I-Vector Dependent Feature Space Transformations for Adaptive Speech Recognition

Xiangang Li, Xihong Wu

Speech and Hearing Research Center,
Key Laboratory of Machine Perception (Ministry of Education),
Peking University, Beijing, 100871
{lixg, wxh}@cis.pku.edu.cn

Abstract
In this paper, we propose a new feature normalization approach for deep neural networks (DNNs) based adaptive speech recognition. Each speaker is represented by an i-vector, and the i-vector dependent block-diagonal transformation matrix is obtained by a tensor and performed on the input features. The parameters of tensor are shared by all the frames in the input window, and factorized into three matrices. The proposed approach is more practical for real-application speech recognition tasks since it eliminates the time-consuming adaptive training process to estimate the transformation matrix in feature-space discriminative linear regression (fDLR). We empirically evaluated the proposed approach on a conversational telephone speech recognition task. Experimental results show that the proposed approach can yield 7% relative improvement for the long short-term memory network based speech recognition system.

Index Terms: speech recognition, speaker adaptation, i-vector, deep neural networks, tensor

1. Introduction
For automatic speech recognition (ASR) systems, the mismatch between the model and operation condition always exists. Although, a great deal of process has been made by introducing the context-dependent deep-neural-network hidden-Markov-model (CD-DNN-HMM) based acoustic models [1][2], the ASR systems may still fail to produce the same level of performance when tested under mismatched conditions. Adaptive techniques can be used to modify the ASR systems to better match variations in environment noise, speakers, styles, and application contexts. Among all factors causing the mismatch, speaker variability is an important one.

A substantial amount of adaptive techniques have been conducted to improve the performance of earlier state-of-the-art Gaussian mixture models (GMMs) HMMs based ASR systems. For example, the well-known maximum a posterior (MAP) [3] and maximum likelihood linear regression (MLLR) [4] has been reported to achieve very good performance when adaptation data is provided. The speaker normalization techniques have also been widely used, such as the vocal tract length normalization (VTLN) [5][6] and the feature-space maximum likelihood linear regression (fMLLR) [7] wherein transformation functions are directly applied in the feature space. The key idea of these techniques is either modifying the trained model parameters towards testing condition or transforming the test features to match the given trained model, so that the mismatch between the training and testing conditions can be reduced. The recent popularity of DNNs has made the adaptation of DNNs become a very active research area. However, unlike the Gaussian means and variances, it is hard to find structure in the weights of a DNN. Rather, the DNNs have significantly more parameters due to wider and deeper hidden layers used, which casts additional challenges to adapt DNNs, especially with only a small amount of adaptation data.

Although the directly portability of adaptation techniques of GMMs to DNNs is not straightforward, various methods have been proposed in the literature for adaptation of DNNs. The first group of methods perform adaptation by applying a linear transformation to the input[8], hidden[9], or output layer[10] of the DNN. The parameters of such transformation are learned via back-propagation algorithm (BP) on adaptation data set, keeping fixed the weights of the original DNN. Meanwhile, efforts have been made to train DNNs on the adapted features. First, the widely used GMM-derived feature-space transformations can be applied to the inputs of DNNs, such as the VTLN and fMLLR, which have been evaluated in [11]. The feature-space transformations can also be derived by DNN based acoustic models. An fMLLR-like technique is proposed and called as feature-space discriminative linear regression (fDLR) [11], in which the weights of linear transformation matrix is shared by all the frames in the input window. Another group of adaptation techniques are called as speaker-aware training methods [12], in which the speaker information is provided to the network in hope that the DNN training algorithm can automatically figure out the speaker normalization. The speaker information can be derived in many different ways, such as the speaker code [13][14][15] which is jointly learned with the rest of the model parameters for each speaker, and the i-vector [16][17] which is a popular technique for speaker verification and recognition [18][19]. Besides, the speaker and speech subspaces can also be estimated and combined using tensors [20], but the total number of tensor network is very large, which makes it not practical for real-world applications.

This paper focuses on how to derive adapted features for DNNs. In fDLR approach, each speaker have a linear transformation, and for the testing speaker which is not shown in the training set, an adaptive training procedure is needed to estimate the speaker dependent transformation, which is an extra computation cost in the decoding phase. From the speaker-aware approaches, we learn that the speaker can be represented by speaker related vectors. When each speaker is represented by an i-vector, the speaker dependent transformation in fDLR approach is changed to i-vector dependent transformation. In this paper, a new adaptation approach is proposed, which uses tensors to allow the i-vector to determine the feature transformation. The proposed approach is similar to [21][22] with one
important difference: the i-vector is used to determine a linear transformation for input features in this work, while the i-vector is used to produce a linear shift for input features in [21][22]. The experiments are conducted on a spontaneous speech recognition task, and the results show significant performance improvements.

The rest of this paper is organized as follows. Section 2 describes the i-vector based techniques. Section 3 presents the proposed i-vector dependent feature-space discriminative linear regression (fDLR). Experiments and results are presented in Section 4. Finally, Section 5 concludes the paper.

2. I-vectors

In the speaker recognition and speaker verification community, the introducing of i-vectors has resulted in the state-of-the-art performance [18][19]. The i-vector represents an utterance in a low-dimensional space named total variability space. Unlike the earlier joint factor analysis (JFA), the i-vector approach has a single variability space, rather than separate speaker and channel subspaces. Given an utterance, the speaker-dependent GMM supervector is defined as follows:

\[ V_s = m + Ti_s \] (1)

Where \( m \) is the supervector of the universal background model (UBM) means, \( T \) is the total variability matrix subsuming principle components of variability in the supervector space, and \( i_s \) is a random vector with a standard normal distribution \( N(0, 1) \). The vector \( i_s \) contains the total factors and is referred to as the i-vector.

3. I-vector dependent feature-space transformation

This section describes the proposed i-vector dependent feature-space transformation, or i-vector dependent feature-space discriminative linear regression (fDLR) for DNN based adaptive speech recognition. This proposed approach comes from the fDLR approach, but represents each speaker with an i-vector. In fDLR approach, the feature-space transformation is speaker dependent, while in fDLR approach, the transformation is speaker’s i-vector dependent.

Pointed out by [17], the i-vector encodes precisely those effects to which we want our system to be invariant: speaker, channel and background noise. Here, we use tensors to allow the i-vector to determine the transformation performed on the input features, which should enable it to normalize the features.

3.1. Feature-space transformation

In fMLLR or fDLR approach, the linear transformation is applied to the input features to bring them closer to the speaker independent models, which can be formulated as follow:

\[ a_t = M(s)a_t + b(s) = W(s)\xi_t \] (2)

where \( a_t \) is the original feature vector from speaker \( s \), the matrix \( M(s) \) and bias \( b(s) \) are speaker dependent. \( \xi_t = [1, a_t]^T \) is the extended observation vector, and \( W(s) = [b(s)^T, M(s)^T]^T \) is the extended transformation matrix. However, it is necessary to point out that, for the inputs of DNNs which always contains neighbor augmentations, the matrix \( W(s) \) is block-diagonal, with blocks and bias tied across neighbor frames. Each speaker has its own transformation. In the decoding phase, for the testing speaker which is not shown in the training set, an additional adaptive training procedure is required to estimate the transformation firstly using the adaptation data or the unsupervised recognition transcripts.

However, when estimating the matrix using BP algorithm, it requires doing propagation from the bottom layer to the top layer, and then back-propagation from the top layer to the bottom layer in the DNN. The computational cost of training the transformation matrix for each speaker could be very expensive. On the other hand, in some difficult speech recognition tasks, the unsupervised recognition transcriptions, which are often obtained by decoding using speaker-independent models, may have too many recognition errors making it unsuitable for the unsupervised adaptive training.

From the speaker-aware approaches, we learn that the individual speakers can be represented with speaker related vectors, for example with the i-vectors. When each individual speaker is represented by an i-vector, estimating the speaker dependent transformation matrix for such speaker becomes computing the transformation matrix with the corresponding i-vector. In other words, we need a model which can produce a matrix for each given vector. This strongly suggests the multiplicative interaction between the raw input features and i-vectors. To achieve this goal, we modify the fDLR so that the transformation matrix is a “learned” function of the i-vector:

\[ a_t = W(i_s)\xi_t \] (3)

Equation 3 is similar to Equation 2, except that \( W(s) \) is replaced with \( W(i_s) \), allowing each i-vector \( i_s \) to specify a different transformation matrix. In other words, when \( i_s \) is given, then \( a_t \) is a linear function of \( \xi_t \).

It is natural to define \( W(i_s) \) using a tensor. If we store \( M \) matrices \( W^{(1)}, W^{(2)}, \ldots, W^{(M)} \), where \( M \) is the number of dimensions of i-vectors \( i_s \), we could define \( W(i_s) \) as:

\[ W(i_s) = \sum_{m=1}^{M} \overline{i}_s^{(m)} W^{(m)} \] (4)

Where \( \overline{i}_s^{(m)} \) is the \( m \)-th coordinate of \( i_s \). The introducing of tensors makes the transformation matrix controlled by i-vectors. However, if we use a 1-of-\( M \) encoding of a speaker (in total \( M \) speakers), it is easily seen that, every speaker has a unique associated weight matrix, and the approach goes back to fDLR. The i-vector can be viewed as the continuous representation of each speaker, resulting in a more compact model for feature normalization.

In the sense of representing speakers, the proposed fDLR approach replaces the 1-of-\( M \) encoding in fDLR approach with continuous representations, specifically the i-vectors. The 1-of-\( M \) encoding in fDLR does not embody the relationships or structures among speakers, making it inefficient for the unseen testing speakers. On the other hand, by introducing i-vectors, the fDLR makes use of the relationships embedded in i-vectors to produce the transformation for unseen testing speakers, rather than an additional adaptive training procedure.

In summary, in the proposed fDLR approach, the tensors are used to allow i-vectors to determine the transformation matrix performed on each frame to get the normalized features.

3.2. Tensor and factorization

A direct way to introduce tensor is as in [20], but the total number of tensor will be very large, which makes it not practical
for real-world applications. Another way suggested by [23] is to factor the tensor into inner product of three matrices which have much fewer parameters. The latter approach is adopted in this paper. The tensor $W_{(i,s)}$ is factored as the “three way inner product” of $W_o, W_f, W_f$

$$W_{(i,s)} = W_o \cdot \text{diag}(W_{i,s}) \cdot W_f \quad (5)$$

The dimensionality of the vector $W_{i,s}$ is the number of the factors, denoted by $F$.

Indicated by [24], when training the tensor, the product of parameters in Equation 5 makes gradient descent learning difficult. If, for example, $W_o$ is very small and $W_f$ is very large, we get a very large derivative for the small matrix $W_o$ and a very small derivative for the large matrix $W_f$. In [24], the 2nd-order approach is used to handle this problem.

However, in this paper, we attempt to stabilize the training by limiting the ranges of derivatives. Thus, a saturating non-linearity function (sigmoid) is applied on the output of $W_{i,s}$, considering the fact that we must ensure $o_i$ is still a linear function of $x_i$ when $i, s$ is given. By this means, if $W_i$ is very large, the derivative for $W_o$ would not be too large. In our experiments, the convergence of this modified three-way transformation is as fast as the other parts of DNNs, even if no 2nd-order approach is used. Then $W_{(i,s)}$ is given as:

$$W_{(i,s)} = W_o \cdot \sigma(\text{diag}(W_{i,s})) \cdot W_f \quad (6)$$

where, $\sigma(\cdot)$ is sigmoid function. As shown in Figure 6, the modified factorization is more like applying sigmoid gate operation on the factors.

In [20], the tensors have been used for speaker adaptation, where the speaker factors are estimated by i-vector, and a tensor is applied to the output layer. In the proposed iFDLR, the speaker information is directly represented by i-vector, and a tensor is applied to inputs and shared by all the frames in the input window. The total number of parameters in iFDLR is small compared to that in [20], making it more practical for real-world applications.

From the perspective of applying i-vector dependent speaker normalization, the i-vector dependent linear feature shift has been exploited for GMMs and DNNs based adaptation [25][21]. The main difference between these approaches and this work is that we adopt linear transformation instead of linear shift.

It deserves to note that a singular value decomposition (SVD) factorization based speaker adaptation approach was proposed in [26], in which SVD is applied on the weight matrices in trained DNNs, and only the singular values are tuned with the adaptation data. Recall that the weights $W$ can be factorized into three components using SVD:

$$W_{m \times n} = U_{m \times n} \Sigma_{n \times n} V^T_{n \times n} \quad (7)$$

where $\Sigma_{n \times n}$ is a diagonal matrix that contains all the singular values. Compared Equation 7 with Equation 6, it is easily seen that, the speaker dependent adapted singular values is replaced with i-vector dependent $\sigma(\text{diag}(W_{i,s}))$.

4. Experiments

Experiments are conducted on a large vocabulary speech recognition task - the HKUST Mandarin Chinese conversational telephone speech recognition [27]. The corpus (LDC2005S15, LDC2005T32) is collected and transcribed by Hong Kong University of Science and Technology (HKUST), which contains 150-hour speech, and 873 calls in the training set (1737 speakers) and 24 calls in the development set, respectively. In our experiments, around 3-hour speech was randomly selected from the training set, used as the validate set for network training. The original development set in the corpus was used as ASR test set, which is not used in the training or the hyper-parameters determination processes.

4.1. Experimental setup

The speech in the corpus is represented with 25ms frames of Mel-scale log-filterbank coefficients (including the energy value), along with their first- and second-order temporal derivatives. The feed-forward DNNs use concatenated features constructed by concatenating the current frame with 5 frames in its left and right contexts. The inputs to recurrent neural networks (RNNs) are only the current frames (no window of frames).

A trigram language model estimated using all the acoustic model training transcriptions is used in all the experiments. The hybrid approach [1][28] is used, in which the neural networks’ outputs are converted as pseudo likelihood as the state output probability in hidden Markov model (HMM) framework. All the networks are trained based on the alignments generated by a well-trained GMM-HMM systems with 5529 senones (realignment by DNNs are not performed), and only the cross-entropy objective function is adopted.

In our experiments, two kinds of networks are adopted as acoustic models: one the rectifier linear units (ReLU) based feed-forward neural networks [29][30], and other is the long short-
term memory (LSTM) based RNNs [31][32]. We implemented the network training on multi-GPU devices with asynchronous stochastic gradient descent algorithm (ASGD) [33][34]. When training the LSTM networks, the approaches described in [32] are adopted. In details, the truncated back-propagation through time (BPTT) learning algorithm [35], and clipping strategy [36] are adopted in the training. In the experiments, the learning rate for training each network is decreased exponentially, and the initial and final learning rates are set specific to each network for stable convergence of training.

4.2. Experimental results

In the proposed IfDLR approach, the number of factors $F$ is an important hyper-parameter. We firstly explore the number of factors with ReLU DNNs based acoustic models. Table 1 shows the performance as a function of the number of factors. These ReLU DNNs in Table 1 have 4 hidden layers, and each layer had 2000 ReLU units. Aside from the baseline ReLU DNN without any adaptation methods, we implement the IfDLR approach for comparison. Note that in the experiments, we used 5529 senones against 3302 senones in [32][37], which led to slightly better baseline experimental results.

It is expected that using adaptation methods significantly improve the performance. In the experiment about the IfDLR approach, the unsupervised recognition transcripts generated by the first-pass recognition are used for training the transformation matrix. On the contrary, in the proposed IfDLR approach, we do not need second-pass recognition or updating transformation matrix, thus the problems caused by the errors in the unsupervised recognition transcripts are avoided. The proposed IfDLR performs much better than the IdLR approach. When comparing the CERs with different number of factors, we were able to obtain the best performance for IfDLR approach using 256 factors.

The approach proposed in [16] where the input features are augmented with speaker level i-vectors also drew our attention. However, we tried to implement this approach on our setups but failed to get gains out of it. We noticed that the authors of [21] also tried to implement it but failed as well. We will continue to investigate the training techniques for this speaker level i-vector-augmented approach in our future work.

<table>
<thead>
<tr>
<th>DNNs</th>
<th>Adaptation Methods</th>
<th>#Factor</th>
<th>CER(%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ReLU</td>
<td>-</td>
<td>-</td>
<td>37.73</td>
</tr>
<tr>
<td>ReLU</td>
<td>IdLR</td>
<td>-</td>
<td>36.84</td>
</tr>
<tr>
<td>ReLU</td>
<td>IfDLR</td>
<td>128</td>
<td>36.16</td>
</tr>
<tr>
<td>ReLU</td>
<td>IfDLR</td>
<td>256</td>
<td><strong>36.02</strong></td>
</tr>
<tr>
<td>ReLU</td>
<td>IfDLR</td>
<td>384</td>
<td>36.29</td>
</tr>
</tbody>
</table>

Next, we discuss the effects of having different i-vector dimensions for IfDLR. It can be seen in Table 2 that, dimension = 100 is the optimal choice for IfDLR approach. We were able to obtain 4.7% relative performance improvement against the baseline system by using 100-dimensional i-vectors and 256 factors.

In many ASR tasks, LSTM RNNs have been shown to give state-of-the-art performance [38][32][39][40]. Thus, we explore applying the IfDLR for LSTM RNNs based ASR systems. In these experiments, two type of RNN architectures are used: Table 2: Speech recognition results of having different i-vector dimensions for IfDLR.

<table>
<thead>
<tr>
<th>DNNs</th>
<th>Adaptation Methods</th>
<th>I-vector Dimensions</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>50</td>
</tr>
<tr>
<td></td>
<td>ReLU</td>
<td>IfDLR</td>
</tr>
<tr>
<td></td>
<td></td>
<td>100</td>
</tr>
<tr>
<td></td>
<td></td>
<td>150</td>
</tr>
</tbody>
</table>

one is the one-layer LSTM [38] based RNNs (denoted as “LSTM”) in Table 3 and other is the networks that have three ReLU hidden layers on top of LSTM layer [32] (denoted as “LSTM+3×ReLU” in Table 3). All the LSTM layers have 2000 LSTM cells, and these cells are projected to 750 nodes. In “LSTM+3×ReLU” network, each ReLU layer has 2000 nodes. When we adopt the IdLR for LSTM RNNs, since only one frame used as the network input, there is no weight sharing.

Results are shown in Table 3. Significant performance improvements are also observed by using the proposed IfDLR approach for LSTM RNNs based ASR systems. More specifically, for the “LSTM+3×ReLU” network, with the IfDLR approach, we were able to achieve the CER of 31.58%, which is a 6.7% relative CER reduction.

5. Discussion and conclusions

In this paper, we present an effective approach, called as i-vector dependent feature space discriminative linear regression (IfDLR), to derive adapted features for DNNs based acoustic models. In the IfDLR approach [11], transformation matrix performed on the input features is speaker dependent. While in the proposed approach, each individual speaker is represented by an i-vector, and an i-vector dependent transformation matrix is obtained by a tensor. In another words, by introducing i-vectors, the proposed approach makes use of the relationships embedded in i-vectors to produce the transformation matrix for unseen testing speakers, rather than an additional adaptive training procedure. Furthermore, in order to accelerate the convergence of the tensor training, the tensor is factorized into three matrices and the sigmoid function is added.

We empirically evaluated this IfDLR approach on a large vocabulary Mandarin Chinese conversational telephone speech recognition task. The experiments show that IfDLR approach can yield significant performance improvements. Specifically, when the acoustic model adopts LSTM RNN, which is the state-of-the-art technique for acoustic modeling, the IfDLR approach can still bring around 7% relative improvement.

However, we believe this work is just a preliminary study on i-vector dependent feature normalization for DNNs based adaptive speech recognition. As future work, we will explore speaker adaptive training framework based on the IfDLR, and conduct experiments for the comparisons of proposed IfDLR with other relative approaches, such as speaker-code, and i-vector based feature shift.
6. References


