Speaker Adaptation of Convolutional Neural Network using Speaker Specific Subspace Vectors of SGMM

Murali Karthick B, Prateek Kolhar, S Umesh
Department of Electrical Engineering, Indian Institute of Technology, Madras
eel3s010@ee.iitm.ac.in, prateek.kolhar@gmail.com, umeshs@ee.iitm.ac.in

Abstract

The recent success of convolutional neural network (CNN) in speech recognition is due to its ability to capture translational variance in spectral features while performing discrimination. The CNN architecture requires correlated features as input and thus fMLLR transform which is estimated in de-correlated feature space fails to give significant improvement. In this paper, we propose two methods for extracting speaker adapted features in a correlated space using SGMMs. First, we estimate fMLLR transforms for correlated features by full covariance Gaussians using SGMM approach. Second, we augment speaker specific subspace vectors with acoustic features to provide speaker information in CNN models. Finally we propose a bottleneck - joint CNN/DNN framework to exploit the effects of both (fMLLR+ivectors) and (SGMM-fMLLR+speaker vectors) features. Experiments on TIMIT task show that our proposed features give 5.7% relative improvement over the log-mel features. Furthermore experiments on switchboard task show that the bottleneck - joint CNN/DNN model achieves 12.2% relative improvement over baseline joint CNN/DNN framework.

Index Terms: speech recognition, DNN, CNN, fMLLR, SGMM, subspace vectors

1. Introduction

Automatic speech recognition (ASR) systems have been through various changes in the way speech is modeled. The recent advent of deep neural networks (DNN) outperformed the conventional Gaussian mixture model - hidden Markov model (GMM-HMM) system [1]. Speaker Adaptation of features using feature space maximum likelihood regression (fMLLR) [2] and speaker identity vectors (i-vectors) [3] in DNN give significant improvements in recognition performance. The ability of DNN to model de-correlated features with i-vectors helps it to perform better speaker normalization. Further improvements in neural network were obtained by using convolutional neural network (CNN) [4] as it models the correlation along time and frequency in local segments of the input features.

The CNN architecture [5] can normalize speaker variance and stay immune to channel distortions, speaking styles, etc. The CNN consists of one or more pairs of convolution and max-pooling layers and the final layer as a softmax layer that produces the posterior probabilities. Each hidden activation is computed by multiplying localized inputs against the weights. These weights are shared across the entire input space, hence act like a filter. The filters help the CNN to model the correlation among the neighboring points in the input. The shared weights capture the translational variance to remove variability in the hidden units. Since CNNs try to model correlations they require input features such as log-mel filter bank (log-mel) which are locally correlated in time and frequency.

The improvement methods for the CNN model investigated in [6], show that vocal tract length normalization (VTLN) [7] over log-mel features improves the performance of CNN while fMLLR transform doesn’t provide much improvement. One such method is multiscale CNN/DNN approach which showed better recognition performance. In this method, convolutional layers and fully connected layers are trained with VTLN warped log mel features with their outputs fed to a final DNN. Although, these methods showed improvement with VTLN warped features, fMLLR transformed log-mel features failed to improve due to the de-correlation effect of diagonal covariance Gaussian model used during estimation. Further study in [6], show that CNN model trained with features containing both fMLLR and i-vectors do not show as much improvement as seen in DNN. It also shows that the i-vectors do not satisfy the locality property and are more challenging to integrate with CNN.

To alleviate the above issues of speaker adaptation in CNN models, we propose three effective techniques: First, we modify the diagonal covariance transform to full-covariance fMLLR transform - using SGMM to preserve the correlation in the log-mel features. Second, in place of the i-vectors, we propose to use speaker specific subspace vectors of SGMM to train CNN model. Finally, we introduce a new framework in which a bottleneck layer placed after CNN layers and fully connected layers helps to extract features of relatively lesser dimension. These extracted features from both layers are then combined to train a final DNN model. The motivation behind using the bottleneck features is that it is more effective compared to directly providing the hidden layer outputs as features as suggested in [8].

To substantiate our hypothesis we conducted experiments on TIMIT and large vocabulary task using 30 hours switchboard corpus. The evaluation results showed that our proposed modifications in input features and model framework gave significant improvement to CNN. Our paper is structured as follows: In section 2, we have discussed about the prior works in modified CNN framework and speaker adaptation of CNN. Section 3 gives the overview of subspace Gaussian mixture model and how fMLLR transforms, speaker vectors are estimated. In section 4 we describe the architecture of bottleneck - joint CNN/DNN model. Section 5 gives the experimental setup and the parameters used in our system. In section 6 investigation is done over the proposed adaptation approaches on input features provided to CNN followed by its performance in our proposed framework. In section 7 the results are tabulated and discussed for LVSCR tasks and finally in 8 we conclude by showcasing the importance of our proposed modifications.
2. Related Work

The past work on applying speaker adaptation techniques to CNN has been proposed by [6]. The authors in [6] suggested a simple technique to apply fMLLR to CNN based input features. In this method, input features are re-correlated by multiplying the inverse of semi-tied covariance matrix with fMLLR transform matrix for training CNN model. However, these methods give relatively very less improvements showing that re-correlating the log mel features is not beneficial. In contrast to this approach, we have shown a simple way to compute fMLLR transforms for full-covariance matrix using SGMM.

The DNN and CNN need to be fed with separate type of features namely fMLLR and log-mel features for better modeling. This problem has been discussed and handled in [9] by using a joint CNN/DNN scheme, where joint training of CNN and DNN is done with different feature sets. In this approach CNN and DNN are trained separately with VTLN+log-mel features and fMLLR features respectively. The output of these two models are then fed as input to a final DNN. Even though this method achieved better results, the number of parameters to be estimated are huge. This paper tries to overcome this problem by placing a bottleneck in both DNN and CNN before feeding it into final DNN.

3. Subspace Gaussian Mixture Model (SGMM)

SGMM [10] is an acoustic modeling technique whose mean and mixture weights are estimated by projecting a globally shared parameter towards a context-dependent state using low dimensional subspace vector. The important property of SGMM is that it is relatively compact and thus each specific state can be modeled with full-covariance Gaussian mixtures with lesser number of parameters. In SGMM, speaker adaptation is incorporated in the form of fMLLR transform approach and speaker specific subspace vectors.

3.1. SGMM-fMLLR Transforms

In [10] it is shown that SGMM is an effective model to estimate full-covariance fMLLR transforms which is an extension of the work done in [11]. A transform matrix \( W \) of dimension \( d \times d + 1 \) is estimated, together with diagonal matrix \( D \) corresponding to eigenvalues in LDA computation. The fMLLR transformed feature vector \( \hat{x} \) is computed as:

\[
\hat{x} = Wx = \left[ A(s); b(s) \right], \quad A \text{ is square transform matrix and } b \text{ represents bias.}
\]

3.2. Speaker Specific Subspace Vectors

The speaker vectors of SGMM [10] are estimated for each speaker separately after fixing the model parameters. These vectors are similar to the low-dimensional vector in joint-factor analysis but are well behaved and can be easily estimated with lesser number of parameters. The speaker vectors perform speaker adaptation in SGMM by moving the mean parameter closer to speaker space. The speaker subspace vectors \( \gamma(s) \) are estimated using:

\[
H(s) = \sum_{i} \gamma_i(s) H_{spk}^{i}, \quad H_{spk}^{i} = N_{i}^{T} \Sigma_{i}^{-1} N_{i}
\]

Here \( \gamma_i(s) \) denotes the posterior probability of speaker \( s \), \( N_i \) denotes speaker projection subspace matrix, \( y(s) \) is an acceleration. This problem has been discussed and handled in [9] by using a joint CNN/DNN scheme, where joint training of CNN and DNN is done with different feature sets. In this approach CNN and DNN are trained separately with VTLN+log-mel features and fMLLR features respectively. The output of these two models are then fed as input to a final DNN. Even though this method achieved better results, the number of parameters to be estimated are huge. This paper tries to overcome this problem by placing a bottleneck in both DNN and CNN before feeding it into final DNN.

4. Bottleneck - Joint CNN/DNN Framework

Bottleneck - Joint CNN/DNN approach is an extension of the existing joint CNN/DNN framework implemented by [9]. In this framework the CNN and DNN models are trained separately with their appropriate features and then their outputs are concatenated to train a final DNN model. This method exploits the benefits of CNN specific features and DNN specific features together. We propose to improve on this framework to reduce the input dimension by adding bottleneck layers to the CNN and DNN as in figure 1. This modification reduces the number of parameters and also provides significant improvement in performance.
transform (MLLT) [17] and fMLLR transformation [2] are then
applied over these MFCC features to build a GMM-HMM sys-
tem. 40-dimensional log-mel filter bank features with their delta
and acceleration coefficients are extracted to build an SGMM.
A full covariance universal background model (UBM) [18] is
then transformed to different context dependent tied-states of
SGMM using subspace vectors.

Table 1: Number of tied-states, Gaussian mixtures, network lay-
ers and nodes, substates of SGMM used for TIMIT and 30 hours
of SWBD corpus

<table>
<thead>
<tr>
<th>Datasets</th>
<th>GMM-HMM</th>
<th>SGMM</th>
<th>DNN</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ts</td>
<td>Gs</td>
<td>ts/ss</td>
</tr>
<tr>
<td>TIMIT</td>
<td>1956</td>
<td>15021</td>
<td>5827/9649</td>
</tr>
<tr>
<td>SWBD</td>
<td>4167</td>
<td>90108</td>
<td>3697/400006</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>hl</th>
<th>ns</th>
<th>ts/ss</th>
</tr>
</thead>
<tbody>
<tr>
<td>TIMIT</td>
<td>57</td>
<td>6</td>
<td>1024</td>
</tr>
<tr>
<td>SWBD</td>
<td>6549</td>
<td>6</td>
<td>1024</td>
</tr>
</tbody>
</table>

5.0.1. DNN and CNN Training

KALDI [12] recipe for TIMIT and SWBD is used to perform
the DNN and CNN experiments. The fMLLR features spliced
over a context of 11 frames are fed as input to build a genera-
tively pre-trained DNN system. The number of output targets
in DNN is kept same as the number of context-dependent tied-
states of GMM-HMM system. We use 5 hidden layers each con-
taining 1024 units with softmax activation function and output
layer with softmax function. The parameter details of GMM-
HMM, SGMM and DNN are shown in table 1.

In [4], it is shown that CNN requires correlated features
such as log-mel. Thus 40-dimensional log-mel filter bank fea-
tures with their delta and acceleration coefficients are used in
all our CNN experiments. The log-mel features are spliced over
a temporal context width of 11 frames. We use two CNN layers
in our experiments, each with a separate configuration [6]. An
overlapping context window of 9 × 9 is applied over the input
features in the first layer. 3 outputs from the first convolutional
layer are then pooled to the max-pooling layer. In the second
CNN layer an overlapping window of 4 × 3 is applied to obtain
8 × 1 windowed input for convolutional layer. These two CNN
layers are trained using frame level cross-entropy training. The
method to incorporate speaker vectors into the CNN model is
described in section 6.2. A common CNN configuration shown
in table 2 is used for both TIMIT and SWBD experiments.

Table 2: CNN configuration for both TIMIT and SWBD exper-
iments. Here Win denotes Windows and Pool denotes pooling.

<table>
<thead>
<tr>
<th>CNN Layer</th>
<th>Feature Space</th>
<th>Temporal Space</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Size</td>
<td>Shift</td>
</tr>
<tr>
<td>Win</td>
<td>Pool</td>
<td>Win</td>
</tr>
<tr>
<td>1</td>
<td>9</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>1</td>
</tr>
</tbody>
</table>

The number of nodes in each convolutional layer is 512.
The input dimensions of first layer is 3 times 9 × 9 window
which is 243 dimensions (243 × 512). The second layer contains
512 × 4 × 3 = 6144 (6144 × 512) dimensions.

6. Analysis

In this section we analyze the effect of our three proposed meth-
ods in the CNN framework. First, we apply SGMM based
fMLLR transform to apply speaker adaptation while preserv-
ing the correlation in the log-mel features. Secondly, we aug-
ment the speaker specific subspace vectors of SGMM with the
speaker normalized input features to provide additional speaker
information. Further we compare the performance of our bot-
tleneck - joint CNN/DNN framework with the existing joint -
CNN/DNN framework. The experiments to analyze the effect of
the proposed speaker adapted features is done using TIMIT
dataset. The model contains two CNN layers and an output
layer in these CNN experiments.

6.1. Effect of SGMM-fMLLR Features

The SGMM-fMLLR transform matrix (40 × 41) is estimated
from SGMM and then applied over the log-mel features to ob-
tain SGMM-fMLLR (40 × 1) features. The intuition behind
applying fMLLR transforms estimated using SGMM over log-
mel features are: 1) it can generate full-covariance transform
matrix with lesser number of parameters as mentioned in [11],
2) it helps to transform the log-mel features by preserving the
correlations and locality. These features of SGMM-fMLLR
transform provide better feature representation that help CNN
to perform better speaker normalization.

To validate our hypothesis, we can observe from figure 2
the frame accuracy improves for our proposed features. The
frame accuracy denotes the classification power of CNN and
varies for different features. From the figure, it is noted that
the accuracy for fMLLR and SGMM-fMLLR features has less
variation. Although, the fMLLR features has gained better ac-
curacy over log-mel features the recognition rate of fMLLR fea-
tures is inferior compared to log-mel features. The similar pat-
tern is reflected in table 3 where conventional fMLLR features
degrade the CNN performance while SGMM-fMLLR features
gain relative improvements. So further experiments on CNN
with speaker vectors are done only using SGMM-fMLLR fea-
tures and log-mel features.

Table 3: Performance Comparison of SGMM-fMLLR features
with basic fMLLR features. Here R.I represents relative improve-
ment

<table>
<thead>
<tr>
<th>Feature Type</th>
<th>% PER</th>
<th>% R.I.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Log-mel</td>
<td>21.2</td>
<td>-</td>
</tr>
<tr>
<td>fMLLR</td>
<td>22.3</td>
<td>-4.9</td>
</tr>
<tr>
<td>SGMM-fMLLR</td>
<td>21.0</td>
<td>0.9</td>
</tr>
</tbody>
</table>
6.2. Effect of Speaker Specific Subspace Vectors

The SGMM speaker vectors are a well structured representation of the speaker’s phonetic information in lower dimensions. Since these vectors act as mean adapting bias towards target speakers in SGMM, we hypothesize that they also can help CNN perform speaker adaptation more effectively. In this method we extract subspace vector of \( S = 40 \) subspace dimensions for each speaker. In CNN, the feature augmentation is done as shown in [4]. A 40 dimensional speaker vector is concatenated in the direction of frequency when a \( 9 \times 9 \) overlapping context window is applied over the input feature matrix resulting in a new window size of \( 9 \times 40 \) similar to the method used for i-vectors in [6]. The same speaker vectors are augmented for all context window shifts and the weights are shared across the shifts.

Here we compare the performance of speaker vectors and i-vectors as augmented features with log-mel and SGMM-fMLLR features. From the table 4 we see that the speaker vectors along with log-mel features performed better than i-vectors. The table also shows that using speaker vectors along with SGMM-fMLLR features further improves the performance. The speaker vectors are more efficient in training CNN than i-vectors. The important reason behind this is that, the i-vectors is of higher dimensions and hence needs huge number of parameters to be estimated. In case of speaker vectors, the low dimensionality helps in helping robust estimation of weights in CNN with lesser parameters for performing better speaker adaptation.

<table>
<thead>
<tr>
<th>Feature Type</th>
<th>% PER</th>
<th>% R.I.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Log-mel + i-vectors</td>
<td>20.9</td>
<td>1.4</td>
</tr>
<tr>
<td>SGMM-fMLLR + i-vectors</td>
<td>20.7</td>
<td>2.3</td>
</tr>
<tr>
<td>Log-mel + spk. vectors</td>
<td>20.4</td>
<td>2.8</td>
</tr>
<tr>
<td>SGMM-fMLLR + spk. vectors</td>
<td>19.8</td>
<td>5.7</td>
</tr>
</tbody>
</table>

6.3. Effect of Bottleneck-Joint CNN/DNN Framework

In our experiments we have used the joint CNN/DNN [9] as a baseline model. Here we have used the (SGMM-fMLLR + speaker vectors) as CNN specific features and (fMLLR + i-vectors) as DNN specific features. Table 5 compares the recognition performance of our proposed joint CNN/DNN model with bottleneck framework with the existing joint CNN/DNN technique for TIMIT dataset. The bottleneck layer helps to perform better classification over feature frames and thus the model show improved performance compared to joint CNN/DNN without having bottleneck layer. The table 5 also shows that the number of parameters involved while training our proposed model reduces by approximately 14 % when compared to the model without bottleneck layer. Also from the table we can observe that bottleneck - joint CNN/DNN gave 2.5 % absolute improvement over the baseline model.

7. Results on LVCSR Task

In this section we compare the results of joint CNN/DNN model with bottleneck-joint CNN/DNN using our proposed features using 30 hours of switchboard task. The results show that our proposed model relatively works well with SGMM-fMLLR+speaker vector features and give an average 10.7 % relative improvement over baseline joint CNN/DNN model. The results in 6 shows that as the data size increases the performance of our proposed framework gave 12% relative improvement while for TIMIT task it gave 8.3 % relative improvement over the baseline system. In case of our proposed speaker adaptation methods, the relative improvement is more for SGMM-fMLLR with switchboard while for speaker vectors the same range of improvement is obtained as in TIMIT dataset.

### Table 5: Performance comparison of our proposed Bottleneck-Joint CNN/DNN with Joint CNN/DNN on TIMIT dataset. Here Bnk denotes bottleneck and # Prms represents number of parameters.

<table>
<thead>
<tr>
<th>Feature Type</th>
<th>Joint CNN/DNN</th>
</tr>
</thead>
<tbody>
<tr>
<td>With/Without Bnk</td>
<td>% WER (# Prms)</td>
</tr>
<tr>
<td>fMLLR + i-vectors</td>
<td>20.3 (13.4)</td>
</tr>
<tr>
<td>With Bnk</td>
<td>19.5 (21.7)</td>
</tr>
<tr>
<td>% R.I.</td>
<td>6.9</td>
</tr>
</tbody>
</table>

### Table 6: Performance comparison of our proposed Bottleneck-Joint CNN/DNN with Joint CNN/DNN for SWBD corpus.

<table>
<thead>
<tr>
<th>Feature Type</th>
<th>Joint CNN/DNN</th>
</tr>
</thead>
<tbody>
<tr>
<td>With/Without Bnk</td>
<td>% WER (# Prms)</td>
</tr>
<tr>
<td>fMLLR + i-vectors</td>
<td>24.9</td>
</tr>
<tr>
<td>With Bnk</td>
<td>23.4</td>
</tr>
<tr>
<td>% Rel. Imp</td>
<td>11.5</td>
</tr>
</tbody>
</table>

8. Conclusion

In this paper we have proposed two simple yet effective techniques for incorporating speaker adaptive features in CNN domain. In the CNN architecture fMLLR transform approach using SGMM and the speaker specific subspace vectors are observed as better alternatives to conventional fMLLR and i-vectors. The addition of bottleneck layer in joint CNN/DNN model gives an improvement in all our experiments and also helps in reducing the number of parameters. The experimental results show that our proposed modification to the features give 5.7 % and 13.1 % relative improvement over the log-mel filterbank features for TIMIT and switchboard corpus respectively. A 12.2 % relative improvement is obtained for joint CNN/DNN with bottleneck over basic joint CNN/DNN model in switchboard task. The above improvements in the CNN framework are observed as a support for our hypothesis that SGMM-fMLLR and speaker vectors play a important role as input features for CNN.

9. References


