a similarity measure for the metric learning due to competitive result obtained in i-vector based speaker verification system. The CSLMNN metric method can be fused with other compensation method to further improve the performance of speaker verification system. This is possible due to the learning methodology followed in the CSLMNN. The experimental results are encouraging and future work will focus on utilizing score normalization techniques in the cosine similarity large margin nearest neighborhood framework.

## References

1. Tomi, K., Li, H.: A tutorial on text-independent speaker verification. Speech Communication 52, 12–40 (2010)
2. Dehak, N., Kenny, P.J., Dehak, R., Dumouchel, P., Ouellet, P.: Front-end factor analysis for speaker verification. IEEE Transactions on Audio, Speech, and Language Processing 19(4), 788–798 (2011)
3. R.D.A.Q.T.F.,, D.R.: Speaker verification using adapted gaussian mixture models. Digital Signal Processing 10(1), 19–41
4. S.M., J.T.: Learning a distance metric from relative comparisons. In: Advances in Neural Information Processing Systems, vol. 16, p. 41 (2004)
5. Davis, J.V., Kulis, B., Jain, P., Sra, S., Dhillon, I.S.: Information theoretic metric learning. In: Proc. Int. Conf. Mach. Learn., pp. 209–216 (2007)
6. B.J.W.K.Q., S.L.K.: Distance metric learning for large margin nearest neighbor classification. In: Advances in Neural Information Processing Systems, pp. 1473–1480 (2005)
7. Yang, L.: An overview of distance metric learning. In: Proceedings of the Computer Vision and Pattern Recognition Conference (2007)
8. Xing, E.P., Jordan, M.I., Russell, S., Ng, A.: Distance metric learning with application to clustering with side-information. In: Advances in Neural Information Processing Systems, pp. 505–512 (2002)
9. Scheffer, K.S.S.N., Graciarena, M., Shriberg, E., Stolcke, A., Ferrer, L., Bocklet, T.: The sri nist 2008 speaker recognition evaluation system. In: IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2009, pp. 4205–4208 (2009)
10. Li, H., Ma, B., Lee, K.-A., Sun, H., Zhu, D., Sim, K.C., You, C.: The i4u system in nist 2008 speaker recognition evaluation. In: IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2009, pp. 4201–4204 (2009)
11. Campbell Jr., J.P.: Testing with the yoho cd-rom voice verification corpus. In: International Conference on Acoustics, Speech, and Signal Processing, ICASSP 1995., vol. 1, pp. 341–344. IEEE (1995)
12. Martin, A., Doddington, G.: The det curve in assessment of detection task performance. In: Proc. Eurospeech, vol. 97(4), pp. 1895–1898 (1997)
13. B.N., de Villiers, E.: The bosaris toolkit: Theory, algorithms and code for surviving the new dcf. arXiv preprint arXiv, 1304.2865 (2013)