Odyssey 2014:
The Speaker and Language Recognition Workshop
16-19 June 2014, Joensuu, Finland

STC Speaker Recognition System for the NIST i-Vector Challenge

Sergey Novoselov[1], *Timur Pekhovsky*[1,2], *Konstantin Simonchik*[1,2]

[1] Department of Speaker Verification and Identification,
Speech Technology Center Ltd., St. Petersburg, Russia
[2]ITMO, Russia

{novoselov,tim,simonchik}@speechpro.com

Abstract

This paper presents a Speech Technology Center (STC) system submitted to the NIST i-vector Challenge. The system includes different subsystems based on PLDA, LDA-SVM, RBM-PLDA and DBN-PLDA. We propose an original iterative scheme for clustering the NIST i-vector Challenge devset. We also introduce the RBM-PLDA subsystem in the NIST i-vector Challenge. Experiments performed on the progress dataset demonstrate that although the RBM-PLDA and DBN-PLDA subsystems are inferior to the other subsystems in terms of absolute minDCF, in the fusion they provide a substantial gain into the efficiency of the resulting STC system, reaching 0.239 at the minDCF point.

Index Terms: NIST, i-vector, PLDA, SVM, RBM.

1. Introduction

The NIST i-vector Challenge [1], like the earlier evaluations, deals with the task of speaker detection. The aim of this task is to determine whether or not the target speaker is speaking in a given segment of dialog speech. At the same time, the current evaluation differs from the previous evaluations in three important aspects:

- Feedback is possible, which means that each time after results are submitted, the participants are informed about the minDCF of their systems.
- All the data allowed for training are not labeled, i.e. they have no ID labels for gender, channel, language, etc.
- All input data are represented as i-vectors obtained using an i-vector extractor unknown to the participants.

The first condition allows participants to endlessly improve their systems (tune their parameters, thresholds, etc), which shifts the accent of the current research from theoretical novelty of methods towards sophisticated empirical work.

The second condition forces researchers to deal with the problem of clustering the training database of the NIST i-vector Challenge as much as with solving the main task: speaker detection. This is due to the fact that today the most successful method is the generative PLDA model in i-vector space [2-4]. Training a PLDA model requires a labeled training database.

The third of the three conditions is caused by the organizers' desire to interest the Machine Learning community in the speaker detection task. This means that all NIST participants are placed in equal conditions with regard to the ability to generate i-vectors.

Studies [5-7] show that the use of models like Deep Neural Networks (DNN) and Boltzmann Machines (BM), which were very successful for Automatic Speech Recognition (ASR), fails in the space of i-vectors for the speaker recognition task. These models clearly perform worse than classical PLDA. In our opinion, the use of acoustic features as input would give stimulus to those participants who work on feature extraction on the basis of new promising approaches, such as pseudo-i-vector extraction from the sets of MFCC features using DNNs [8-10].

To sum up, it is our belief that under the conditions of the current NIST i-vector Challenge, the most successful strategy is using the classic PLDA model, assuming the task of efficient training of the PLDA model on an unlabeled training dataset can be solved, which means searching for reliable methods of database labeling.

The aim of this paper is to provide a detailed description of the text-independent speaker verification system developed at Speech Technology Center Ltd for participation in the NIST i-vector Challenge.

The NIST i-vector Challenge organizers provided the devset as a set of telephone channel i-vectors of both genders, uniform in terms of channel and language, so for our first PLDA subsystem we used only one gender-independent Gaussian PLDA analyzer, instead of using, for instance, a mixture of PLDA analyzers [11] in order to capture a more complex structure in data.

The second subsystem that could be useful in the NIST i-vector Challenge was an SVM-based subsystem. For the SVM subsystem the problems of an unlabeled training dataset are not so critical, because it can use the whole development set, which according to the NIST conditions does not overlap with the evaluation set.

Our work has two key aspects that are marked by their novelty. The first deals with the problem of clustering and the search for the true labeling of the NIST i-vector Challenge devset. We propose using a version of agglomerative clustering in i-vector space as a clustering algorithm. The second is our third subsystem, in which we tested the use of Boltzmann Machines for the NIST i-vector Challenge.

The paper is organized as follows. A detailed description of the STC speaker verification system is given in Section 2. Section 3 shows how we performed the clustering of the NIST development set. Section 4 describes our final experiments on the test dataset of the NIST i-vector Challenge. Section 5 concludes the paper.

2. Description of the STC system

In this section we provide a description of all speaker verification subsystems used in our work.

2.1. Baseline cosine i-vector scoring

State-of-the-art speaker verification systems are systems working in the i-vector space. Each such vector is extracted using total variability factor analysis (TV-FA) [12] from a whole speaker utterance and is a good representation of the speaker for any subsequent classifiers. The TV-FA method makes it possible to obtain such a representation in the following way. In it the mean supervector

$$\mu = \mu_0 + T \cdot i, \tag{1}$$

where μ_0 is the mean supervector, T is the matrix that defines the Total Variability subspace, and i is the low dimension vector having the prior distribution $N(0, I)$.

The baseline system provided by the organizers of the NIST i-vector Challenge uses cosine evaluation, which is standard in i-vector technology:

$$\cos\ (i_{enrol}, i_{test}) = \frac{< i_{enrol}, i_{test} >}{\|i_{enrol}\| \cdot \|i_{test}\|}, \tag{2}$$

where i_{test} is the i-vector of the test utterance, i_{enrol} is the i-vector of the target speaker from the evaluation dataset. Since according to the conditions of the competition each target speaker has five model i-vectors, in the baseline system the i_{enrol} vector is obtained by simply averaging those five vectors. We should note that all i-vectors of the test set must be whitened.

2.2. PLDA subsystem

Among state-of-the-art speaker verification systems, leading positions are occupied by PLDA systems [3,4,13] working in the i-vector space. In our work we used a PLDA model both for verification and for the clustering task. In the case of the PLDA verification system we used the following model:

$$i_r(s) = m_0 + Vy(s) + Ux_r + \varepsilon_r, \tag{3}$$

where $i_r(s)$ is an F-dimensional i-vector from set $\{i_1, ..., i_R\}$, obtained from R utterances belonging to speaker $\boldsymbol{s}$, and $\boldsymbol{y}$, $\boldsymbol{x}$, $\varepsilon_r \propto N(0, \Sigma)$ are hidden speaker factors, channel factors and Gaussian noise, respectively.

In this paper, we assume the Gaussian nature of the priors of these variables. In (3) the model parameters are an [F × 1] mean vector m_0; a matrix V of dimension [F × N_1] whose columns are referred to as eigenvoices; a matrix U of dimension [F × N_2] whose columns are referred to as eigenchannels; and [F × F] diag-covariance matrix of the noise covariance matrix Σ. In the case of the PLDA clustering system, we used model

$$i_r(s) = m_0 + Vy(s) + \varepsilon_r, \tag{4}$$

without a channel term and full-covariance matrix Σ of the noise. This choice of models will be explained in detail in Section 3.5.

To obtain PLDA evaluations we used normalization of i-vectors, as proposed in [4]. The PLDA model makes it possible to calculate $P\ (i \mid tar)$, $P\ (i \mid imp)$ − the marginal likelihood for target and impostor hypotheses and, correspondingly, the PLDA score:

$$Score_{PLDA} = \ln \frac{P(i_{enroll}, i_{test} \mid tar)}{P(i_{enroll} \mid imp) \cdot P(i_{test} \mid imp)}. \tag{5}$$

In this paper we also obtained the target speaker i-vector i_{enroll} for the PLDA method using simple averaging of the given five model i-vectors. For the PLDA subsystem we used i-vector normalization proposed in [4].

2.3. LDA-SVM subsystem

It is well-known that using a discriminative SVM method in combination with another generative method, for example PLDA, produces a highly efficient speaker verification system [14,15].

In our verification system SVM was applied to the i-vectors after LDA projection (*l*-vectors). The distance from a test *l*-vector l_{test} to the SVM hyperplane of the *a*-th speaker $\Omega^{(a)}_{enroll}$ is given below:

$$f\ (l_{test}, \Omega^{(a)}_{enroll}) = \sum_{k=1}^{L} \alpha_k \cdot y_k \cdot K(l_{test}, l_k) - \alpha_0, \tag{6}$$

where l_k are the i-vectors after LDA (the L support vectors obtained by training the speaker's SVM hyperplane), y_k are the target values of two classes: {+1} for the *Target* class and {-1} for the *Imposter* class for the given speaker. A linear kernel $K(l_{test}, l_k)$ was used.

For the NIST i-vector Challenge our SVM system had three specific features:

- SVM was applied in the space of LDA projections of i-vectors.
- We used its own devset clustering algorithm for the SVM subsystem, in contrast to the PLDA and RBM-PLDA (see Section 2.4) subsystems, which yielded a labeled data set for LDA matrix training.
- S-normalization of the resulting SVM scores was used.

In this paper we used *s-normalization* [16] for SVM subsystem scores. In our case, if the enrollment is represented by $R=5$ mean *l*-vectors, the s-*normalized* score is calculated using the *Z*-normalized distance $f^{Z-norm}(l_{test}, \Omega^{(R)}_{enroll})$ obtained beforehand from the test *l*-vector l_{test} to the multi-session SVM hyperplane $\Omega^{(R)}_{enroll}$ and the z-normalized distance $f^{Z-norm}(l_{enroll}, \Omega_{test})$ from the mean *l*-vector of the enrollment l_{enroll} to the test SVM hyperplane Ω_{test}:

$$score = \frac{1}{2}\left[f^{Z-norm}(l_{test}, \Omega^{(R)}_{enroll}) + f^{Z-norm}(l_{enroll}, \Omega_{test})\right], \tag{7}$$

where the target speaker *l*-vector l_{enroll} was obtained using simple averaging of the given five model *l*-vectors, and the target

speaker hyperplane $\Omega_{enroll}^{(R)}$ was obtained using the imposter set and the target speaker *l*-vector l_{enroll}. The whole development set was used as an imposter set for the SVM and as the s-normalization set. This was done because the development and evaluation sets have non-overlapping speakers.

It is known that an SVM system is typically weaker than PLDA, but under the conditions of incomplete compensation of automatic labeling noise for the PLDA system, it can be expected that SVM, which does not need a labeled imposter set, will have an advantage. That was what we observed in our final experiments.

As a result of our submission on the NIST i-vector Challenge progress set, our SVM subsystem obtained minDCF = 0.286.

2.4. RBM-PLDA subsystem

The recent success of Deep Neural Networks [17] for Automatic Speech Recognition prompted the speaker recognition community to try to use Restricted Boltzmann Machines (RBM) for pseudo i-vector extraction [8-10].

We also decided to test this technology for the NIST i-vector Challenge. Figure 1 shows the diagram of our RBM for pseudo i-vector (b-vector) extractor. We will use this term further on, even though, strictly speaking, we are dealing with non-linear RBM transformation of the TV i-vector. We will call b-vector a vector of log posterior probabilities of the softmax layer.

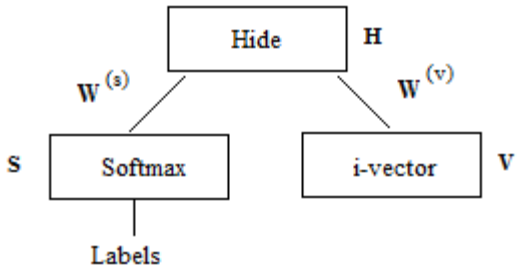

Figure 1. *Diagram of the supervized learning of the proposed RBM for pseudo-i-vector extractor*.

The visible input layer consists of $D = 600$ Gaussian units. The binary hidden layer consists of *H* units. The softmax layer consists of *S*=1745 units (the number of target speakers). $W^{(S)}$ and $W^{(v)}$ are the weights of the corresponding links. As output we use posteriors of the softmax layer:

$$p(s_i|h) = \frac{\exp\{s_i\}}{\sum_k^S \exp\{s_k\}}, \quad (8)$$

where the full input for the softmax unit is $s_i = a_i^{(s)} + \sum_k^H W_{ik}^{(s)} h_k$, and h_k are the states of the hidden units [18].

We did not use discriminative "fine-tuning" phase for training our extractor, limiting ourselves to generative pretraining of the extractor. This pretraining phase is the standard procedure of generative RBM training using contrastive divergence.

Following [18,19] we create a concatenated training set $X = \{v_k, s_k\}_{k=1}^K$, where v_k is the *k*-th input i-vector out of *K* development set vectors that were earlier clustered in *S* = 1745 cluster speakers by means of PLDA (see Section 3.5.). The binary vector s_k that corresponds to the *s*-th target speaker contains zeros, except for the *s*-th component, which equals 1. Thus, we implement a supervised scheme for generative RBM training, by feeding both the i-vectors and the labels of the target speakers to the hybrid binary Gaussian input layer *X*. Such an RBM is parametrized by joint distribution of hidden and observed variables:

$$p(s, v, h) = \frac{1}{Z} e^{-E(s,v,h)} \quad (9)$$

where the energy function *E* is:

$$E(s, v, h) = -\sum_i^D \left(a_i^{(v)} - v_i\right)^2 - \sum_i^S a_i^{(s)} s_i - \sum_k^H b_k h_k - \sum_i^D \sum_k^H W_{ik}^{(v)} v_i h_k - \sum_i^S \sum_k^H W_{ik}^{(s)} s_i h_k \quad (10)$$

and *Z* is the partition function, $a_i^{(v)}$, b_k and $a_i^{(s)}$ are the biases for the visible, hidden and softmax layers, respectively. Because of i-vector normalization, we suppose that standard deviations for the Gaussian visible layer $\sigma_i = 1$. The posteriors are defined in the following way:

$$p(h_k = 1|s, v) = sigm\left[b_k + W_k^{(s)} + \sum_i^D W_{ik}^{(v)} v_i\right], \quad (11)$$

where *sigm* denotes the sigmoid function. This training scheme allows us to model input data structure taking into account target speaker labels. In models with a hidden layer, effects on visible variables are highly correlated, so we can expect that the outputs of our softmax layer will be highly correlated as well. For this reason we apply PCA to the log of the 1745-dimensional vector of the softmax layer output, in order to obtain a pseudo i-vector with the dimension N_c, which we will refer to as b-vector.

Our experiments demonstrated that the best efficiency of the RBM-PLDA subsystem can be obtained when $N_c = H = 500$. After obtaining the pseudo i-vectors of 1745 speakers from the labeled part of the NIST i-vector Challenge development set (see Section 3.5), we used them for maximum-likelihood training of the standard PLDA model [2]. For the RBM b-vector PLDA (RBM-PLDA) subsystem we took a Gaussian PLDA model in the form (4), where the number of eigenvoices was $R_y = 400$. However, in contrast to the TV i-vector PLDA subsystem, where the noise covariance matrix Σ had a diagonal form, here it has the full covariance form.

As a result of our submission on the NIST i-vector Challenge progress set, our RBM-PLDA system obtained minDCF = 0.293.

2.5. Subsystems fusion

The fusion score over all subsystems was based on the linear model. Consequently, for our three subsystems we had:

$$Score = \sum_{k=1}^{3} C_k \cdot w_k \cdot score_k, \quad (12)$$

where the index *k* ranges over the set {LDA-SVM, PLDA, RBM-PLDA}, $score_k$ is the output value of the *k*-th subsystem, w_k is the weight coefficient for the *k*-th subsystem, $Score$ is the final score of the subsystems fusion. For PLDA and RBM-

PLDA subsystems we used σ-normalizing coefficients C_{PLDA} and $C_{RBM-PLDA}$ which were calculated on the clustered development set as follows:

$$C_{PLDA} = \sigma_{PLDA}^{-1}, \quad (13)$$

$$C_{RBM-PLDA} = \sigma_{RBM-PLDA}^{-1}, \quad (14)$$

where σ_{PLDA} and $\sigma_{RBM-PLDA}$ are the standard deviations of imposter scores of the PLDA and RBM-PLDA methods respectively. For LDA-SVM subsystem the σ-normalization coefficient was equal to 1 because of the use of s-norm (see Section 2.3):

$$C_{SVM} = \sigma_{SVM}^{-1} = 1. \quad (15)$$

Weight coefficients w_{PLDA}, $w_{RBM-PLDA}$ and $w_{LDA-SVM}$ were defined up to the second digit after the decimal by several submissions with the aim of minimizing the minDCF value of fused system.

2.6. Quality measure function

It is well-known that there is a dependence between the value of the minDCF threshold of a verification system and the duration of the speech segments that were used for extracting the enroll and test i-vectors. Using a quality measure function (QMF) [20] makes it possible to compensate the shift in minDCF thresholds for different speech segment durations, which improves the minDCF value of a verification system.

In the NIST i-vector Challenge we deal with 5 session speaker models. The total duration of all 5 model segments on average is much larger than the duration of the test segments. For this reason we ignored the dependence between the threshold shift and the durations of enroll fragrments and focused on estimating the dependence of the minDCF threshold shift on the test segment durations. QMF was examined in the PLDA verification system.

We used the result of clustering the devset data, as will be demonstrated in Section 3.5, and divided it into two subsets. One subset consisting of 1000 speaker classes was used for PLDA training. The other subset was used for testing and estimating the minDCF threshold values for different durations of test speech segments. In our experiments we also used 5 session speaker models (selected from the second subset of the development set) for making estimates at the minDCF point.

According to the competition conditions, segment durations followed a log normal distribution. We examined several points of test segment duration values around the maximum of the log normal distribution (Figure 2).

Figure 3 shows the dependence θ_{plda} of the minDCF threshold on the log values of test utterance durations. We applied linear approximation for describing such a dependence:

$$\theta_{PLDA} = k_{PLDA} \cdot \log(t) + b_{PLDA}. \quad (16)$$

This allowed us to use (17) as the QMF function for PLDA:

$$QMF_{PLDA}(t) = -k_{PLDA} \cdot \log(t). \quad (17)$$

In Formulas (16) and (17) t is the duration of the test segment in seconds. Using the approximation (16) we also calculated the expected value of the parameter $k_{PLDA} = 0.0205$, which was further specified by means of submissions on the progress set of the NIST i-vector Challenge.

In our experiments we discovered that the function $\log(t)$ in (17) can be easily replaced with $\sqrt{t}$. Then the value $k_{PLDA} = 0.0073$ provides the same minDCF on the NIST i-vector Challenge progress set as in the case of the function $\log(t)$.

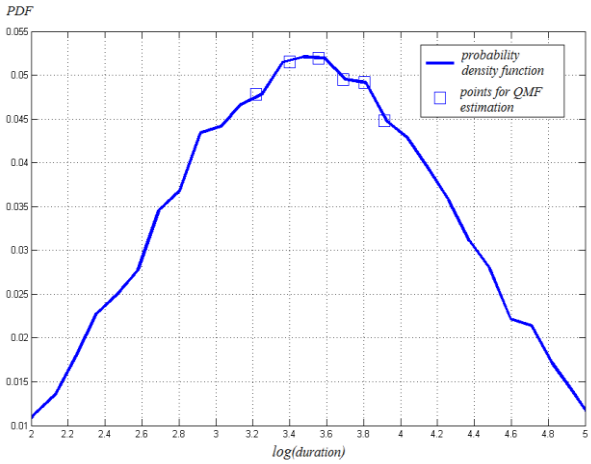

Figure 2. *Log normal distribution of speech segment durations in the development set and the points for QMF calculation.*

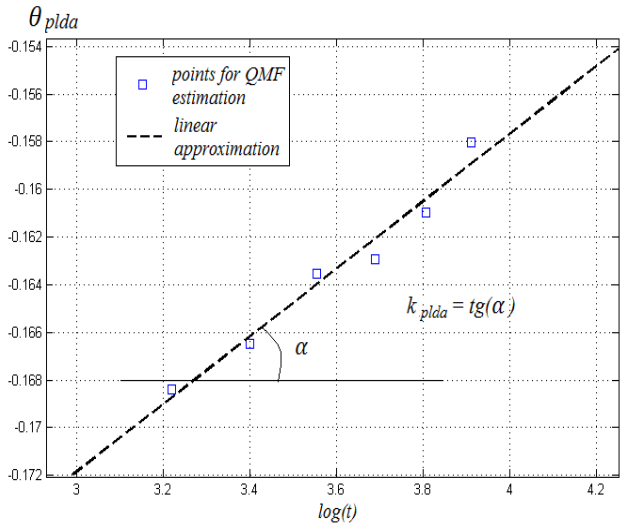

Figure 3. *Linear approximation of the dependence of the minDCF threshold value θ_{PLDA} on the log of the test utterance duration.*

Thus we used the function $\sqrt{t}$ as QMF:

$$QMF_{PLDA}(t) = -k_{PLDA} \cdot \sqrt{t}, \quad (18)$$

$$QMF_{LDA-SVM}(t) = -k_{LDA-SVM} \cdot \sqrt{t}, \quad (19)$$

$$QMF_{RBM-PLDA}(t) = -k_{RBM-PLDA} \cdot \sqrt{t}. \quad (20)$$

The coefficients $k_{RBM-PLDA}$ and $k_{LDA-SVM}$ were further specified by submissions as 6.1404 and 0.14, respectively.

3. The clustering problem

This section describes the clustering algorithms that we used in our subsystems. Description of the experiments performed on an in-House dataset is given.

3.1. In-House data

We assumed that the properties we revealed on the in-House data could be generalized for the i-vector space from the NIST i-vector challenge data. Our evaluation dataset consists of 600-dimensional i-vectors (as in the data for NIST i-vector challenge) created by our gender independent T-extractor based on the previous NIST SREs. These vectors were obtained from recordings of 500 speakers in the telephone channel, half of whom were men and half women. Each speaker has several sessions. The minimum number of sessions per speaker is one, the maximum is 50. The total number of i-vectors in the database is 5213.

3.2. Clustering error metrics

Let us introduce definitions that are relevant for the task of effective PLDA training on an unlabeled dataset.

1. The i-vector of the speaker *S* is considered to be clustered correctly if it belongs to a cluster in which the majority of vectors belong to the speaker *S*.
2. If there is more than one cluster in which there are vectors belonging to the speaker *S*, only vectors belonging to the cluster with most of i-vectors of the speaker *S* are considered to be clustered correctly.
3. In case such clusters have an equal number of vectors of the speaker *S*, we consider vectors of only one cluster to be clustered correctly.

Let us consider the value of clustering purity Q which characterizes the portion of correctly clustered i-vectors in the general dataset:

$$Q = 100\frac{M_{true}}{M}\,[\%], \tag{21}$$

where M_{true} is the number of correctly clustered vectors; M is the total number of vectors in the dataset.

Let us consider two clustering errors that influence training quality of the PLDA model.

- Err_{ass} is the error of assigning i-vectors of different speakers to one cluster.
- Err_{sep} is the error of separating the i-vector set of one speaker into several clusters.

Let us define a "clean" speaker cluster as a cluster which contains only the i-vectors of one speaker. Let us define a "contaminated" cluster as a cluster which contains i-vectors of different speakers (even only one i-vector of a different speaker). Then the error of assigning i-vectors of different speakers to one cluster equals:

$$Err_{ass} = 100\,\frac{N_{con}}{N_{all}}\,[\%], \tag{22}$$

where N_{con} is the number of contaminated clusters after performing the clustering; N_{all} is the number of found clusters. The error of separating the i-vector set of one speaker into several clusters equals:

$$Err_{sep} = 100\frac{N_{clear}^{bad}}{N_{all}}\,[\%], \tag{23}$$

where N_{clear}^{bad} is the number of erroneous clean speaker clusters that appeared as a result of dividing the vector set of one speaker into several clusters.

Let us define the total clustering error as:

$$Err_{sum} = Err_{sep} + Err_{ass}. \tag{24}$$

3.3. Clustering algorithm for the PLDA subsystems

For automatic i-vector segmentation into speaker clusters we used our own modification of the classic Agglomerative Hierarchical Clustering (AHC) algorithm [21]. AHC has been widely used as a speaker clustering strategy in many speaker diarization systems [22-24].

In order to perform i-vectors clustering, we need to solve the problem of choosing the similarity measure of i-vectors based on the nature and specific characteristics of the i-vector space. From verification tasks it is well-known that:

- First, the cosine metric [14] is a convenient comparison metric in the i-vector space that does not require training.
- Second, the model of averaging normalized i-vectors (searching for the speaker center [25,26]) is considered the most efficient multi-session model.

It follows that for initial clustering it is convenient to use the cos metric. After this initial clustering step, it makes sense to use the more efficient PLDA metric (5), which explicitly takes into account between-speaker and within-speaker covariance. This idea leads to an iterative clustering algorithm, which will be described in 3.2. Consequently, the similarity measure φ of i-vectors is defined as:

$$\varphi(\boldsymbol{x}, \boldsymbol{c}) = cos(\boldsymbol{x}, \boldsymbol{c}) \tag{25}$$

or

$$\varphi(\boldsymbol{x}, \boldsymbol{c}) = score_{PLDA}(\boldsymbol{x}, \boldsymbol{c})\,. \tag{26}$$

Our two-stage algorithm for speaker clustering in the i-vector space is given below.

At the first stage, cluster search with the threshold τ_1 is performed.

Algorithm 1 (*Data*, τ_1)
Input data:

*Data***[***L*x*M***]** are the whitened and normalized i-vectors that need to be labeled (*L* is the i-vector dimension; *M* is the number of vectors), τ_1 is the threshold for including the vector $\boldsymbol{x}$ into a cluster with the center $\boldsymbol{c}$:

$\varphi(\boldsymbol{x}, \boldsymbol{c}) > \tau_1$ – including $\boldsymbol{x}$ into the cluster with the center $\boldsymbol{c}$

$\varphi(\boldsymbol{x}, \boldsymbol{c}) \leq \tau_1$ – not including $\boldsymbol{x}$ into the cluster with the center $\boldsymbol{c}$

Output data:

- *Cluster Data* **[***L*x*M***]** are the clustered vectors;
- *Labels* [1x*M*] are the clustering labels, $Labels(m) \in [1, \dots, n]$, where *n* is the number of found clusters.

Init: $n \coloneqq 1$;

while $Data \neq \emptyset$

1. We randomly select one of the *Data* vectors as the center $\boldsymbol{c}_n^1$ of the current cluster $\boldsymbol{C}_n$.
   Init: $\Delta c := 1; t := 1;$
2. **while** $\Delta c \neq 0$
   2.1 Using the threshold τ_1 we collect vectors into $\boldsymbol{C}_n$ so as $\forall\, \boldsymbol{x}_k \in \boldsymbol{C}_n : \varphi(\boldsymbol{x}_k, \boldsymbol{c}_n^t) > \tau_1$;
   2.2 We re-evaluate the normalized center of the current cluster

$$\boldsymbol{c}_n^{t+1} = \frac{1}{K}\sum_{k=1}^{K} \boldsymbol{x}_k;$$ (*K* is the number of vectors in the cluster $\boldsymbol{C}_n$)

$$\boldsymbol{c}_n^{t+1} = \frac{\boldsymbol{c}_n^{t+1}}{\|\boldsymbol{c}_n^{t+1}\|}.$$

   2.3 We calculate $\Delta c_n = 1 - cos(\boldsymbol{c}_n^t, \boldsymbol{c}_n^{t-1})$;
   2.4 $t := t + 1;$

   end
3. We move the vectors of the cluster $\boldsymbol{C}_n$ from the *Data* set to the clustered set *Cluster Data* and add the label of the new cluster for these vectors to *Labels*;
4. $n := n + 1;$

end

Our first stage of the clustering is in fact similar to the algorithm proposed in [27, 28], which implemented an extension of the standard Mean Shift (MS) algorithm [29] to MS based on the cos distance:

$$d(\boldsymbol{x}, \boldsymbol{c}) = 1 - cos(\boldsymbol{x}, \boldsymbol{c})\ . \qquad (27)$$

At the second stage we combine the obtained clusters with the threshold τ_2. The cluster with the center $\boldsymbol{c_k}$ and the cluster with the center $\boldsymbol{c_j}$ have to be combined if the following condition is met:

$$cos(\boldsymbol{c_k}, \boldsymbol{c_j}) \geq \tau_2. \qquad (28)$$

At this stage we used simple Repeat-Until loop algorithm.

This stage is necessary to compensate for the error of dividing a set of vectors for one speaker into several clusters.

3.4. Clustering quality

In this section we examine the dependence between clustering quality and its parameters, using the cos metric as an example.

In our in-House experiments we obtained the dependence Q on the threshold values τ_1 and τ_2, which is illustrated in Figure 4. The maximum value of Q obtained on the in-House database using the proposed algorithm is obtained with τ_1 = 0.27 and $\tau_2 \geq 0.27$, and equals 85%.

Figure 5 demonstrates the dependence of the total error Err_{sum} on the value of the threshold τ_2 , when τ_1 = 0.27. This condition maximizes Q. The figure shows that when the threshold value is $0.2 < \tau_2 < 0.3$ the total error reaches the minimum value.

The condition of not using the second Bottom-Up stage is the condition $\tau_2 = 1$. As can be seen from Figure 5, the presence of the minimum Err_{sum} forces us to use this second clustering stage. We also confirmed these conclusions after making several submissions on the NIST i-vector challenge progress set.

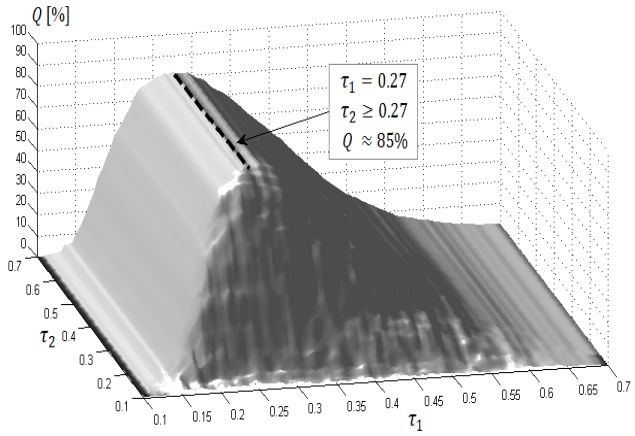

Figure 4. *The dependence of the clustering purity Q on the value of the thresholds* τ_2 *,and* τ_1

Besides, it follows from Figures 4 and 5 that it is sufficient to use the condition $\tau_2 \cong \tau_1$ in order to achieve the optimum both for Err_{sum} and for *Q*.

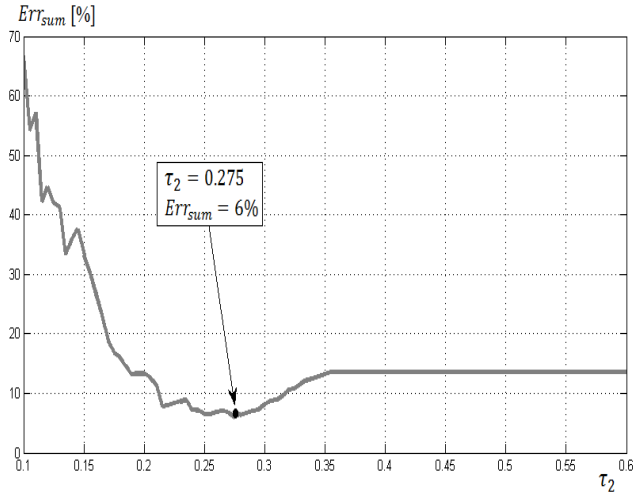

Figure 5. *The dependence of the error* Err_{sum} *on the value of the threshold* τ_2.

We would like to note that the experiments performed on an in-House data allowed us to formulate our clustering strategy. But the selection of the thresholds τ_1 and τ_2 for the NIST i-vector challenge devset clustering was performed experimentally by means of several submissions on the NIST i-vector Challenge progress set.

3.5. PLDA clustering

As mentioned above, in our work we used both stages of our clustering algorithm, with the cos metric and with the PLDA metric. Figure 6 shows the iterative scheme that we used for clustering the NIST i-vector Challenge devset.

Before using this scheme, the NIST i-vector Challenge devset was preprocessed to construct the $devset^{(1.0)}$. To do that,

we selected only those i-vectors that were produced from devset segments longer than 20 seconds.

At the initialization step of the PLDA clustering ("COS Clustering" block in Figure 6) we applied our clustering algorithm with the cos metric to the $d$evset$^{(1.0)}$ 16 times, with 16 random clustering initializations. The thresholds for the two stages $\tau_2 = \tau_1 = 0.29$ were found by several submissions. As a result of 16 passes we obtained 16 sets of speaker clusters. The intersection of these 16 sets gave us a raw set of clusters. This strategy allowed us to lower the dependence on the random choice of the initial point of our clustering algorithm and to make the raw set more robust. This raw set was then post-processed in the following way. We selected from it only the speaker clusters that contained no less than 2 and no more than 50 i-vectors, thus obtaining $d$evset$^{(1.1)}$. As demonstrated by the results of our progress set submissions, such a choice of the interval of possible cluster sizes led to the minimum minDCF. It follows that the proposed cluster post-processing method results in clustering purity that provides efficient PLDA model training.

The resulting set included 1542 speaker clusters containing 8682 i-vectors. Then on this "cleaned" $d$evset$^{(1.1)}$ set we trained the PLDA model for the verification task. The configuration of this model was as follows: the number of eigenvoices for the V-matrix was $N_1 = 350$, and the number of eigenchannels for the U-matrix was $N_2 = 20$. The noise matrix Σ was diagonal. As a result of the submission on the NIST i-vector Challenge progress set (not shown in Figure 6) this PLDA model achieved minDCF = 0.293 without using any QMF function. The choice of such a model configuration for the verification task is motivated by high labeling noise which will inevitably be present at the initialization step. To make the PLDA model more robust it was necessary to use a diagonal covariance and minimize the number of eigenchannels.

The first iteration of the PLDA clustering starts with ML training of the PLDA model on $d$evset$^{(1.1)}$ (see Formula (4)) for the clustering task (see "PLDA Training" block in Figure 6). The configuration of this model was as follows: the number of eigenvoices $N_1 = 300$, and the number of eigenchannels $N_2 = 0$. The noise matrix Σ was full covariance. Interestingly, submissions on the NIST i-vector challenge progress set showed that choosing a full-covariance PLDA model for the clustering task turned out to be more efficient than choosing the diagonal one.

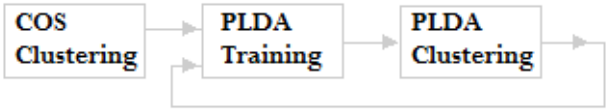

Figure 6: *The proposed scheme for clustering the NIST i-vector Challenge devset.*

Then we used this model for clustering $d$evset$^{(1.0)}$ and applied our clustering algorithm with the PLDA metric (see "PLDA Clustering" block in Figure 6). This process of PLDA re-clustering can be continued iteratively, as shown in Figure 6. We performed the first stage of our algorithm with $\tau_1 = -0.20$. However, the second stage was again performed with the cos metric (see Formula (28)) with the threshold $\tau_2 = 0.27$. These thresholds were found simply by several submissions. We should note that, in contrast to the initialization step, here we used not 16 initial points but only one. This departure from the sensible strategy of the initialization step was motivated simply by saving our resources.

After this PLDA clustering, we obtained new speaker clusters, so that their number reached 2492 clusters, and the total number of i-vectors in these clusters was 17186. After that we performed the same post-processing as at the initialization step, but with a different lower boundary. In this case we only used speaker clusters that contained no less than 3 and no more than 50 i-vectors. In this way we obtained the $d$evset$^{(1.2)}$set consisting of 1745 speaker clusters with 13093 i-vectors. This $d$evset$^{(1.2)}$ was used to train the PLDA model with the following verification configuration: $N_1 = 350$, $N_2 = 55$, Σ was diagonal. In this PLDA model configuration, increasing the number of eigenchannels to $N_2 = 55$ was motivated by the decrease in clustering noise, which enabled us to make the verification model more robust. The result of the submission on the NIST i-vector Challenge progress set was that this PLDA model achieved minDCF = 0.288 without using any QMF function.

In our work only the one iteration of the iterative PLDA clustering was performed. Increasing the number of iterations further made them difficult to control, which meant difficulty in finding the optimal model configuration at each iteration.

3.6. Clustering for the SVM subsystem

As mentioned before, the LDA-SVM subsystem used its own clustering algorithm. This algorithm is in fact the Bottom-Up stage of the classic AHC, but details of its implementation differ from the algorithm for the PLDA and RBM-PLDA subsystems described above. Let us call it Algorithm 2, in contrast to Aglorithm 1 from Sections 3.3 and 3.5.

This overlap in Algorithms 1 and 2 is caused by the independent development of the subsystems by different authors of this paper during the whole time of the NIST i-vector Challenge. As demonstrated by the subsequent fusion of the subsystems, using different clustering algorithms increases fusing efficiency. Clustering was performed only on those data from the development set that had speech duration longer than 10 seconds. This constraint was selected a priori based on the assumption that short recordings have very noisy i-vectors that are difficult to cluster. We will call this set $d$evset$^{(2.0)}$.

The clustering algorithm consisted of the following steps:

Algorithm 2

1. Each i-vector was taken as a separate cluster.
2. The two closest clusters were merged into one. The degree of similarity between clusters was defined as the value of the cos metric between the "averaged" i-vectors of these clusters.
3. For the merged cluster, the "averaged" i-vector was recalculated as the average value of all i-vectors in this cluster.
4. Steps 2-3 were repeated while the value of the cos metric was greater than the threshold 0.45.

The threshold for stopping clustering at the cos metric value of 0.45 was based on the results of the submissions of the SVM-LDA subsystem.

The result of this cos clustering was a set of about 3000 speaker clusters containing over 25000 i-vectors. After that, just as in Section 3.5, a PLDA model was trained on this set. The configuration of this model was as follows: the number of eigenvoices was $N_1 = 300$, the number of eigenchannels $N_2 = 0$. However, in contrast to the clustering algorithm in Section 3.5, the noise matrix Σ was diagonal. In fact, we performed PLDA clustering that corresponded to only the first PLDA iteration of Section 3.5.

Then we used this model for obtaining the scores (5) on $d\text{evset}^{(2.0)}$. We applied Algorithm 2 to these scores using the PLDA metric and the threshold 0.43. The threshold for stopping clustering at the PLDA metric value of 0.43 was based on the results of the submissions of the SVM-LDA subsystem. The final clusters were filtered by size (number of merged i-vectors): clusters smaller than 2 and larger than 30 were deleted. The final clustering was used for LDA module training.

4. Final experiments

The final experiments were conducted on the NIST i-vector Challenge data. The data available for the NIST i-vector Challenge are development data for training systems and a separate evaluation set for the Challenge. The speakers used in these datasets are disjoint. The i-vectors are obtained from spoken telephone speech in the NIST Speaker Recognition (SRE's) from 2004 to 2012. The dimension of the i-vectors is 600. Each vector has meta information, namely the amount of speech (in seconds) used to compute the i-vector. Segment durations were sampled from a *log normal* distribution with a mean of 39.58 seconds [1].

4.1. Development and evaluation sets of the NIST i-vector Challenge

Development data contain a very large number of unlabeled i-vectors obtained from segments of telephone speech. These vectors are constructed from telephone recordings of various male and female speaker voices.

Evaluation data consist of sets of 5 i-vectors defining the target speaker models and of single i-vectors representing test segments. The number of target speaker models is 1,306 (comprising 6,530 i-vectors) and the number of test i-vectors 9,634 (one i-vector each).

4.2. Trials for submission and scoring

The full set of trials for the Challenge consists of all possible pairs involving a target speaker model and a single i-vector test segment. Thus the total number of trials is 12,582,004.

The trials are divided into two subsets: progress subset and evaluation subset. The progress subset comprises 40% of the trials and is used to monitor progress in the scoreboard. The remaining 60% of the trials forms the evaluation subset, and will be used to generate the official final scores determined at the end of the Challenge.

4.3. The influence of QMF functions

Even though, as described in Section 2.6, we explored QMF functions only for the PLDA verification subsystem, we decided to use the linear form of QMF (17) in all our other subsystems.

Table 1 shows the minDCFs of different subsystems with and without QMF functions calculated with Formulas (18), (19) and (20). Table 1 shows that using QMF for the SVM subsystem leads to the highest reduction of minDCF by 10%. The same reduction for PLDA and RBM-PLDA is 2% and 1.5% respectively. The last row of the table shows the results for fusing all three subsystems. Using QMF functions for the resulting fused system results in minDCF = 0.241, which means a 7% minDCF reduction.

It is obvious that for this NIST i-vector Challenge, taking into account the variations in test utterance duration is one of the key issues.

Table 1. *Experimental results for the NIST i-vector Challenge.*

Method of model estimation	**minDCF on progress set**	
	without QMF	**with QMF**
LDA-SVM subsystem	0.286	0.259
PLDA subsystem	0.288	0.282
RBM-PLDA subsystem	0.293	0.289
Fusion ($w_{LDA-SVM} = w_{RBM-PLDA} = w_{DBN-PLDA} = 0.33$)	**0.259**	**0.241**

Table 1 also demonstrates that regardless of QMF, the most efficient of our subsystems was SVM, which is somewhat at odds with the state-of-the-art in past NIST SRE's, where discriminative SVM-based systems were outperformed by generative PLDA-based systems. However, in the NIST i-vector Challenge, a PLDA system requires a perfect labeling of the development set in order to outperform an SVM system, which must be very difficult to accomplish in practice. In contrast, an SVM system can use the whole devset as an imposter set without clustering.

4.4. Fusing different configurations

Tables 2-5 demonstrate the results of fusing all our subsystems in different combinations. All results are shown with QMF functions. Tables 2-4 show that the best combination of two subsystems is fusing LDA-SVM and RBM-PLDA, which achieves minDCF = 0.241. The least efficient combination was fusing two PLDA subsystems (see Table 3), with minDCF = 0.263. We can explain it by two reasons.

First, the combinations of different PLDA subsystems with LDA-SVM used two different clustering algorithms, while the combination of PLDA and RBM-PLDA used the same clustering result obtained by Algorithm 1 from Section 3.5. This resulted in the two PLDA subsystems being more correlated than in combination with the LDA-SVM subsystem, so that fusing them was less effective.

Second, a comparison of Tables 2 and 4 shows that under equal clustering conditions fusing the RBM-PLDA and LDA-SVM subsystems is more efficient.

Table 2. *Experimental results for LDA-SVM and PLDA subsystems.*

Method of model estimation	**minDCF on progress set with QMF**
LDA-SVM subsystem	0.259
PLDA subsystem	0.282
Fusion ($w_{LDA-SVM}$= 0.51, w_{PLDA} = 0.49)	**0.252**

Table 3. *Experimental results for PLDA and RBM-PLDA subsystems*

Method of model estimation	**minDCF on progress set with QMF**
PLDA subsystem	0.282
RBM-PLDA subsystem	0.289
Fusion (w_{PLDA} = 0.5, $w_{RBM-PLDA}$= 0.5)	**0.263**

We find the explanation for this is that the RBM classifier performs a nonlinear transformation. It enables the transition into a new b-vector space. PLDA subsystems trained in the b-space are decorrelated with subsystems trained in the i-vector space, which leads to successful fusion. It should be noted in Tables 2-4 that by comparing the results of separate subsystems, we find that the RBM-PLDA subsystem based on i-vectors is inferior to both classic PLDA and SVM subsystems. This is in accordance with similar results obtained by [5-7], where methods using Boltzmann Machines in i-vector space are also outperformed by PLDA.

Table 4. *Experimental results for LDA-SVM and RBM-PLDA subsystems.*

Method of model estimation	**minDCF on progress set with QMF**
LDA-SVM subsystem	0.259
RBM-PLDA subsystem	0.289
Fusion ($w_{LDA-SVM}$ = 0.51, $w_{RBM-PLDA}$ = 0.49)	**0.241**

We also tried to include a second hidden layer to our RBM-PLDA model (referred as DBN-PLDA), but we observed that adding another hidden layer did not yield any substantial reduction at the minDCF point. Table 5 includes our best result minDCF = 0.239 obtained by fusing three subsystems LDA-SVM, RBM-PLDA and DBN-PLDA.

Table 5. *Experimental results for LDA-SVM, RBM-PLDA and DBN-PLDA subsystems.*

Method of model estimation	**minDCF on progress set with QMF**
LDA-SVM subsystem	0.259
RBM-PLDA subsystem	0.289
DBN-PLDA subsystem	0.290
Fusion ($w_{LDA-SVM}$= $w_{RBM-PLDA}$ = $w_{DBN-PLDA}$= 0.33)	**0.239**

5. Conclusions

In this paper we presented the STC NIST i-vector Challenge speaker verification system, which includes different subsystems based on PLDA, LDA-SVM, RBM-PLDA and DBN-PLDA.

We proposed a version of agglomerative clustering in i-vector space for use as the clustering algorithm for the NIST i-vector Challenge devset, based on PLDA iterations. Non-linear transformation of the TV i-vector is performed using RBM, which leads to successful fusion with classic i-vector systems. Experiments conducted on the NIST i-vector evaluation set show that fusing LDA-SVM, RBM-PLDA and DBN-PLDA subsystems is the best option. It enabled us to achieve minDCF = 0.239.

In our future work we plan to focus on exploring different DNN configurations as pseudo i-vector extractors.

6. Acknowledgments

We would like to thank the reviewers of the first version of this paper for their valuable remarks and suggestions, including references to literature on clustering.

7. References

[1] The 2013-2014 Speaker Recognition i-vector Machine Learning Challenge, http://www.nist.gov/itl/iad/mig/upload/sre-ivectorchallenge_2013-11-18_r0.pdf

[2] S. J. D. Prince, "Probabilistic linear discriminant analysis for inferences about identity," in Proc. International Conference on Computer Vision (ICCV), Rio de Janeiro, Brazil, 2007.

[3] P. Kenny, "Bayesian speaker verification with heavy-tailed priors," in Proc. Odyssey-2010, 2010.

[4] D. Garcia-Romero and C. Y. Espy-Wilso, "Analysis of i-vector length normalization in speaker recognition systems," in Proc. of Interspeech-2011, Florence, Italy, Aug. 2011.

[5] M. Senoussaoui, N. Dehak, P. Kenny, R. Dehak, and P. Dumouchel, "First attempt at Boltzmann Machines fo speaker recognition," in Proc. of Odyssey-2012, pp. 117–121, 2012.

[6] T. Stafylakis, P. Kenny, M. Senoussaoui, and P. Dumouchel, "Preliminary investigation of Boltzmann Machine classifiers for speaker recognition," in Proc. of Odyssey-2012, pp. 109–116, 2012.

[7] T. Stafylakis, P. Kenny, M. Senoussaoui, P.Dumouchel, "PLDA using Gaussian Restricted Boltzmann Machines with application to Speaker Verification," in Proc. of Interspeech- 2012, Portland, USA, September 2012.

[8] V. Vasilakakis, S. Cumani and P. Laface, "Speaker recognition by means of Deep Belief Networks," Biometric Technologies in Forensic Science, Nijmegen, October, 2013.

[9] Y. Lei, N. Scheffer, L. Ferrer and M. McLaren, "A novel scheme for speaker recognition using a phonetically aware Deep Neural Network," in Proc. of ICASSP-2014, 2014.

[10] P. Kenny, V. Gupta, T. Stafylakis, P. Ouellet and J. Alam "Deep Neural Networks for extracting Baum-Welch statistics for Speaker," in Proc. of Odyssey-2014, 2014.

[11] T. Pekhovsky, A. Sizov, "Comparison Supervised and Unsupervised Learning Mixture of PLDA Models for Speaker Verification ", Pattern Recognition Letters, v.34, pp.1307–1313 (Apr. 2013)

[12] N. Dehak, P. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, "Front-end factor analysis for speaker verification," IEEE Trans. on Audio, Speech, and Language Processing, v.19, pp. 788-798., 2010.

[13] A. Kozlov, O. Kudashev, Y. Matveev, T. Pekhovsky, K.. Simonchik, A. Shulipa, "SVID Speaker Recognition System for NIST SRE 2012," In Proc. of International Conference SPECOM-2013, pp. 278-285, Springer International Publishing, 2013.

[14] N. Dehak et al., "Support Vector Machines versus Fast Scoring in the Low-Dimensional Total Variability Space for Speaker Verification," in Proc. of Interspeech-2009, Brighton, UK, 2009.

[15] I. N. Belykh, A. I. Kapustin, A. V. Kozlov, A. I. Lohanova, Yu. N. Matveev, T. S. Pekhovsky, K. K. Simonchik, A. K. Shulipa, "The speaker identification system for the NIST SRE 2010", Informatics and its Applications, 6 (1):24-31, 2012.

[16] S. Novoselov, T. Pekhovsky, A.Shulipa, A. Sholokhov, "Text-dependent GMM-JFA system for password based speaker verivication," of in Proc. of ICASSP 2014, Florence, Italy, May 2014.

[17] G. Hinton, L. Deng, D. Yu, G. Dahl, A. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. Sainath, and B. Kingsbury, "Deep neural networks for acoustic modeling in speech recognition," IEEE Signal Processing Magazine, v. 29 (6), pp. 82-97, 2012.

[18] Hinton, G. E., Osindero, S., & Teh, Y. (A fast learning algorithm for deep belief nets. Neural Computation, v. 18, pp. 1527–1554, 2006.

[19] H. Larochelle, Y. Bengio, "Classification using discriminative restricted Boltzmann machines," Proc. of the 25-th International Conference on Machine Learning, pp. 536–543, Helsinki, Finland, 2008.

[20] M. I. Mandasari, R. Saeidi and D. A. van Leeuwen, Calibration based on duration quality measure function in noise robust speaker recognition for NIST SRE'12, in Proc. Biometric Technologies in Forensic Science, Nijmegen, Oct. 2013.

[21] R. O. Duda, P. E. Hart, and D. G. Stork, "Pattern classification", 2-nd edition, John Wiley & Sons, 2001.

[22] Kyu Jeong Han, Shrikanth S. Narayanan, "Agglomerative hierarchical speaker clustering using incremental Gaussian mixture cluster modeling", in Proc. of Interspeech-2008 pp.20-23, Brisbane, Australia, 2008.

[23] D. A. Reynolds and P. Torres-Carrasquillo, "The MIT Lincoln Laboratory RT-04F Diarization Systems: Applications to Broadcast Audio and Telephone Conversations," NIST Rich Transcription Workshop, Nov. 2004.

[24] S. Tranter and D. Reynolds, "An overview of automatic speaker diarization systems," IEEE Transactions on Audio, Speech, and Language Processing, v.14, n.5, pp.1557–1565, Sept. 2006.

[25] P. Rajan, T. Kinnunen and V. Hautamäki, "Effect of multicondition training on i-vector PLDA configurations for speaker recognition," in Proc. of Interspeech-2013, pp. 3694-3697, Lyon, France, Aug. 2013.

[26] K. Simonchik, T. Pekhovsky and A Shulipa, "Effective Estimation of a Multi-Session Speaker Model using Information on Signal Parameters", in Proc. of Interspeech-2013, pp. 1604-1608, Lyon, France, Aug. 2013.

[27] M. Senoussaoui, P. Kenny, P. Dumouchel and T. Stafylakis, "Efficient Iterative Mean Shift based Cosine Dissimilarity for Multi-Recording Speaker Clustering," in Proc. of ICASSP-2013, 2013.

[28] M. Senoussaoui, P. Kenny, T.Stafylakis and P. Dumouchel, "A Study of the Cosine Distance-Based Mean Shift for Telephone Speech Diarization," IEEE Trans. on Audio, Speech, and Language Processing, v. .22, n. 1, pp. 217-227, Jan. 2014.

[29] D. Comaniciu and P. Meer, "Mean shift: A robust approach toward feature space analysis," IEEE Trans. Pattern Analysis and Machine Intelligence, v. 24, n. 5, pp. 603 – 619, May 2002.