Deep Bottleneck Network based I-Vector Representation for Language Identification

Yan Song, Xinhai Hong, Bing Jiang, Ruilian Cui, Ian McLoughlin, Lirong Dai

1National Engineering Laboratory of Speech and Language Information Processing, University of Science and Technology of China, Hefei, China
{songy, lvm, lrdai}@ustc.edu.cn, {xhhong, cuirl, bing2010}@mail.ustc.edu.cn

Abstract

This paper presents a unified i-vector framework for language identification (LID) based on deep bottleneck networks (DBN) trained for automatic speech recognition (ASR). The framework covers both front-end feature extraction and back-end modeling stages. The output from different layers of a DBN are exploited to improve the effectiveness of the i-vector representation through incorporating a mixture of acoustic and phonetic information. Furthermore, a universal model is derived from the DBN with a LID corpus. This is a somewhat inverse process to the GMM-UBM method, in which the GMM of each language is mapped from a GMM-UBM. Evaluations on specific dialect recognition tasks show that the DBN based i-vector can achieve significant and consistent performance gains over conventional GMM-UBM and DNN based i-vector methods [1][2]. The generalization capability of this framework is also evaluated using DBNs trained on Mandarin and English corpuses.

Index Terms: Language Identification, Deep Neural Network, Deep Bottleneck Feature, i-vector representation

1. Introduction

The i-vector representation [1, 3] has received significant interest in both spoken language identification (LID) and speaker verification (SV) domains, due to its compact nature and ability to achieve excellent performance. Conventionally, the i-vector extraction procedure includes: 1) Front-end feature extraction, to convert a given utterance into a sequence of continuous-valued acoustic feature vectors, such as mel-frequency cepstral coefficients (MFCC), perceptual linear prediction (PLP) or shifted delta cepstra (SDC) information; 2) Back-end modeling, to construct a low dimensional representation using factor analysis, with respect to sufficient collected statistics, e.g. the zero-order and first-order Baum-Welch statistics using the occupancy posterior probability calculated on components of a pre-trained Gaussian mixture model-universal background model (GMM-UBM).

Recently, various extensions and additions have been proposed for the conventional i-vector extraction framework. Some authors have proposed collecting sufficient statistics using a deep neural network (DNN) pre-trained for automatic speech recognition (ASR) [4, 5], where better quality and quantity training data is available. In this framework, the sufficient statistics are computed by projecting the front-end features onto sonones (i.e., tied-states with context-dependent phones from an ASR decision tree) using the posterior probabilities estimated from the DNN. Compared to the conventional GMM-UBM based i-vector, this may provide benefit by exploiting the excellent modeling capabilities of DNNs, and provide better frame alignment for sufficient statistics calculation.

Other methods exploit alternative front-end features. For example, Hasan et al. proposed a factor analysis modeling scheme in the MFCC feature space, which performs dimension reduction, de-correlation, normalization and enhancement in a single linear transformation [6]. Diez et al. introduced phone log-likelihood ratios (PLLR) as front-end features for i-vector extraction, which achieved competitive performance as a stand-alone system for LID [7]. Meanwhile Song et al. proposed using deep bottleneck features (DBF), the output from a deep bottleneck network (DBN) which is a DNN with a constricted internal bottleneck layer, trained for ASR [8]. The resulting DBF based i-vector system achieved excellent performance for the NIST LRE2009 evaluation. Compared to conventional acoustic features, the DBF can effectively mine the contextual information embedded in speech frames by using the transcribed corpus.

Each of the above methods shows a certain degree of improvement over conventional i-vector approaches for LID or SV tasks, especially through the introduction of the pre-trained DNNs. In [9], it was shown that the DNN can be considered as a bridge from low-level spectral or acoustic features to high-level phonetic features, hence exploiting the output from different layers of the DNN may lead to improved utterance representation. In this paper, we adopt a similar approach, but apply it to i-vector extraction. For LID tasks, due to the lack of transcribed data, the DNN is effectively constrained to certain languages (e.g., English and Mandarin). Therefore the resulting phonetically-aware DNN may not be optimal as a universal model, especially for highly confusable languages. This motivates us to propose a unified DBN based i-vector extraction framework, as shown in Fig. 1. Specifically, using a DBN that has been well trained on a transcribed corpus, the output of different layers is first calculated. Then a universal model is derived from the sufficient statistics collected from the LID corpus. This is somewhat an inverse process to the GMM-UBM method for LID, in which the GMM of each language is mapped from a UBM. Finally, the i-vector representation can be extracted and applied for LID using standard methods. The contributions of this paper can be stated as follows:

- A unified i-vector framework is proposed, based on a single DBN which covers both front-end feature extraction and back-end-modeling stages by exploiting the output from different layers in the DBN. The framework thus incorporates a graded mixture of acoustic and phonetic information (from lower to higher layers respectively), leading to an enhanced i-vector.
representation.

- A universal model is derived from the sufficient statistics collected from the LID corpus using the pre-trained DBN. This reverse mapping process makes the DBN based i-vector feasible for LID tasks.

In summary, the resulting DBN based i-vector will be shown to significantly outperform the current best-performing method w.r.t a pre-trained GMM-UBM. For the DNN based i-vector method, senones are used as the components for sufficient statistics calculation: $\gamma_{k,t}$ is the senone posterior of the $t$-th frame on the $k$-th senone, and $m$ is derived from the speech frames assigned to senones, as described in [4, 5].

Li et al. further investigated various configurations that use the phoneme posteriors estimated from different phone recognizers (PR), termed phoneme-MFCC [11]. It was shown that the performance of phoneme-MFCC is slightly worse than the conventional i-vector for LID task. A similar conclusion could be drawn for SV, except for an i-vector using an English PR, probably due to the fact that most of the SV utterances are spoken English [11]. These results may provide evidence that the effectiveness of the phonetically-aware i-vector may be constrained by the PR that is used to calculate the posterior probability, i.e., $\gamma_{k,t}$ in eqn.(1). Therefore it follows that improving the i-vector representation may be possible by deriving a language independent universal model from a specific PR.

Besides the posterior probability calculation, the front-end feature $x_t$ also plays an important role in i-vector representation. In our previous work [8], it was shown that the DBF can significantly outperform conventional spectral features, such as MFCC and SDC, on the NIST LRE09 task. In the current paper, we aim to combine the front-end (spectral) feature extraction and back-end (phonetic) modeling stages in a unified DBN based i-vector framework, which will be detailed in the following sections.

3. DBN based i-vector for LID

In this section, we first introduce the DBN structure for ASR, and then derive a universal model from the output of the middle and final DBN layers. Finally, the unified i-vector is extracted and then used for LID in the same way as a conventional i-vector.

3.1. DBN structure

The DBN structure we used has 7 layers comprising 1 input layer, 5 hidden layers, and 1 output layer, which is configured as: $n \times d = 2048 - 2048 - d_m = 2048 - 2048 - d_q$, where $n \times d$, $d_m$, and $d_q$ are the sizes of input, bottleneck, and output layer respectively. For each speech frame, a $d$ dimensional acoustic feature is first extracted. The $n \times d$ dimensional input feature is then obtained by concatenating $n$ frames centered around the current one. The output layer contains $d_q$ nodes corresponding to senones which are automatically determined by a decision tree using maximum likelihood [12].

Two types of DBN are evaluated. The first DBN is trained on a Mandarin corpus of about 1000 hours, termed Mandarin-DBN. The configuration of Mandarin-DBN is $21 \times 43 \times 2048 - 2048 \times 43 \times 2048 \times 2048 - 6004$. Each frame feature comprises 39-dimensional MFCC+ΔMFCC+ΔΔMFCC, and 4-dimensional pitch features corresponding to the static pitch, 1st and 2nd derivatives and voiced speech confidence respectively. The input feature is a concatenation of the 21 frames centered around the current one. The middle layer size $d_m$ is set to 43 and the output layer size $d_q$ is set to 6004, according to the ASR performance.

The second DBN is trained on about 300 hours of Switchboard corpus, termed English-DBN. The configuration of English-DBN is $21 \times 48 \times 2048 \times 2048 \times 50 \times 2048 \times 2048 - d_q$. For each speech frame, a 48 dimensional feature
is first extracted, which comprises 39-dimensional dimensional PLP + ΔPLP + ΔΔPLP and 9 dimensional pitch features, and their 1st and 2nd derivatives. Then a $21 \times 48 = 1008$ input feature is obtained by concatenating 21 frames centred around the current one. The DBN is trained using the Kaldi speech recognition toolkit [13]. The middle layer size $d_m$ is set to 50. The output layer size $d_o$ is either 1536 or 2444, obtained by setting the decision tree threshold.

3.2. DBN outputs

If $y_p = \{y_1, \ldots, y_{M_p}\}$ is the output of the $p$-th layer with $M_p$ nodes, the output $y_p$ can be calculated as

$$y_i = f(x; \theta) = \sum_{j=1}^{M_p} w_{ij} \delta \left( - \sum_{k=1}^{M_2} w_{ik} \delta \left( \sum_{d=1}^{M_1} w_{kd} x_d \right) \right)$$

(5)

where $w_{ij}$ is the real-valued weight between visible unit $k$ and hidden unit $j$ in the $i$-th layer, $x = \{x_1, x_2, \ldots, x_{M_1}\}$ denotes the input feature, $\theta$ is the parameter set of the pre-trained DBN structure, $\delta(x) = \frac{1}{1 + e^{\gamma x}}$ represents the logistic sigmoid activation function.

Let $y_l = \{y_1, \ldots, y_{M_l}\}$ denote the output of the last layer, the posterior probability of the $k$-th node can be calculated using the soft-max function

$$\gamma_k = \frac{\exp(y_k)}{\sum_{l=1}^{M_l} \exp(y_l)}$$

(6)

It is shown that in the DNN, the output features from the first layer are general, while the features from the last layer are specific, depending greatly on the chosen dataset and task [14]. In our case, the output from the middle layer, i.e., DBF, may be considered to be general, capturing contextual information that we know is useful for various recognition tasks, such as ASR and LID. The output feature from the last layer, by contrast, is senone posterior, which are specific to ASR.

3.3. DBN based universal model

As shown in Fig. 1, a universal model is derived from the output of different layers in the DBN. Let $x_l$ be the DBF, and $\gamma_{k,t}$ be senone posteriors. Let $\lambda = \{\lambda_1, \ldots, \lambda_K\}$ be a universal model with $K$ components. The parameters of the $k$-th component $\lambda_k = \{w_k, \mu_k, \sigma_k\}$ can be computed from $x_l$ and $\gamma_{k,t}$

$$N_k = \sum_{t=1}^T \gamma_{k,t}$$

(7)

$$w_k = \frac{N_k}{\sum_{j=1}^K N_j}$$

(8)

$$\mu_k = \frac{1}{N_k} \sum_{t=1}^T \gamma_{k,t} x_t$$

(9)

$$\sigma_k = \frac{1}{N_k} \sum_{t=1}^T \gamma_{k,t} x_t^2 - \mu_k \mu_k^T$$

(10)

This process is similar to the supervised UBM method described in [2]. The major difference is that we use the LID dataset for deriving the universal model. This can also be considered as an inverse mapping process of the traditional GMM-UBM method, in which each language model is mapped from a UBM. With the derived universal model $\lambda$, the i-vector representation can then be extracted with standard methods by using eqns.(1)-(4) in section 3.

3.4. I-vector representation for LID

After the i-vector representation, two intersession compensation techniques are applied. The first step is within-class covariance normalisation (WCCN), which normalizes the i-vector with the inverse of the within-class covariance. The second step is linear discriminative analysis (LDA), a popular dimension reduction techniques for removing noise. After these two steps, the language model can be represented as the center of the corresponding i-vectors. Given a test utterance, the confidence score is calculated as the cosine distance from each model.

4. Experiments

To evaluate the effectiveness of the proposed DBN based i-vector framework, we conducted extensive experiments on 4 Arabic dialects (i.e., Iraqi, Levantine, MSA and Maghrebi) taken from NIST LRE2011. The training utterances for each language mainly come from two different channels in the dataset of Conversational Telephone Speech (CTS) and the narrow band Voice of America (VOA) radio broadcast dataset. The evaluation utterances are from NIST LRE 2011 with 30s, 10s, and 3s durations. There are about 400 utterances for each language and test condition as detailed in Table 4.

For comparison, we implement the following three i-vector extraction methods based on DBN:

**GMM-UBM-DBF:** a conventional i-vector approach using front-end DBF features and Gaussian posterior probabilities for back-end modeling [8].

**DNN-DBF:** an i-vector system using DBF front-end features, and senone posterior probabilities for back-end modeling [2].

**DBN-UBM-DBF:** a i-vector system using DBF front-end features, and a DBN based universal model to calculate the posterior probabilities for back-end modeling.

The dimension of the i-vector representation is fixed at 400 for all three methods and performance is evaluated in terms of the equal error rate (EER) as illustrated in [15]. To simplify the analysis, we report results without using a Gaussian score back-end.

4.1. Evaluation of Arabic dialect using Mandarin-DBN

We first evaluate the performance of the Mandarin-DBN based i-vector system on the Arabic dialect recognition task. The configuration of the Mandarin-DBN is $21 \times 43 - 2048 - 2048 - 2048 - 6004$, which is the same as in [9]. Results are presented in Table 2. For the GMM-UBM-DBF i-vector, we evaluate performance with different GMM-UBM model sizes

<table>
<thead>
<tr>
<th>Language</th>
<th>Channel source</th>
<th># of Training</th>
<th># of Test</th>
</tr>
</thead>
<tbody>
<tr>
<td>Iraqi</td>
<td>CTS</td>
<td>618</td>
<td>408</td>
</tr>
<tr>
<td>Levantine</td>
<td>CTS</td>
<td>653</td>
<td>408</td>
</tr>
<tr>
<td>Maghrebi</td>
<td>CTS</td>
<td>589</td>
<td>405</td>
</tr>
<tr>
<td>MSA</td>
<td>VOA</td>
<td>3879</td>
<td>406</td>
</tr>
</tbody>
</table>
of 512,1024, and 4096. From Table 2, we can see that the EER gradually reduces with increasing size of the GMM-UBM model as expected. The relative performance improvement of the GMM-UBM-DBF(1024) over the GMM-UBM-DBF(512) is about 16%, 8%, and 6% for 30s, 10s, and 3s test conditions respectively. However, due to the limited size of the training dataset, further increasing the GMM-UBM size may yield only slight improvements or even degrade performance.

For the DNN-DBF i-vector, as described in section 2, the model size is determined by the number of nodes in the Mandarin-DBN output layer of, which is fixed to 6004. Compared to the GMM-UBM-DBF system, DNN-DBF performs much better. This validates that the pre-trained DNN for ASR, with a well-annotated corpus, will provide better frame alignment for sufficient statistics calculation.

For the DBN-UBM-DBF i-vector, a universal model instead of the language specific DNN is used for sufficient statistics calculation. It is shown that DBN-UBM-DBF achieves the best performance. Compared to the DNN-DBF system, the relative improvement is about 46%, 7%, and 13% for 30s, 10s, and 3s test conditions respectively. This validates that a universal model derived from the pre-trained DNN may improve the i-vector representation for LID tasks.

### 4.2. Evaluation of Arabic dialect using English-DBN

We also conduct the evaluations of English-DBN based i-vector systems on the same task, to evaluate different DBN configurations, and the generalization capabilities of the proposed method. The configuration of the DBN are described in section 3, and two types of DBN are evaluated, namely 1) 21 × 48 = 2048 − 2048 − 50 − 2048 − 2048 − 1536 2) 21 × 48 = 2048 − 2048 − 50 − 2048 − 2048 − 2444 The English DBNs are trained using the publicly available Switchboard corpus, and the Kaldi speech recognition toolbox.

Experimental results are given in Table 3. For the English-DBN with 1536 nodes in the output layer, the DNN-DBF outperforms the GMM-UBM-DBF, and the DBN-UBM-DBF performs much better than the other two systems, as shown in the upper part of Table 3. The relative performance improvement is about 22% and 11% for 30s and 10s test conditions respectively. For the 3s test condition, the performance of DBN-UBM-DBF is still competitive (20.33% vs. 19.83%).

Similar conclusions can be drawn for the English-DBN with 2444 nodes in the output layer, as shown in the lower part of Table 3. However, the DNN-DBF performs worse than the GMM-UBM-DBF system. Furthermore, we can see that with the increased model size, the performance of GMM-UBM-DBF and DBN-UBM-DBF will consistently be better. There is a larger relative improvement in GMM-UBM-DBF than DBN-UBM-DBF. For DNN-DBF, degraded performance is observed, which may indicate that DNN-DBF is phonetically sensitive, and depends greatly on the components selected for sufficient statistics calculation. Therefore, it is better to derive a universal model for i-vector representation.

### 5. Conclusion

In this paper, we proposed a unified DBN based i-vector representation for LID tasks. In contrast to previous authors [1][11][9][2], this framework covers both the front-end feature extraction and back-end modeling stages. Specifically, with the DBN structure pre-trained for ASR task, the output from the middle bottleneck layer and last layer are both exploited to calculate the i-vector representation. Since there exists a transition from low-level acoustic features at the input of the DBN to high-level phonetic ones at the output, the proposed framework incorporates a graded mixture of phonetic and acoustic information for forming the enhanced i-vector. Furthermore, a universal model is derived from the DBN output to improve the transferability of the pre-trained DBN. Experimental results from the challenging Arabic dialect recognition task have demonstrated the superiority of the DBN based i-vector for LID. The generalization capability of the framework is also validated separately for DBNs trained on either an English or a Mandarin corpus.

Future work on this framework may include: 1) conducting experiments with different LID tasks, such as NIST LRE2009 and LRE2011, to further evaluate the generalization capability. 2) the high complexity of the DBN makes it difficult to be applied on real-time LID tasks. It is therefore of broad interest to find an effective model compression technique to reduce the DNN computation complexity.

### 6. Acknowledgments

The authors would like to acknowledge the support of National Natural Science Foundation of China grant no. 61172158 and Chinese Universities Scientific Fund (grants no WK2100060008 and WK2100000002 for this work.)
7. References


