Score Fusion Methods for Text-Independent Speaker Verification Applications

Florin Răstoceanu, Marilena Lazăr
Information Systems and Communications Test & Evaluation Scientific Research Center
Military Equipment and Technologies Research Agency
Bucharest, Romania
rastoceanu_florin@yahoo.com

***Abstract*—Speaker verification methods are various and use different types of features, but each system alone do not perform satisfactory results. This paper makes a comparison of different features and methods for score fusion for an independent speaker verification application. Several types of spectral features are used as speaker data. The scores obtained with these types of features were fusioned with combination methods (as: mean, sum, max, min, weighted sum) and classification methods (as: SVM, linear discriminant). The experiments were performed on a laboratory registered database for Romanian language and demonstrate that fusion methods outperformed the baseline GMM-UBM method.**

Keywords-SVM; GMM; score fusion; speaker verification.

I. Introduction

In order to successfully use in practice speaker verification systems they must have several features such as: accuracy, efficiency, robustness, practical application and generality. For realization of speaker verification systems now are used different methods that achieve one or more of those features. To integrate all of these properties within one single system it could be used the fusion on different levels. In literature there are known four great fusion levels for speaker verification systems [1]. Fusion could be realized at sensors level, features level, scores level and decision level.

Sensor level fusion could be achieved in speaker recognition systems by using, for speech sequence recording, several types of microphones. Using fusion on this level an improvement of system accuracy it is to be expected.

Feature level fusion could be achieved by extracting different types of discriminative features from speech signal and concatenating them in one single feature vector. The main reason for which it is used this kind of method is that the features vector, obtained by concatenation, comprise more information about the speakers. The problem that could appear is that the dimension of this vector could become very large and therefore the computation capacity would have to be increased.

Score level fusion consists in score combination, using different feature types or different classifiers.

Decision level fusion is similar with score level fusion with the mention that the scores are used in the acceptance/rejection decision before fusioning.

Scores fusion seems to be the most favorable and, so, the most used method because of its simplicity and good performance.

II. Score Fusion techniques

In the context of a speaker verification task, two distinct approaches of score-level fusion could be considered: the combination approach and the classification approach. The first one formulates the score fusion as a combination problem. In this case it is necessary a score normalization before fusion. By these methods the individual matching scores are combined to generate a single scalar score, which is then used to make the final decision. In the classification approach the matching scores are considered as input features for a second–level pattern classification problem, between two classes, either client or impostor. For this reason, feature vectors are created from the matching scores obtained with different methods. These feature vectors are used to train for every speaker two models: a client model from scores obtain by him with his speech data and an impostor model obtain with the others speech data. These models are used afterwards in taking the final decision: accept as a client or reject as an impostor. In contrast to the combination approach, these classifiers are capable of learning the decision boundary irrespective of how the vector is generated, so that the scores of the different modalities can be non-homogeneous and no normalization is required prior to using the classifier in the fusion process.

In this paper there are used for experiments several types of combination methods which are also the most popular techniques for such types of systems [2], [3].

Simple sum, also known as the **sum rule**, is the most straightforward fusion method based on the combination approach. All the scores *(S_i)* of the *N* normalized subsystems are directly summed, resulting in a final score $\mathrm{S}_{\mathrm{final}}$:

$$\mathrm{S}_{\mathrm{final}} = \sum_{i=1}^{n} S_i \ . \qquad (1)$$

978-1-4577-0441-3/11/$26.00 ©2011 IEEE

In the **product rule**, the resulting score S_{final} is obtained by multiplying the normalized scores of the individual subsystems:

$$S_{final} = \prod_{i=0}^{n} S_i \,. \tag{2}$$

Other fusion rules are based on the extreme values of the subsystems scores. **Max rule**, for example, takes the maximum normalized score from all the N subsystems as the final score, while **min rule** takes the minimum value:

$$S_{final} = \max(S_1, S_2, ..., S_n) \tag{3}$$

$$S_{final} = \min(S_1, S_2, ..., S_n) \tag{4}$$

All methods presented above are based on fixed rules. For increasing the effectiveness are used some information about the performances obtained by individual subsystems alone. Therefore the EERs (Error Equal Rate) of the individual subsystems are used for weighting the scores in a method named **weighted sum**. In this method each individual score is weighted by a factor proportional to it's recognition rate, so that the weights for more accurate matchers are higher than those of less accurate matchers. The weight for each score is proportional with inverse of its EER. Denoting w_i and E_i the weighting factor and the EER for the i^{th} subsystem, respectively, S_i the individual score of such subsystem, and N the number of subsystems, the final score S_{final} is expressed as:

$$S_{final} = \sum_{i=1}^{N} w_i S_i \tag{5}$$

where

$$w_i = \frac{1}{E_i \left(\sum_{m=1}^{N} \frac{1}{E_m} \right)} \quad \text{and} \quad \sum_{i=1}^{N} w_i = 1 \,.$$

The experiments include also several types of classification methods. One of these methods is Linear Discriminant (LD). LD is a simple linear projection of the input vector ***x*** onto a uni-dimensional space, so that a linear boundary between classes can be satisfactory obtained. The equation for the linear boundary is given as [4],[5]:

$$h(x) = w^T x + b \tag{6}$$

where ***w*** is a transformation vector obtained on the development data using a Fisher criterion, *T* is the transpose operation, and ***b*** is a threshold determined on the development data to give the minimum error of classification in respective classes. The rule for class allocation of any data vector is given by:

$$x \in \begin{cases} c_1 \\ c_2 \end{cases} \quad for \quad w^T x + b \begin{cases} \geq 0 \\ < 0 \end{cases} \tag{7}$$

where c_1 and c_2 are the client and impostor classes respectively and w is given by:

$$w = (\Sigma_1 + \Sigma_2)^{-1} (\mu_2 - \mu_1) \tag{8}$$

where Σ_i and μ_i are variances and means of client and impostors scores.

A support vector machine (SVM) is a state-of-the-art binary classifier and one of the most currently fusion techniques based on the classification approach. This technique can be successfully used in pattern recognition and information retrieval tasks. The main idea in training a SVM system is to find a hyperplane as a decision boundary between two classes of objects. In this case the classes are composed from client and impostor scores.

A. *Linear Case*

Consider the problem of separating the set of *N* training vectors $\{(x^1, y^1), ..., (x^n, y^n)\}$, x Є $\Re^m$, belonging to two different classes y_i Є$\{-1, 1\}$. The goal is to find the linear decision function *D(x)* and the separating plane *H*.

$$H :< w, x > +b = 0 \tag{9}$$

$$D(x) = sign(w \cdot x + b) \tag{10}$$

where *b* is the distance of the hyperplane from the origin and *w* is the normal to the decision region.

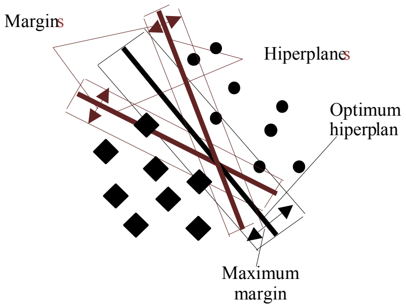

Figure 1. Separation hyperplanes

Let the "margin" of the SVM be defined as the distance from the separating hyperplane to the closest two classes. The SVM training finds the optimal separating hyperplane. The optimal hyperplane is the one with the maximum margin. The margin is equal to 2/||w||. Once the hyperplane is obtained, all the training examples satisfy the following inequalities:

$$x_i * w + b \geq +1 \quad for\ y = +1 \tag{11}$$

$$x_i * w + b \geq -1 \quad for\ y_i = -1 \tag{12}$$

We can summarize the above procedure to the following [6]:

$$\begin{gathered} Minimize\ L(w) = \frac{1}{2}\|w\|^2 \\ Subjecto\ \ y_i(x_i * w + b) \geq +1, \quad i = 1, 2, \cdots, N \end{gathered} \tag{13}$$

B. Non-linear Case

Real-world classification problems typically involve data that can only be separated using a nonlinear decision surface. Optimization on the input data in this case involves the use of a kernel-based transformation that transforms data in a higher dimensional space (feature space) in which data are linear separable.

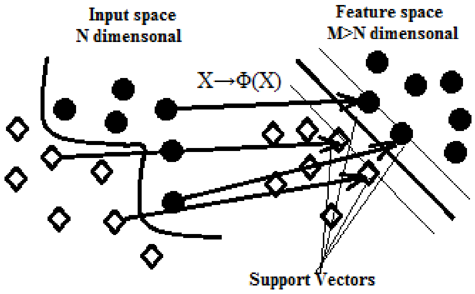

Figure 2. SVM principle

Kernels allow a dot product to be computed in a higher dimensional space without explicitly mapping the data into these spaces.

The resulting decision function is of the form:

$$f(x) = \mathrm{sgn}\left(\sum_{i=1}^{n} \alpha_i y_i K(x_{iv}, x) + b^* \right) \qquad (14)$$

where $\boldsymbol{x}$ is a vector containing the individual scores, α_i are the Lagrange multipliers, x_{iv} are support vectors, K represent the kernel function and b^* is a constant.

III. Speaker Verification System

A. General speaker verification method

Speaker verification is the task of deciding whether a speech utterance is delivered by a given claimant speaker or not. More formally, it is the task of deciding, given a speech signal *x* and a hypothesized speaker *S*, whether *x* was spoken by *S*. This is also referred to as speaker detection or single-speaker detection. The binary decision can be reformulated as a hypothesis test between the following statements [7]:

H0 : x is from the hypothesized speaker.

H1 : x is not from the hypothesized speaker.

Then the decision in an optimal manner is:

$$T(x) = \frac{f(H_0 \mid x)}{f(H_1 \mid x)} \begin{cases} \geq \eta, & accept\ H_0 \\ < \eta, & rejectt\ H_0 \end{cases} \qquad (15)$$

where T(x) is denoted as the test ratio (for speaker verification systems using GMM is the likelihood ratio) and η is the threshold value. In this paper the case function *f()* is represented by the Gaussian Mixture Models (GMM). A typical speaker verification system operates as follows in Fig. 3. The model defined by the function *f(H₁|x)* is trained on speech from other different speakers and it is denoted as the Universal Background Model (UBM) or the reference model. The speaker model defined by *f(H₀|x)* is simply trained on the speaker's voice in a procedure denoted as enrollment. Finally, a speaker claims an identity, a test utterance is recorded and a decision is made.

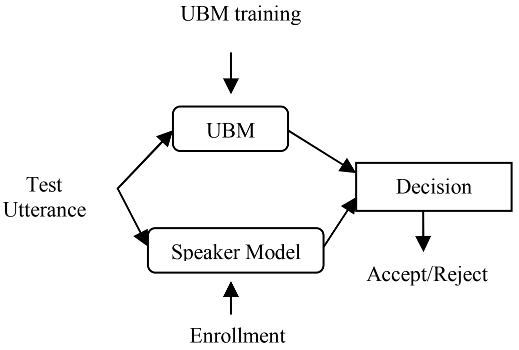

Figure 3. A typical speaker verification system [7]

B. Gaussian Mixture Models

The use of Gaussian Mixture models for modeling speaker identity is motivated by the interpretation that the Gaussian components represent some general speaker dependent spectral shapes and the capability of Gaussian mixtures to model arbitrary densities. A GMM is the weighed sum of ***M*** component densities. According to this, the likelihood of a sequence X of feature vectors from the audio data is given by the equation [8]:

$$p(X \mid \lambda) = \sum_{i=1}^{M} p_i b_i(x) \qquad (16)$$

where $\boldsymbol{x}$ is $\boldsymbol{D}$ dimensional speech feature vector, $\boldsymbol{b_i(x)}$, $i=1....M$ are component densities and $\boldsymbol{p_i}$, $i=1....M$ are the mixture weights. Each component density is a $\boldsymbol{D}$ Gaussian function of the form:

$$b_i(x) = (2\pi)^{-\frac{D}{2}} |\Sigma_i|^{-\frac{1}{2}} \exp\{-\frac{1}{2}(x-\mu_i)^T \Sigma_i^{-1} (x-\mu_i)\} \qquad (17)$$

with mean vector $\boldsymbol{\mu_i}$ and covariance matrix $\boldsymbol{\Sigma_i}$. The mixture weights are such that $\Sigma_{i=1}^{M} p_i = 1$.

For speaker verification, each speaker is represented by a GMM, λ_i, which is completely parameterized by its mixture weights, means and covariance matrices collectively represented as:

$$\lambda_i = \{p_i, \mu_i, \Sigma_i\} \qquad (18)$$

For computational ease and improved performance, the covariance matrices are constrained to be diagonal.

There are several techniques that can be used to estimate the parameters of a GMM, λ$_i$, which describes the distribution of the training feature vectors. By far the most popular and well-established is Maximum Likelihood (ML) estimation.

These GMMs are trained separately on each speaker's enrollment data using the Expectation Maximization (EM) algorithm. The update equations that guarantee a monotonic increase in the model's likelihood value are for mixtures weights equation (20), for means equation (21) and for variances equation (22).

$$p_i = \frac{1}{T}\sum_{t=1}^{T} p(i \mid x_t, \lambda) \tag{19}$$

$$\mu_i = \frac{\sum_{t=1}^{T} p(i \mid x_t, \lambda) x_t}{\sum_{t=1}^{T} p(i \mid x_t, \lambda)} \tag{20}$$

$$\sigma_i^2 = \frac{\sum_{t=1}^{T} p(i \mid x_t, \lambda) x_t^2}{\sum_{t=1}^{T} p(i \mid x_t, \lambda)} - \mu_i^2 \tag{21}$$

In speaker verification using GMM, the decision is given by:

$$\begin{aligned} &\mathit{If}\ \ p(X_{test} \mid \lambda_i) \geq \eta \Rightarrow X_{test}\ \mathit{is\ spoken\ by\ speaker}\ i \\ &\mathit{If}\ \ p(X_{test} \mid \lambda_i) < \eta \Rightarrow X_{test}\ \mathit{is\ spoken\ by\ another\ speaker} \end{aligned} \tag{23}$$

C. *Speaker verification method using score fusion*

In this section we briefly describe the method used to demonstrate the utility of using score fusion in speaker verification systems. A text-independent speaker verification method was implemented. The implemented method is based on a Gaussian Mixture Models with Universal Background Model (GMM-UBM).

The method for score fusion is shown in Fig. 4.

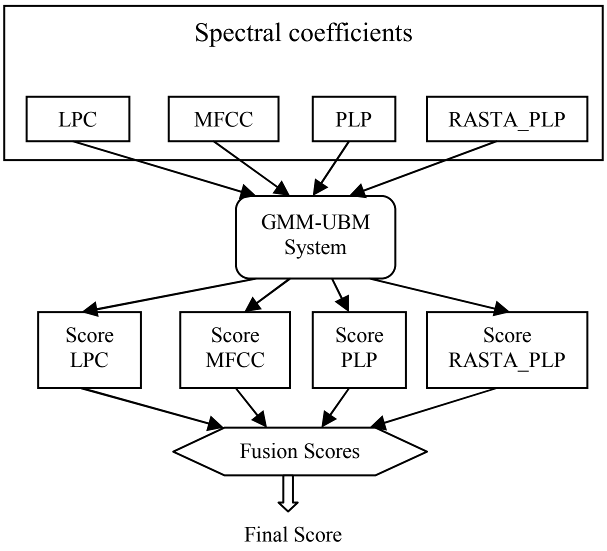

Figure 4. Fusion score method

The fused score were obtained from the baseline system GMM-UBM, using four different coefficient types: MFCC (Mel-Frequency Cepstral Coefficients), LPC (Linear Prediction Coefficients), PLP (Perceptual Linear Prediction), and RASTA (Relative Spectral Transform - Perceptual Linear Prediction). For MFCC and LPC a number of 12 coefficients with first and second derivates are used and for PLP and RASTA_PLP a number of 13 coefficients also with first and second derivates are used. The scores obtained with these coefficients types were then fusioned using combination methods as mean, sum, max, min, weighted sum and classification methods as SVM and linear discriminant.

IV. Experiments and Results

The experiments were performed on a small database with 22 speakers for Romanian language. The database was recorded in a lab and contains a number of 120 different sentences spoken by each speaker. The baseline GMM-UBM system was constructed as follows: in the training phase for each speaker a GMM model was trained with a number of 40 sentences. For each speaker an UBM model was trained by using 10 sentences from the other's speakers from the database. In the testing phase the scores obtained by the client phrases with his model are normalized by decreasing with the score obtained with the UBM. The other 70 phrases were used in the testing phase for each speaker therefore for each one the same number of client score was obtained. A part of these scores were used in training phase for classification fusion methods and the rest for testing the methods. All the GMM models – speakers' models and UBM models – are created using a number of 30 Gaussians. The reason for using this number of Gaussians was according to the fact that using a bigger number on this database, the performances would not increase significantly.

TABLE I. TABLE I RESULTS (EER) COMPARING FUSION METHODS

Fusion methods **Feature combinations**	**SVM**	**Weighted sum**	**Sum**	**Prod**	**LD**	**Min**	**Max**
C1: PLP, RASTA_PLP, MFCC, LPC	**1.9481**	4.1126	5.1948	5.1948	4.3974	5.8442	5.8442
C2: RASTA_PLP, MFCC, LPC	**2.3810**	3.6797	4.7619	4.9784	4.7704	5.6277	5.8442
C3: PLP, MFCC, LPC	**2.3810**	4.1126	4.5455	4.5455	4.9271	4.7619	5.4113
C4: PLP, RASTA_PLP, LPC	**2.1645**	*3.4632*	4.7619	4.7619	5.0581	5.6277	5.1948
C5: PLP, RASTA_PLP, MFCC	**3.2468**	*5.6277*	5.8442	5.8442	6.5733	6.7100	6.5733
C6: RASTA_PLP, MFCC	**5.4113**	**5.8442**	**6.2771**	**6.4935**	7.8435	**6.2771**	8.4416
C7: PLP, RASTA_PLP	**4.7619**	5.8442	5.8442	5.8442	6.5846	7.3593	6.2771
C8: PLP, MFCC	*3.4632*	6.0606	6.0606	6.0606	6.9948	6.0606	6.0606
C9: LPC, RASTA_PLP	**2.8139**	4.5455	4.3290	4.5455	5.6590	5.6277	4.5455
C10: MFCC, LPC	**2.1645**	4.3290	4.7619	4.9784	4.9527	4.9784	4.7619
C11: LPC, PLP	**2.8139**	3.8961	4.3290	4.3290	5.2888	3.8961	4.9784

The equal error rate (EER) is used to measure the performance in all our evaluations. In this paper EER represents the location on a DET (Detection Error Rate –for details see [9]) curve where the false accept rate and false reject rate are equal.

TABLE II. RESULTS OBTAINED BY THE BASELINE SYSTEMS

Feature type	EER
PLP	5.6277
RASTA_PLP	9.0909
MFCC	7.1429
LPC	3.4632

The experiments have been realized in Matlab 7.0. The baseline systems were realized with GMM and UBM by using the features types listed in table II (PLP, RASTA_PLP, MFCC, LPC). For the implementation of the UBM-GMM baseline systems has been used a Matlab toolbox named VOICEBOX [10], and for SVM, LIBSVM a library, developed by the National Taiwan University [11].

To measure the performance in all our evaluations we used equal error rate (EER). The result obtained by the baseline systems are listed in table II.

From table II it is to be noticed that the best results are obtained by LPC and the worst by RASTA_PLP.

All the fusion methods mentioned in chapter 2 have been implemented. In table I are presented the EER values obtained by the seven fusion methods (SVM, WEIGHTED_SUM, SUM, PROD, LD, MIN, MAX) and used for the eleven feature combinations (C1÷C11).

For each fusion method and a certain feature combination a different value for the EER was obtained. The purpose was to find the fusion method which, used with proper feature combination, gives the smallest EER value. To better observe which fusion method outperformed the baseline system the results, which exceeded the best result obtained by a baseline system, are represented in bold style, the results, that are the same with the baseline system performance that are represented in italic style and the others are the one that did not exceeded the baseline system performance.

According to values displayed in table II, with some exceptions, only SVM fusion method succeed to outperform the baseline system performances for all the types of the feature combinations. The results, obtained with SVM fusion method, compared with the results of the best baseline method, are represented in Fig. 5.

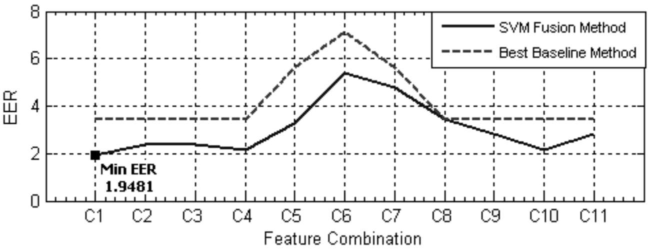

Figure 5. SVM fusion method results

From Fig. 5, it could be observed that the best results are obtained when all the four types of features are used and the differences in performance are as bigger as the number of features used in combination increases (as it can be observed explicitly in Fig. 6) .

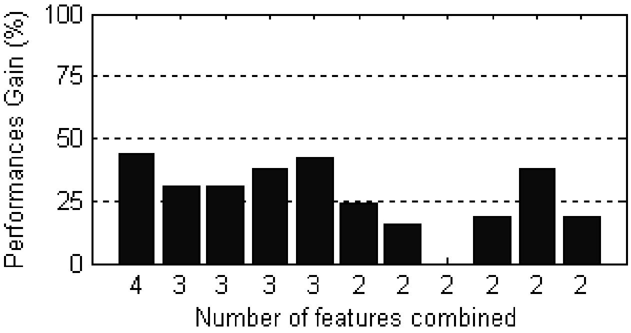

Figure 6. Performance gain against number of features combined

Starting from this observation it was attempted a combining SVM fusion method with the others methods which, used by themselves, would not obtain satisfactory results. On this line for SVM training have been used vectors obtained by scores of all baseline systems plus scores of the combination fusion methods with the best results. Because WEIGHTED SUM fusion method for C2 and C4 feature combinations obtained the lower EER, this was the one used in our experiments.

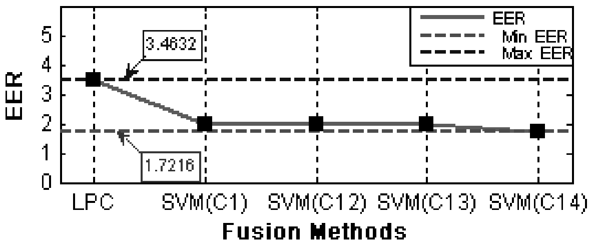

Figure 7. Combined fusion methods results

The results are presented in Fig. 7 as SVM(C12), SVM(C13) and SVM(C14), in comparison with the best baseline method LPC and the combination of features with best result (C1), obtained until now. In combinations, features C12 and C13 have been fusioned baseline systems scores with WEIGHTED SUM fusion method scores obtained for C2 and C4 respective. For C14 have been fusioned baseline systems scores with WEIGHTED SUM fusion method scores obtained for C2 and C4. If C12 and C13 combinations do not succeed to obtain better results, C14 has obtained the smallest error from which is concluded that by using a combination methods and SVM method (trained method) can give better result.

V. Conclusions

In this paper a comparison among the most used scores fusion methods in biometric systems has been made. These methods' performances have been compared using a text-independent speaker verification method with GMM-UBM, by using a clear speech database for Romanian language. From the performed experiments we concluded that not all fusion methods can obtain better results than baseline systems, but a combination between a learning and one or more combination methods can obtain better results. In our case we combined the best learning method, SVM, with the best combination method (WEIGHTED SUM) and achieved an EER two times smaller than the best ones obtained by a baseline system.

From all methods used, only SVM obtained satisfactory results. This is due to discriminative capacity of these classifiers which supplement the generative capacity of the Gaussian mixture used in baseline system.

References

[1] M. Faundez-Zanuy, ”Data fusion in biometrics,” IEEE Aerospace and Electronic Systems Magazine, vol. 20, pp. 34-38, 2005.

[2] J. Fierrez-Aguilar, J. Ortega-Garcia, and J. Gonzalez-Rodriguez, ”Fusion Strategies in Multimodal Biometric Verification,” Proceedings of the IEEE International Conference on Multimedia and Expo, ICME ’03, pp. 5 - 8, 2003.

[3] F. Roli, J. Kittler, G. Fumera, and D. Muntoni, ”An Experimental Comparison of Classifier Fusion Rules for Multimodal Personal Identity Verification Systems,” Proceedings of Multiple Classifier Systems, Sringer-Verlag, LNCS 2364, pp. 325-336, 2002.

[4] R. Fisher (1936), ”The Use of Multiple Measurements in Taxonomic Problems In: Annals of Eugenics”, 7, p. 179 – 188.

[5] McLachlan, ”Discriminant Analysis and Statistical Pattern Recognition In: Wiley Interscience”, 2004.

[6] N. Cristianini, J. Shawe-Taylor, ”An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods,” Cambridge University Press, 2000, chapther 6.

[7] D.□Neiberg, ”Text Independent Speaker Verification Using Adapted Gaussian Mixture Models,” 2000.

[8] D. A. Reynolds, R. C. Rose, ”Robust text independent speaker identification using Gaussian mixture speaker models, ” IEEE Transaction on speech and audio processing, vol 3, No 1, January 1995.

[9] A. Martin, G. Doddington, T. Kamm, M. Ordowski, M. Przybocki, “The DET Curve In Assesment Of Detection Task Performance,” Proceedings. Eurospeech '97, Rhodes, Greece, September 1997, Vol. 4, pp. 1899-1903.

[10] VOICEBOX: Speech Processing Toolbox for MATLAB http://www.ee.ic.ac.uk/hp/staff/dmb/voicebox/voicebox.html

[11] LIBSVM - A Library for Support Vector Machines developed by National Taiwan University - http://www.csie.ntu.edu.tw/~cjlin/libsvm/