Interspeech 2018
2-6 September 2018, Hyderabad

# Supervised i-vector Modeling - Theory and Applications

*Shreyas Ramoji, Sriram Ganapathy*

Learning and Extraction of Acoustic Patterns (LEAP) Lab, Electrical Engineering,
Indian Institute of Science, Bengaluru, India

shreyasr@iisc.ac.in, sriramg@iisc.ac.in

## Abstract

Over the last decade, the factor analysis based modeling of a variable length speech utterance into a fixed dimensional vector (termed as i-vector) has been prominently used for many tasks like speaker recognition, language recognition and even in speech recognition. The i-vector model is an unsupervised learning paradigm where the data is initially clustered using a Gaussian Mixture Universal Background Model (GMM-UBM). The adapted means of the Gaussian mixture components are dimensionality reduced using the Total Variability Matrix (TVM) where the latent variables are modeled with a single Gaussian distribution. In this paper, we propose to rework the theory of i-vector modeling using a supervised framework where the speech utterances are associated with a label. Class labels are introduced in the i-vector model using a mixture Gaussian prior. We show that the proposed model is a generalized i-vector model and the conventional i-vector model turns out to be a special case of this model. This model is applied for a language recognition task using the NIST Language Recognition Evaluation (LRE) 2017 dataset. In these experiments, the supervised i-vector model provides significant improvements over the conventional i-vector model (average relative improvements of 5 % in terms of $C_{avg}$).

**Index Terms**: Supervised Expectation Maximization, Total Variability Matrix, i-vector Modeling, Gaussian Back-end.

## 1. Introduction

The popular paradigm of speaker and language recognition consists of modeling the database of speech recordings (in the form of a sequence of short-term feature vectors) with a Gaussian mixture model-universal background model (GMM-UBM) [1]. At the utterance level, the GMM-UBM model is then adapted at the level of mixture component means [2]. The initial approaches for speaker and language classification were replaced with the factor analysis modeling [3] where the adapted Gaussian mean components (spliced as a single high dimensional vector called the supervector) are expressed as a sum of speaker and session factors. The parameters in this model are derived using a maximum likelihood (ML) framework with an iterative expectation maximization (EM) approach.

The approach of joint factor analysis (JFA) was further simplified by a total variability modeling (TVM) where all variabilities were captured by a single fixed dimensional latent vector [4]. With a prior of standard normal distribution (having zero mean and identity covariance), the latent variables were called i-vectors. The i-vector features, extracted using a EM framework with a ML objective, were used for further processing in speaker/language/speech recognition systems. For example, the speaker verification systems use a probabilistic linear discriminant analysis (PLDA) [5] to model channel variability [6]. The language recognition systems with i-vectors used a cosine based scoring [7] or a support vector machine (SVM) model for language classification.

Recently, speech recognition systems have also incorporated i-vector features for speaker adaptation [8]. The replacement of GMM-UBM with a deep neural network (DNN) speech recognition front-end has also shown improvements in speaker/language recognition tasks [9, 10]. While normalization methods like length normalization have been proposed in the post processing of the i-vectors [11], all the efforts outlined above use the unsupervised ML framework for the training the i-vector models. A previous attempt was made to utilize supervision in i-vector learning by using the label information included with the supervector [12].

In this paper, we derive the i-vector modeling framework in a supervised setting. This modification involves the use of additional variable for the class label and the application of class dependent Gaussian mixture model (GMM) prior density for the latent variable where each mixture component of the GMM corresponds to a label. This choice of prior is motivated by the use of a Gaussian back-end [13], where the conventional i-vectors for each language are modeled with a single Gaussian distribution. The use of class dependent prior also allows us to weigh the importance of the prior with a factor. Instead of the approach of deriving i-vectors using a TVM followed by a Gaussian back-end [13], we show that the proposed supervised i-vector framework is able to perform these modeling steps with a joint EM based learning.

The proposed approach is applied for a language recognition task in the NIST Language Recognition Evaluation (LRE) 2017 dataset. For the test data (without labels), we extract the i-vector equivalent under each language specific prior and merge all the i-vectors into a single vector. A dimensionality reduction is applied using principal component analysis (PCA) to provide the final feature representation. These representations are used in SVM training for language classification. In our experiments comparing the supervised and unsupervised i-vector features, the proposed approach provides significant improvements in LRE task (average relative improvements of 5 % in $C_{avg}$ measure). Furthermore, we show that weighting the prior distribution on the latent variable appropriately can significantly improve the duration specific performance.

The rest of the paper is organized as follows. Sec. 2 describes the modeling framework of the conventional i-vectors. In Sec. 3, we provide the mathematical derivation of the proposed supervised EM framework for i-vector model parameter estimation. The language recognition experiments are reported in Sec. 5. This is followed by a brief summary in Sec. 6.

This work was funded partly by grants from the Defense Research Development Organization (DRDO) and the Department of Science and Technology (DST) Early Career Award.



10.21437/Interspeech.2018-2012

## 2. The Conventional i-vector Model

We provide the mathematical derivation of the conventional i-vector modeling [3]. This is required as the supervised i-vector modeling is built on this model. The notations used here follow those from the work of Kenny and Dehak et. al. [3, 4].

Given a dataset of $S$ recordings, let $X(s) = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{H(s)}]$ denote the sequence of $F$ dimensional front-end feature vectors, where $H(s)$ is the number of voiced frames in recording $s$. Let $\lambda = \{\pi_c, \boldsymbol{\mu}_c, \Sigma_c\}_{c=1}^{C}$ denote the parameters of a $C$ component Gaussian mixture universal background model (GMM-UBM). The UBM mean supervector is denoted by $\boldsymbol{M}_0 = [\boldsymbol{\mu}_1^\intercal, \ldots, \boldsymbol{\mu}_c^\intercal]^\intercal$. It is assumed that for each recording $s$, an adapted mean supervector $\boldsymbol{M}(s)$ was used to generate $X(s) = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{H(s)}]$. The total variability model (TVM) is a generative model for $\boldsymbol{M}(s)$ and is given by

$$\boldsymbol{M}(s) = \boldsymbol{M}_0 + T\boldsymbol{y}(s) \tag{2.1}$$

where $T$ is a matrix of dimension $CF \times R$ called the total variability matrix, and $\boldsymbol{y}(s)$ is a latent vector of dimension $R \times 1$. A standard normal distribution is used as the prior density for $\boldsymbol{y}(s)$. The i-vector of recording $s$ is defined as the MAP estimate of $\boldsymbol{y}(s)$ given $X(s)$. The Baum-Welch statistics of recording $s$ for mixture $c$ is given by

$$N_c(s) = \sum_{i=1}^{H(s)} \mathrm{p}_\lambda(c \mid \boldsymbol{x}_i) \tag{2.2}$$

$$\boldsymbol{F}_{X,c}(s) = \sum_{i=1}^{H(s)} \mathrm{p}_\lambda(c \mid \boldsymbol{x}_i)(\boldsymbol{x}_i - \boldsymbol{\mu}_c) \tag{2.3}$$

$$S_{X,X,c}(s) = \sum_{i=1}^{H(s)} \mathrm{p}_\lambda(c \mid \boldsymbol{x}_i)(\boldsymbol{x}_i - \boldsymbol{\mu}_c)(\boldsymbol{x}_i - \boldsymbol{\mu}_c)^\intercal \tag{2.4}$$

The BW statistics are sufficient statictics for estimation of the model parameters. In matrix form, the zeroth and first order statistics are written as:

$$N(s) = \begin{pmatrix} N_1(s)I & & 0 \\ & \ddots & \\ 0 & & N_C(s)I \end{pmatrix}, \quad \boldsymbol{F}_X(s) = \begin{pmatrix} \boldsymbol{F}_{X,1}(s) \\ \vdots \\ \boldsymbol{F}_{X,C}(s) \end{pmatrix}$$

where $I$ is an identity matrix of size $F \times F$. It can be shown that the log-likelihood function of $\boldsymbol{y}(s)$ is

$$\log \mathrm{p}_T(X(s) \mid \boldsymbol{y}(s)) = G(s) + H_T(s, \boldsymbol{y}(s)) \tag{2.5}$$

where $G(s)$ is a term containing the second order BW stats (Eq (2.4)). $G(s)$ is independent of $T$ and $\boldsymbol{y}(s)$, so it will not play a role in estimating $T$ and $\boldsymbol{y}(s)$. The second term is

$$H_T(s, \boldsymbol{y}(s)) = \boldsymbol{y}(s)^\intercal T^\intercal \Sigma^{-1} \boldsymbol{F}_X(s) - \frac{1}{2}\boldsymbol{y}(s)^\intercal T^\intercal \Sigma^{-1} N(s) T \boldsymbol{y}(s) \tag{2.6}$$

It can be shown that the *aposteriori* density function of $\boldsymbol{y}(s)$ given $X(s)$ is Gaussian with covariance $\mathcal{L}(s)^{-1}$ and mean $\mathcal{L}(s)^{-1} T^\intercal \Sigma^{-1} \boldsymbol{F}_X(s)$ where

$$\mathcal{L}(s) = I + T^\intercal \Sigma^{-1} N(s) T \tag{2.7}$$

Estimating $T$ by maximum likelihood is done by using the Expectation-Maximization algorithm.

### 2.1. Expectation (E) step:

The $Q$ function is given by:

$$\begin{aligned} Q\left(T \mid T^{(t)}\right) &= \sum_{s=1}^{S} \mathbb{E}_{\boldsymbol{y}(s)|X(s),T^{(t)}} \log \mathrm{p}_T(X(s), \boldsymbol{y}(s)) \\ &= \sum_{s=1}^{S} \mathbb{E}_{\boldsymbol{y}(s)|X(s),T^{(t)}} \log \mathrm{p}_T(X(s) \mid \boldsymbol{y}(s)) \\ &\quad + \sum_{s=1}^{S} \mathbb{E}_{\boldsymbol{y}(s)|X(s),T^{(t)}} \log \mathrm{p}(\boldsymbol{y}(s)) \end{aligned} \tag{2.8}$$

The second term is independent of $T$ and doesn't play a role in the M-Step, and hence it can be ignored. Substituting for the likelihood term, ignoring the irrelevant terms and simplifying yields:

$$\begin{aligned} Q\left(T \mid T^{(t)}\right) = \sum_{s=1}^{S} \Bigg[ &\operatorname{tr}\left\{ T^\intercal \Sigma^{-1} \boldsymbol{F}_X(s) \hat{\boldsymbol{y}}^{(t)}(s)^\intercal \right\} \\ &- \frac{1}{2} \operatorname{tr}\left\{ \Sigma^{-1} N(s) T E_{yy}^{(t)}(s) T^\intercal \right\} \Bigg] \end{aligned} \tag{2.9}$$

where

$$\mathcal{L}^{(t)}(s) = I + T^{(t)^\intercal} \Sigma^{-1} N(s) T^{(t)} \tag{2.10}$$

$$\hat{\boldsymbol{y}}^{(t)}(s) = \mathcal{L}^{(t)}(s)^{-1} T^{(t)^\intercal} \Sigma^{-1} \boldsymbol{F}_X(s) \tag{2.11}$$

$$E_{yy}^{(t)}(s) = \mathcal{L}^{(t)}(s)^{-1} + \hat{\boldsymbol{y}}^{(t)}(s) \hat{\boldsymbol{y}}^{(t)}(s)^\intercal \tag{2.12}$$

The E-step can be summarized as computing the quantities in Eq (2.10 - 2.12). The vector $\hat{\boldsymbol{y}}^{(t)}(s)$ is also the MAP estimate of the latent variable computed using the total variability matrix $T^{(t)}$ which is usually termed as the i-vector.

### 2.2. Maximization (M) step:

The update equation for the matrix $T$ is obtained by maximizing the $Q$ function w.r.t $T$.

$$T^{(t+1)} = \underset{T}{\operatorname{argmax}}\, Q\left(T \mid T^{(t)}\right) \tag{2.13}$$

Differentiating Eq (2.9), equating it to zero and simplifying yields the system of linear equations in every row of $T^{(t+1)}$:

$$\begin{aligned} \sum_{s=1}^{S} N(s) T^{(t+1)} \left( \mathcal{L}^{(t)}(s)^{-1} + \hat{\boldsymbol{y}}^{(t)}(s) \hat{\boldsymbol{y}}^{(t)}(s)^\intercal \right) \\ = \sum_{s=1}^{S} \boldsymbol{F}_X(s) \hat{\boldsymbol{y}}_{T^{(t)}}^\intercal(s) \end{aligned} \tag{2.14}$$

$T^{(t+1)}$ is obtained by solving the above systems of linear equations.

## 3. The Supervised i-vector Model

In the traditional i-vector approach, the TVM together with the GMM-UBM constitute a latent variable based generative model for the front end feature sequence $X(s)$. In the supervised TVM, we incorporate the class label information in this generative model by assuming that the sequence of front-end features $X(s)$ and the label $l(s)$ follow a joint distribution. Assuming that there are $L$ classes under consideration, and $\mathrm{p}(l)$ as the

prior probability of class $l$, we can write the prior distribution of the latent variable $\boldsymbol{y}$ as

$$\mathrm{p}(\boldsymbol{y}) = \sum_{l=1}^{L} \mathrm{p}(l)\mathrm{p}(\boldsymbol{y} \mid l) \tag{3.1}$$

where $\mathrm{p}(\boldsymbol{y} \mid l)$ is the prior distribution of $\boldsymbol{y}$ conditioned on class $l$. We shall assume the usual total variability model (Eq (2.1)) for the GMM-UBM adapted mean supervectors, and introduce a supervised flavour to it by making the following assumptions:

1. The *apriori* probability of observing label $l$ is uniform, i.e., $p(l) = \frac{1}{L} \quad \forall l \in \{1, \ldots, L\}$
2. The "class conditional prior" of the latent variable $\boldsymbol{y}(s)$ is Gaussian with mean $\boldsymbol{m}_l$ and identity covariance matrix. That is, for each $l \in \{1, \ldots, L\}$

$$\mathrm{p}(\boldsymbol{y}\,|\,l) = \frac{1}{(2\pi)^{\frac{R}{2}}} e^{\left(-\frac{1}{2}(\boldsymbol{y}-\boldsymbol{m}_l)^{\intercal}(\boldsymbol{y}-\boldsymbol{m}_l)\right)} \tag{3.2}$$

3. Given the latent variable $\boldsymbol{y}(s)$, $X(s)$ is conditionally independent on the class label $l(s)$

$$\mathrm{p}_T(l(s) \mid \boldsymbol{y}(s), X(s)) = \mathrm{p}_T(l(s) \mid \boldsymbol{y}(s)) \tag{3.3}$$

The class conditional means $\boldsymbol{m}_1, \ldots, \boldsymbol{m}_l$ can be parametrized along with the matrix $T$. We define the new parameter set as $\Theta = \{T, \boldsymbol{m}_1, \ldots, \boldsymbol{m}_l\}$

The likelihood function of $\Theta$ in terms of $X(s), l(s)$ and $\boldsymbol{y}(s)$ is

$$\begin{aligned}\mathrm{p}_\Theta(X(s), \boldsymbol{y}(s), l(s)) &= \mathrm{p}_\Theta(X(s), \boldsymbol{y}(s) \mid l(s))\mathrm{p}(l(s)) \\ &= \frac{1}{L}\mathrm{p}_\Theta(X(s) \mid \boldsymbol{y}(s))\mathrm{p}_\Theta(\boldsymbol{y}(s) \mid l(s))\end{aligned} \tag{3.4}$$

The complete data log likelihood is

$$\begin{aligned}&\sum_{s=1}^{S} \log \mathrm{p}_\Theta(X(s), \boldsymbol{y}(s), l(s)) \\ &\quad = S \log \frac{1}{L} + \sum_{s=1}^{S} \log \mathrm{p}_\Theta(X(s) \mid \boldsymbol{y}(s)) \\ &\qquad + \sum_{s=1}^{S} \log \mathrm{p}_\Theta(\boldsymbol{y}(s) \mid l(s))\end{aligned} \tag{3.5}$$

It can be shown that the *aposteriori* density function of $\boldsymbol{y}(s)$ given $X(s)$ and $l(s)$ is Gaussian with covariance $\mathcal{L}^{-1}(s)$ which is the same as Eq (2.7) and mean given by $\mathcal{L}^{-1}(s)\{\boldsymbol{m}_{l(s)} + T^{\intercal}\Sigma^{-1}\boldsymbol{F}_X(s)\}$.

The EM algorithm (supervised EM) is used to solve for the parameters $\Theta = \{T, \boldsymbol{m}_1, \ldots, \boldsymbol{m}_l\}$ which maximize the joint likelihood function $\sum_{s=1}^{S} \log \mathrm{p}_\Theta(X(s), l(s))$.

### 3.1. Expectation (E) Step:

The $Q$ function in terms of $X(s)$ and $l(s)$ is given by

$$\begin{aligned}Q\left(\Theta \,|\, \Theta^{(t)}\right) &= \sum_{s=1}^{S} \mathbb{E}_{\boldsymbol{y}(s)|X(s),l(s),\Theta^{(t)}} \log \mathrm{p}_\Theta(X(s), l(s), \boldsymbol{y}(s)) \\ &= \sum_{s=1}^{S} \mathbb{E}_{\boldsymbol{y}(s)|X(s),\Theta^{(t)}} \log \mathrm{p}_\Theta(X(s) \mid \boldsymbol{y}(s)) \\ &\quad + \sum_{s=1}^{S} \mathbb{E}_{\boldsymbol{y}(s)|X(s),\Theta^{(t)}} \log \mathrm{p}(\boldsymbol{y}(s) \mid l(s))\end{aligned} \tag{3.6}$$

Substituting the required terms from Eq (2.5, 3.2 & 3.5) and simplifying gives

$$\begin{aligned}Q\left(\Theta \,|\, \Theta^{(t)}\right) = \sum_{s=1}^{S} \Bigg[ &\mathrm{tr}\left\{T^{\intercal}\Sigma^{-1}\boldsymbol{F}_X(s)\hat{\boldsymbol{y}}_l^{(t)}(s)^{\intercal}\right\} \\ &- \frac{1}{2}\mathrm{tr}\left\{\Sigma^{-1}N(s)TE_{yy,l}^{(t)}(s)T^{\intercal}\right\}\Bigg] \\ &+ \sum_{l=1}^{L}\sum_{\substack{s=1 \\ l(s)=l}}^{S}\left(\hat{\boldsymbol{y}}_{\Theta^{(t)}}(s)^{\intercal}\boldsymbol{m}_l - \frac{1}{2}\boldsymbol{m}_l^{\intercal}\boldsymbol{m}_l\right)\end{aligned} \tag{3.7}$$

where $\mathcal{L}^{(t)}(s)$ is the same as established in Eq (2.10) and

$$\hat{\boldsymbol{y}}_l^{(t)}(s) = \mathcal{L}^{(t)}(s)^{-1}\left\{\boldsymbol{m}_{l(s)} + T^{(t)\intercal}\Sigma^{-1}\boldsymbol{F}_X(s)\right\} \tag{3.8}$$

$$E_{yy,l}^{(t)}(s) = \mathcal{L}^{(t)}(s)^{-1} + \hat{\boldsymbol{y}}_l^{(t)}(s)\hat{\boldsymbol{y}}_l^{(t)}(s)^{\intercal} \tag{3.9}$$

The term $\hat{\boldsymbol{y}}_l^{(t)}(s)$ represents a "class conditional i-vector" computed using the parameters $\Theta^{(t)}$.

### 3.2. Maximization (M) step:

The update equation for the matrix $T$ is same as in Eq (2.14), and the update equation for $\boldsymbol{m}_l$ can be obtained by partially maximizing the $Q$ function w.r.t $\boldsymbol{m}_l$ and is given by

$$\boldsymbol{m}_l = \frac{1}{S_l}\sum_{\substack{s=1 \\ l(s)=l}}^{S} \hat{\boldsymbol{y}}_l^{(t)}(s) \tag{3.10}$$

where $S_l$ is the number of utterances having class label $l$. It can be noted that $S = \sum_{l=1}^{L} S_l$.

### 3.3. Weighting of Priors

Although the proposed model incorporates label information, the algorithm only tries to maximize the joint likelihood. This does not ensure that the model is discriminative. This problem is evident when the duration of the recording is long and the first order BW statistics is large. The term $T^{(t)\intercal}\Sigma^{-1}\boldsymbol{F}_X(s)$ in Eq (3.8) would dominate, making $\boldsymbol{m}_{l(s)}$ negligible. In an attempt to make the proposed model more discriminative, the class conditional prior covariance (Eq (3.2)) can be scaled by a factor $\frac{1}{\lambda}$. By doing this, it can be shown that the E-step will be modified as follows:

$$\mathcal{L}^{(t)}(s) = \lambda I + T^{(t)\intercal}\Sigma^{-1}N(s)T^{(t)} \tag{3.11}$$

$$\hat{\boldsymbol{y}}^{(t)}(s) = \mathcal{L}^{(t)}(s)^{-1}\left(\lambda\boldsymbol{m}_l^{(t)} + T^{(t)\intercal}\Sigma^{-1}\boldsymbol{F}_X(s)\right) \tag{3.12}$$

The weighting parameter $\lambda$ may be fixed or made duration dependent. We report the performances of language recognition systems for different values of $\lambda$ in Sec. 5.

## 4. I-Vector Extraction Methods

In the traditional TVM, the i-vectors were defined as the MAP estimate of the latent variable $\boldsymbol{y}(s)$ given the front-end features $X(s)$ of recording $s$. This is given by

$$\hat{\boldsymbol{y}}(s) = \left(I + T^{\intercal}\Sigma^{-1}N(s)T\right)^{-1}T^{\intercal}\Sigma^{-1}\boldsymbol{F}_X(s) \tag{4.1}$$

Table 1: *Overall $C_{avg}$ scores for different i-vector and backend models on LRE 2017 development and evaluation datasets.*

| Model | LDA+SVM | | GB | |
|---|---|---|---|---|
| | Dev | Eval | Dev | Eval |
| Baseline | **0.309** | 0.187 | **0.287** | 0.201 |
| MMSE i-vectors using true labels, $\lambda = 1$ | 0.308 | 0.174 | 0.321 | 0.188 |
| PCA i-vector $\lambda = 1$ | 0.321 | 0.185 | 0.292 | **0.199** |
| PCA i-vector $\lambda = 3$ | 0.347 | **0.180** | 0.416 | 0.199 |
| MMSE i-vectors $\mathrm{p}(l\|X(s)) = \frac{1}{L}$ | 0.309 | 0.187 | 0.322 | 0.201 |

In the proposed model for a test utterance of unknown label, the posterior distribution of $\boldsymbol{y}$ turns out to be a mixture of Gaussians which is multi-modal. This is given by

$$\mathrm{p}(\boldsymbol{y}(s) \mid X(s)) = \sum_{l=1}^{L} \mathrm{p}(l \mid X(s))\, \mathrm{p}(\boldsymbol{y}(s) \mid X(s), l) \quad (4.2)$$

Hence, there would be more than one locally maximum *aposteriori* estimates, and choosing just one of them would be meaningless. However, if we have access to another model like a deep neural network which gives us the class posteriors $\mathrm{p}(l \mid X(s))$, we could use the minimum mean-square-error (MMSE) estimate of $\boldsymbol{y}(s)$ using:

$$\begin{aligned} \hat{\boldsymbol{y}}_{MMSE}(s) &= \sum_{l=1}^{L} \mathrm{p}(l \mid X(s))\, \mathbb{E}(\boldsymbol{y}(s) \mid X(s), l) \\ &= \sum_{l=1}^{L} \mathrm{p}(l \mid X(s)) \hat{\boldsymbol{y}}_l(s) \end{aligned} \quad (4.3)$$

MMSE i-vectors perform the best when the posteriors $\mathrm{p}(l \mid X(s))$ is close to 1 for all the true labels. To show the lower bounds on the costs ($C_{avg}$), we extract the MMSE i-vectors by setting $\mathrm{p}(l \mid X(s)) = 1$ for the true class $l$ and 0 for the other classes. This gives us the lower bounds on costs that can be achieved with when the posteriors denote the true class.

When we do not have access to the posteriors from another model, one way to extract i-vector like representations is by averaging the class conditional i-vectors over all the classes. This is a special case of MMSE i-vectors where $\mathrm{p}(l \mid X(s))$ is set to $\frac{1}{L}$. The costs ($C_{avg}$) computed this way could serve as an upper bound.

Another approach to extract i-vector like representations is concatenate the $L$ class conditional i-vectors (of size $R \times 1$) into one single vector of size $RL \times 1$ and then apply dimensionality reduction techniques such as principle component analysis (Referred as PCA i-vectors in Table. 1).

## 5. Experiments and Results

The NIST Language Recognition Evaluation (LRE) 2017 [14] datasets were used in all our experiments. The training dataset consists of 16205 files from fourteen closely related languages and dialects grouped into five clusters. The total duration of the train files was about 2069 hours. The dev dataset had 3661 files and eval dataset has 25451 files. Both dev and eval datasets contain files of duration 3, 10, 30 and 1000 sec. We trained all the systems using the LRE 2017 train dataset (LDC2017E22) and we report the evaluation metric $C_{avg}$ computed on the development and evaluation datasets (LDC2017E23).

Table 2: *Duration specific $C_{avg}$ scores for LDA+SVM system with different vlues of $\lambda$ computed on Dev and Eval datasets*

| Model | Dev $C_{avg}$ | | | Eval $C_{avg}$ | | |
|---|---|---|---|---|---|---|
| | 3 sec | 10 sec | 30 sec | 3 sec | 10 sec | 30 sec |
| Baseline | **0.527** | 0.271 | 0.131 | 0.35 | 0.14 | 0.071 |
| $\lambda = 1$ | 0.556 | 0.276 | 0.132 | 0.34 | 0.143 | 0.071 |
| $\lambda = 2$ | 0.693 | 0.302 | 0.132 | 0.337 | 0.137 | 0.071 |
| $\lambda = 3$ | 0.63 | 0.286 | 0.128 | **0.335** | 0.135 | 0.07 |
| $\lambda = 5$ | 0.577 | 0.272 | **0.121** | 0.341 | 0.134 | 0.071 |
| $\lambda = 10$ | 0.569 | **0.267** | 0.126 | 0.35 | **0.131** | **0.07** |

The front-end features used were bottleneck features of 80 dimensions from an automatic speech recognition (ASR) trained deep neural network (DNN) which was trained on switchboard and Fisher corpora. Speech activity detection (SAD) [15] was applied to retain only the voiced frames. A GMM-UBM of 2048 mixtures was trained using the bottleneck features with a subsample factor of 8. The $T$ matrix was randomly initialized, the class conditional means $\boldsymbol{m}_1, \ldots, \boldsymbol{m}_l$ were initialized to zero and TVM was trained using the supervised EM algorithm for 10 iterations.

We performed several experiments using different i-vector extraction strategies discussed in sec. 4. We extracted i-vectors using different values of $\lambda$ and used two different backends for obtaining the language log-likelihoods, namely the Gaussian backend (GB) and Support Vector Machines (SVM) with radial basis function (RBF) kernel.

In Table. 1, we compare the overall $C_{avg}$ scores (excluding the 1000 sec files) of different i-vector representations using the two backends. In Table. 2, we compare the duration specific scores for different models on the evaluation dataset. Here, "baseline" represents the conventional i-vector systems.

## 6. Summary and Conclusion

We have established the theory for supervised i-vector modeling and it can be noted that the conventional i-vector model is a special case of the proposed model where the class conditional means $\boldsymbol{m}_l$ is forced to $\mathbf{0}$ for all classes, and not updated as mentioned in Eq (3.10). The early results suggest that this approach provides features with much more information than conventional unsupervised i-vectors. Even simple techniques such as concatenating class conditioned i-vectors followed by PCA results in improved performance over the unsupervised i-vectors. Furthermore, weighting of priors are shown to not only improve the overall performance, but choosing the right value of $\lambda$ given the duration for extracting i-vectors can significantly improve the duration specific performance.

## 7. Acknowledgements

We would like to thank Dr. Omid Sadjadi for providing the python baseline system [16] as part of the NIST LRE 2017 which was helpful in our implementation.

# 8. References


[1] D. A. Reynolds and R. C. Rose, "Robust text-independent speaker identification using gaussian mixture speaker models," *IEEE transactions on speech and audio processing*, vol. 3, no. 1, pp. 72–83, 1995.

[2] D. A. Reynolds, T. F. Quatieri, and R. B. Dunn, "Speaker verification using adapted gaussian mixture models," *Digital signal processing*, vol. 10, no. 1-3, pp. 19–41, 2000.

[3] P. Kenny, "Joint factor analysis of speaker and session variability: Theory and algorithms," *CRIM, Montreal,(Report) CRIM-06/08-13*, vol. 14, pp. 28–29, 2005.

[4] N. Dehak, P. J. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, "Front-end factor analysis for speaker verification," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 4, pp. 788–798, 2011.

[5] S. J. Prince and J. H. Elder, "Probabilistic linear discriminant analysis for inferences about identity," in *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*. IEEE, 2007, pp. 1–8.

[6] L. Burget, O. Plchot, S. Cumani, O. Glembek, P. Matějka, and N. Brümmer, "Discriminatively trained probabilistic linear discriminant analysis for speaker verification," in *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on*. IEEE, 2011, pp. 4832–4835.

[7] N. Dehak, P. A. Torres-Carrasquillo, D. Reynolds, and R. Dehak, "Language recognition via i-vectors and dimensionality reduction," in *Twelfth annual conference of the international speech communication association*, 2011.

[8] G. Saon, H. Soltau, D. Nahamoo, and M. Picheny, "Speaker adaptation of neural network acoustic models using i-vectors." in *ASRU*, 2013, pp. 55–59.

[9] Y. Lei, N. Scheffer, L. Ferrer, and M. McLaren, "A novel scheme for speaker recognition using a phonetically-aware deep neural network," in *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*. IEEE, 2014, pp. 1695–1699.

[10] F. Richardson, D. Reynolds, and N. Dehak, "Deep neural network approaches to speaker and language recognition," *IEEE Signal Processing Letters*, vol. 22, no. 10, pp. 1671–1675, 2015.

[11] D. Garcia-Romero and C. Y. Espy-Wilson, "Analysis of i-vector length normalization in speaker recognition systems," in *Twelfth Annual Conference of the International Speech Communication Association*, 2011.

[12] M. Li and S. Narayanan, "Simplified supervised i-vector modeling with application to robust and efficient language identification and speaker verification," *Computer Speech & Language*, vol. 28, no. 4, pp. 940–958, 2014.

[13] D. Martinez, O. Plchot, L. Burget, O. Glembek, and P. Matějka, "Language recognition in ivectors space," in *Twelfth Annual Conference of the International Speech Communication Association*, 2011.

[14] Omid Sadjadi, Craig Greenberg. NIST 2017 Language Recognition Evaluation. https://www.nist.gov/itl/iad/mig/nist-2017-language-recognition-evaluation.

[15] J. Sohn, N. S. Kim, and W. Sung, "A statistical model-based voice activity detection," *IEEE Signal Processing Letters*, vol. 6, pp. 1–3, 1999.

[16] O. Sadjadi, "Official NIST language recognition python baseline system," NIST LRE 2017.