LATTICE BASED OPTIMIZATION OF BOTTLENECK FEATURE EXTRACTOR WITH LINEAR TRANSFORMATION

Diyuan Liu[1]*,Si Wei*[2]*,Wu Guo*[1]*,Yebo Bao*[1]*, Shifu Xiong*[1]*,Lirong Dai*[1]

[1]National Engineering Laboratory for Speech and Language Information Processing
University of Science and Technology of China, Hefei, Anhui, P. R. China
[2]iFlytek Research, Anhui USTC iFlytek Co., Ltd., Hefei, Anhui, P. R. China
Email: {ldy2012,bybillow,domine}@mail.ustc.edu.cn,
siwei@iflytek.com,{guowu,lrdai}@ustc.edu.cn

ABSTRACT

This paper proposes a lattice-based sequential discriminative training method to extract more discriminative bottleneck features. In our method, the bottleneck neural network is first trained with cross entropy criteria, and then only the weights of bottleneck layer are retrained with sequential criteria. If the outputs of the layer before bottleneck are treated as the raw features, the new method is an equivalent to a linear feature transformation algorithm. This linearity makes the optimization much easier than updating the whole neural network. Just like the fMPE and RDLT, the neural network is retrained with batch mode gradient descent, making the training to be easily implemented in parallel. Meanwhile, batch mode optimization can naturally deal with the indirect gradient to make the optimization more precise. Experimental results on a Mandarin transcription task and the Switchboard task have shown the effectiveness of the proposed method with the CER decreases from 12.2% to 11.3% and the WER from 16.1% to 15.0%,respectively.

Index Terms— speech recognition, bottleneck features, neural networks, discriminative training, sequence training

1. INTRODUCTION

In the past 30 years, the hidden Markov models (HMMs) and Gaussian mixture models(GMMs) play a major role in the automatic speech recognition(ASR). Traditionally, GMMs are used to model the output distributions of tied-states, and the parameters of GMMs are trained with the EM algorithm. During the past decade, discriminative training techniques have brought significant improvement to the estimation of GMM-HMMs. Discriminative training can be applied on model space such as maximum mutual information(MMI)[1], minimum classification error(MCE)[2, 3],and minimum phone error(MPE)[4]. Alternatively, discriminative training can also be applied on feature space such as fMPE[5], fbMMI[6], NN-fMMI [7] and RDLT[8]. In fMPE, the acoustic features are projected into high dimensional posterior vectors by a trained GMM, and then be projected into a lower dimensional space with a linear transformation matrix in order to correct some predefined features.

Recently, with the successful application of deep neural network (DNN) on speech recognition [9, 10], the performance of ASR is greatly improved. DNN parameters (weights and biases) are obtained through two steps: the layer-wise pre-training[11, 12, 13] and the back propagation based on a cross entropy criterion. Similar to discriminative training techniques in GMM-HMMs, sequence-discriminative training [14, 15, 16] for neural networks has been proposed. After sequence-discriminative training, the recognition accuracy can be further improved.

Another important application of neural networks is discriminative feature extraction through bottleneck(BN) features [17, 18, 19, 20]. Bottleneck features are usually extracted as follows: At first, a multi-layer neural network with a narrow middle layer is trained with cross entropy criterion. Then in feature extraction procedure, the outputs of the BN layer are treated as the acoustic features. After the BN features are extracted, traditional GMM/HMMs are built with the conventional training paradigm.

In [21], a more targeted manner is proposed to train bottleneck features with sequential criterion. Lattice based MMI criterion is utilized to refine the BN feature extractor. The objective function is calculated with the GMM/HMM models trained from the BN feature. In that paper, the parameters of the GMM/HMM are kept unchanging during the training process of BN network. After training, new BN features are extracted and then the GMM/HMM acoustic models are re-estimated. It's easy to see that directly optimizing the whole neural network is a complex non-convex problem. So the author in [21] adopt stochastic gradient descent (SGD) to train the network. SGD algorithm is hard to parallel while Hessian-free [16] can solve this problem to some extent. However, both SGD and Hessian-free are difficult to consider the indirect gradient.

Considering all of these problems, this paper proposes a new method to train the BN feature extractor with GMM/HMM sequential criterion. First, we suggest to update only the last layer's weights, which makes the optimization like a generalized linear feature transformation problem. This assumption makes the optimization much simpler. Then, just like the fMPE or the RDLT method, we optimize the linear transformation with batch mode gradient descent. In batch mode, we can calculate the direct and the indirect gradient, and this process makes the optimization more precise. During the training process, the parameters of the BN network and GMM are optimized jointly. Finally, we use model space discriminative training to refine the parameters of the GMM components. Experimental results show significant superiority of the proposed methods over the regular BN feature extractor. In addition, the experimental results on switchboard task demonstrate the indirect gradient is very important.

The paper is organized as follows: in section 2 we describe the method of discriminative bottleneck feature extractor with linear transformation; in section 3 we describe the overall training recipe; in section 4 we conduct the experiments and analyze experimental results, and in section 5 we present the conclusions.

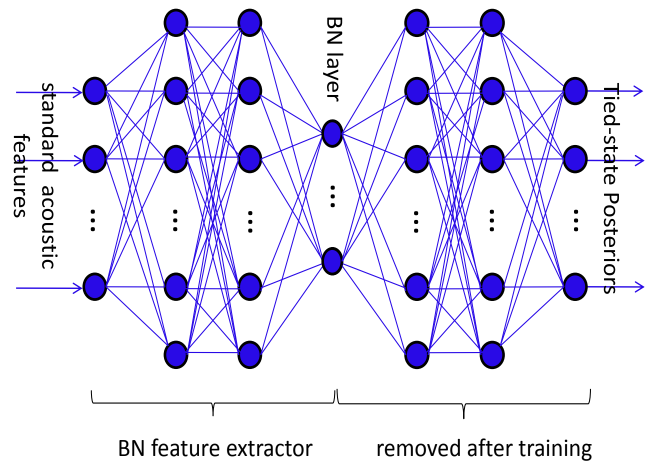

Fig. 1. Baseline Bottleneck Neural Network.

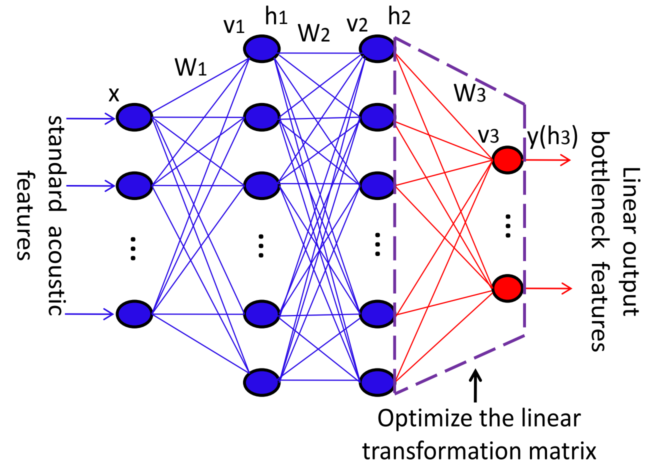
standard acoustic features
x
W1
v1
h1
W2
v2
h2
W3
v3
y(h3)
Linear output bottleneck features
Optimize the linear transformation matrix

Fig. 2. BN feature extractor.

2. DISCRIMINATIVE BOTTLENECK FEATURE EXTRACTOR WITH LINEAR TRANSFORMATION

2.1. A overview of bottleneck neural network

Figure 1 is our baseline bottleneck neural network .The architecture of this network is similar as described in [17, 22]. This network is a specially structured DNN, which includes a small bottleneck layer in the middle to control information flowing from the input layer to the output layer. The inputs to this network are standard acoustic features, such as MFCC, PLP. The output layer corresponds to senone activations. After training the bottleneck neural network, we get the BN feature extractor as shown in Figure2. The relationship of the parameters in figure 2 is

$$\begin{aligned} v_1 &= W_1 x, \quad h_1 = \sigma(v_1) \\ v_2 &= W_2 h_1, h_2 = \sigma(v_2) \\ y &= h_3 = v_3 = W_3 h_2 \end{aligned} \tag{1}$$

Where x denotes the standard acoustic features and y denotes linear output bottleneck feature. $W = \{W_1, W_2, W_3\}$ are the weights of BN feature extractor. The size of W_i is $n_i \times (n_{i-1} + 1)$ where n_i denotes the number of units of layer i. σ is the sigmoid function.

2.2. Discriminative training of bottleneck neural network with linear transformation

In this paper, we will only update the weights of W_3 (red part in figure2) discriminatively to optimize the BN feature extractor while remaining W_1 and W_2 to be fixed. From formula (1), the acoustic features x will be projected to h_2 though the weights W_1 and W_2, and we view h_2 as the raw features. Since $y = W_3 h_2$, we treat W_3 in the BN feature extractor as a linear transformation matrix M. We will train this matrix based on MPE criterion. The training procession bears similarity to work described in [5]. Comparing with fMPE, the transformation is more simple since the dimension of h_2 is far lower than the high-dimensional posterior features in fMPE.

2.3. The objective function

The objective function that we use in this paper is given by the minimum phone error (MPE) [4] between the bottleneck features y_r and the sequence of reference words w_r:

$$\begin{aligned} \mathcal{F}_{MPE}(\lambda) &= \sum_r \sum_{w_i \in \mathbf{w^r}} p(w_i|y_r) A(w_i, w_r) \\ &\approx \sum_r \sum_{w_i \in \mathbf{w^r}} \frac{p_\lambda(y_r|w_i)p(w_i)}{\sum\limits_{w_k \in \mathbf{w^r}} p_\lambda(y_r|w_k)p(w_k)} A(w_i, w_r) \end{aligned} \tag{2}$$

where $p(w_i|y_r)$ is defined as the posterior sentence probability of the hypothesized sentence w_r, λ is the model parameters and y_r is the r'th file of bottleneck features. The function $A(w_i, w_r)$ is the raw phone accuracy of w_i given w_r.

As (2) is difficult to optimize directly, we usually optimize the weak-sense auxiliary function to maximize (2).

$$g_{MPE}(\lambda, \bar{\lambda}) = \sum_r \sum_{q \in W^r_{lat}} \sum_{t=s_q}^{t=e_q} \sum_m \gamma_q^{rMPE} \gamma_{qm}^r(t) logN(y_r(t), \mu_m, \sigma_m) \tag{3}$$

Where γ_q^{rMPE} denotes the weight of minimum phone error of the q'th phone arc and $\gamma_{qm}^r(t)$ denotes the Gaussian occupation probability within the phone arc at time t.

2.4. Optimization of the linear transformations

The matrix is trained using linear method with gradient algorithm. The update on each iteration is:

$$M_{ij} := M_{ij} + \eta \frac{\partial \mathcal{F}}{\partial M_{ij}} \tag{4}$$

Where η is the learning rate, and $\frac{\partial \mathcal{F}}{\partial M_{ij}}$ is the differential of object function to M_{ij}.

$$\frac{\partial \mathcal{F}}{\partial M_{ij}} = \sum_{t=1}^{T} \frac{\partial \mathcal{F}}{\partial y_{ti}} h_{tj} \tag{5}$$

Where $\frac{\partial \mathcal{F}}{\partial y_{ti}}$ corresponds to the differential of the objective function w.r.t. the i'th dimension of BN features at time t. T corresponds to the total frames of all the training data. As is mentioned above, we only update red part of BN feature extractor in figure 2, and this can be treated as linear transform matrix update. We can use all the training data as a big batch rather than small mini-batches to calculate the differentials of the parameters, and this is particularly suited for distributed computing.

The differential $\frac{\partial \mathcal{F}}{\partial y_{ti}}$ includes a direct and an indirect component because the model parameters are also a function of the transformed feature vectors.

$$\frac{\partial \mathcal{F}(M,\overline{M})}{\partial y_{t,i}} = \underbrace{\frac{\partial \mathcal{F}(y_{t,i},\overline{M})}{\partial y_{t,i}}}_{\text{direct}} + \underbrace{\frac{\partial \mathcal{F}(y_{t,i},\overline{M})}{\partial \lambda}\frac{\partial \lambda}{\partial y_{t,i}}}_{\text{indirect}} \tag{6}$$

The direct derivative is

$$\frac{\partial \mathcal{F}(y_{t,i},\overline{M})}{\partial y_{t,i}} = \sum_{q \in W_{lat}^{r}} \sum_{m} \gamma_q^{MPE} \gamma_{qm}(t) \frac{\mu_{m,i} - y_{t,i}}{\sigma_{m,i}^2} \tag{7}$$

where $\mu_{m,i}, \sigma_{m,i}^2$ represent the Gaussian mean and variance, which are assumed to be estimated with maximum likelihood on the BN features.

The indirect derivative is obtained by differentiating the object function with respect to $\mu_{m,i}$ and $\sigma_{m,i}^2$.

$$\begin{aligned}&\frac{\partial \mathcal{F}(y_{t,i},\overline{M})}{\partial \lambda}\frac{\partial \lambda}{\partial y_{t,i}} = \\ &\sum_{m} \frac{\partial \mathcal{F}(y_{t,i},\overline{M})}{\partial \mu_{m,i}}\frac{\partial \mu_{m,i}}{\partial y_{t,i}} + \sum_{m} \frac{\partial \mathcal{F}(y_{t,i},\overline{M})}{\partial \sigma_{m,i}^2}\frac{\partial \sigma_{m,i}^2}{\partial y_{t,i}}\end{aligned} \tag{8}$$

The calculation of (8) is similar as described in [5] and will not be detailed here. By using Eq.(6), (7) and (8), we can optimize M with Eq.(5).

3. OVERALL TRAINING RECIPE

Overall training recipe is as follows:

1. Obtain frame-level labels of the training corpus through a standard GMM-HMM system.
2. Train a BN neural network using cross entropy criterion.
3. Train a BN feature based GMM-HMM system as the baseline.
4. Optimize BN feature extractor with the proposed method.
5. Train the GMM-HMM system using model space MPE criterion.

In step 1 the acoustic model (AM) of the GMM-HMM system starts with some iterations of ML estimation, and then is refined by several iterations of model space MPE training. The model in step 1 is adopted to get frame-level labels of all the training data. Then we train a standard BN neural network based on cross-entropy criterion in step 2. In step3, the BN feature extractor is used to transform the acoustic features to BN features, and the BN features are used to replace the acoustic features in step 1 to train another GMM-HMM system. In step 4 we refine the BN feature extractor with the proposed method. Just like fMPE, each training iteration involves three passes over the data: one to accumulate normal MPE statistics, a second to accumulate statistics of optimizing BN feature extractor and update the weights based on gradient algorithm, and a third pass to update the GMM-HMM with the newly transformed data. In step 5, model space MPE training is used to optimize the GMM-HMM system in step 4.

4. EXPERIMENTS

In this paper, we have evaluated the proposed method on two LVCSR tasks, namely the 70-hours Mandarin transcription task and the Switchboard task.

4.1. Mandarin transcription task

For the Mandarin transcription task, the training set contains 76,858 utterances (about 70 hours) from 1,539 speakers. The test set contains 3,720 utterances from 50 other speakers, about 3-hour speech. All speech are very clean. Evaluation is measured in terms of character error rate (CER).

4.1.1. Baseline Systems

First of all, we build the baseline system based on the standard tied-state cross-word tri-phone GMM/HMMs. We use the regular 43-dimension features, including 39-dimension MFCC features (static, first and second derivatives) and 4-dimension pitch features. The features are pre-processed with cepstral mean normalization (CMN). The baseline models are first trained based on maximum likelihood estimation (MLE), including 3,978 tied states and 30 Gaussian components per state. Then model space discriminative training is performed using MPE criterion. In the first row of Table 1, we give recognition performance of the MFCC-based GMM/HMM.

Next, a 5-hidden-layer bottleneck neural network is trained using the standard procedure. The bottleneck layer in the middle contains 43 hidden nodes, and each other hidden layer has 2048 nodes. The BN neural networks' inputs are long concatenated feature vectors, stacking from all consecutive frames within a context window of 11 frames. We use RBM pre-training to initialize this network and train it in mini-batches of 1024 frames based on cross-entropy error criterion. After training, we use the BN feature extractor to extract BN features and train another GMM/HMM. The recognition performance of BN-based GMM/HMM is listed in the second row of Table 1. We can see that BN-based GMM/HMM yields a significant performance improvement over the MFCC-based GMM-HMM. The baseline results are same as reported in [22].

4.1.2. Lattice based system

Following the training recipe described in section 3, we get the optimized BN feature extractor and GMM-HMM. The recognition performance is listed in the last row of Table 1. It can be seen that the MLE-trained GMM-HMMs using optimized BN features give an 8.8% relative error reduction over baseline BN features. After MPE discriminative training, it yields a 7.4% relative error reduction over the same MPE-trained GMM-HMM using the baseline BN features.

Table 1. Performance comparison (CER in %) using different features in Mandarin task.

	MLE	MPE
MFCC	18.2	16.7
BN	13.6	12.2
LAT-BN	12.4	11.3

Table 2. Performance comparison (WER in %) using different features in Switchboard task.

	MLE	MPE
PLP	28.7	24.7
BN	18.3	16.1
LAT-BN	16.3	15.0

4.2. Switchboard task

For the Switchboard task, the training data consists of 300-hour Switchboard-I training set and 20-hour Call Home English data. We use the switchboard part of NIST 2000 Hub5 evaluation sets, denoted as Hub5e00-swb, to evaluate recognition performance.

4.2.1. Baseline Systems

The baseline system is a standard tied-state cross-word triphone GMM/HMMs trained with both MLE and MPE criterion using 39-dimension PLP features (static, first and second derivatives) that are pre-processed with cepstral mean and variance normalization (CMVN) per conversation side. GMM/HMM consists of 8,991 tied states and 40 Gaussians per state. In decoding, we use a tri-gram LM trained with all training transcripts. Next, we train a BN neural network with 429, 2048, 2048, 39, 2048, 2048, 8991 units in each layer, and train a GMM/HMM using 39-dimension BN features. The middle two rows of table 2 list the WERs of these baseline GMM/HMMs.

4.2.2. Lattice based system

In the last row of Table 2, we list the WERs of the lattice based system. As we can see, the performance improvements are very significant comparing with the baseline BN GMM/HMM. The relative word error reductions are 10.9% and 6.8% for the MLE and MPE systems, respectively.

4.3. performance of keeping GMM model fixed

Furthermore, we conduct experiments on the switchboard task to evaluate the importance of indirect part of formula (6). While ignoring the indirect part, the parameters of GMM model are fixed during the training process. It is similar to the method of [21], the difference is that we only update the weights in the bottleneck layer with a big batch rather than update the whole network with mini-batches. The comparison of recognition performance is given in Table 3. Results in the first row are copied from the last row of table 2, which means the GMM model is updated in the optimization procedure. The results of keeping GMM model fixed are listed in the last row.

The numbers shown in brackets in table 3 correspond to the decoding results while using BN features in combination with the ML trained AM of the baseline BN system. ML re-training the AM using BN features results in a slight increase in WER. Despite this increase in WER, the subsequent discriminative AM training benefits from such a re-training. This phenomenon is same as the results described in [21]. Results show that the performance of the LAT-BN without updating model parameters is slightly better than the baseline BN features, but is significant worse than the LAT-BN with updating GMM model. This results shown that the indirect gradient is an imperative part of formula (6), and it's not wise to ignore this part.

Table 3. Performance (WER in %) comparison of different LAT-BN in Switchboard task.

LAT-BN	MLE	MPE
update GMM model	16.3	15.0
GMM model fixed	17.9(17.2)	15.7

5. CONCLUSION

In this paper, we propose a method to optimize bottleneck feature extractor for GMM-HMM based speech recognition. Our approach allows to train BN feature extractor using lattice-based sequence classification criteria. Different from the traditional training strategies of BN neural networks, this method is suited for distributed computing. More importantly, we have demonstrated that our approach can yield significant performance gain compared to regular BN feature extractor in all evaluated speech recognition tasks. As we only update the weights of BN layer in this paper, future work will address ways of updating all layer weights in the BN feature extractor.

6. ACKNOWLEDGEMENT

This work was supported by the Science and Technology Development of Anhui Province, China (Grant No. 13Z02008-5) and the National Nature Science Foundation of China (Grant No. 61273264).

7. REFERENCES

[1] S. Kapadia, V. Valtchev, and S. J. Young, "MMI training for continuous phoneme recognition on the TIMIT database," in *Proc. ICASSP*, 1993, pp. 491–494.

[2] B. H. Juang, W. Chou, and C. H. Lee, "Minimum classification error rate methods for speech recognition," *IEEE Trans. Speech Audio Process*, vol. 5, no. 3, pp. 257–265, 1997.

[3] E. McDermott, T. Hazen, J. L. Roux, A. Nakamura, and S. Katagiri, "Discriminative training for large-vocabulary speech recognition using minimum classification error," *IEEE Trans. Speech Audio Process*, vol. 15, no. 1, pp. 203–223, 2007.

[4] D. Povey and P. Woodland, "Minimum phone error and I-smoothing for improved discriminative training," in *Proc. ICASSP*, 2002, pp. 105–108.

[5] D. Povey, B. Kingsbury, L. Mangu, G. Saon, H. Soltau, and G. Zweig, "fMPE: Discriminatively trained features for speech recognition," in *Proc. ICASSP*, 2005, pp. 961–964.

[6] D. Povey, D. Kanevsky, B. Kingsbury, B. Ramabhadran, G. Saon, and K. Visweswariah, "Boosted MMI for model and feature-space discriminative training," in *Proc. ICASSP*, 2008, pp. 4057–4060.

[7] G. Saon and B. Kingsbury, "Discriminative feature-space transforms using deep neural networks," in *Proc. INTERSPEECH*, 2012.

[8] B. Zhang, S. Matsoukas, and R. Schwartz, "Discriminatively trained region dependent feature transforms for speech recognition," in *Proc. ICASSP*, 2006, pp. 313–316.

[9] G. E. Dahl, D. Yu, L. Deng, and A. Acero, "Context-dependent pre-trained deep neural networks for large vocabulary speech recognition," *IEEE Trans. Speech Audio Process*, vol. 20, no. 1, pp. 30–42, 2012.

[10] T. N. Sainath, A. Mohamed, B. Kingsbury, and B. Ramabhadran, "Deep convolutional neural networks for LVCSR," in *Proc. ICASSP*, 2013, pp. 8614–8618.

[11] A. Mohamed, G. E. Dahl, and G. E. Hinton, "Deep belief networks using discriminative features for phone recognition," in *Proc. NIPS Workshop on Deep Learning for Speech Recognition and Related Applications*, 2009.

[12] D. Yu, L. Deng, and G. E. Dahl, "Roles of pre-training and fine-tuning in context-dependent DNN-HMMs for real-world speech recognition," in *Proc. NIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2010.

[13] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[14] B. Kingsbury, "Lattice-based optimization of sequence classification criteria for neural network acoustic modeling," in *Proc. ICASSP*, 2009, pp. 3761–3764.

[15] J. S. Bridle and L. Dodd, "An Alphanet approach to optimising input transformations for continuous speech recognition," in *Proc. ICASSP*, 1991, pp. 277–280.

[16] B. Kingsbury, T. N. Sainath, and H. Soltau, "Scalable minimum bayes risk training of deep neural network acoustic models using distributed Hessian-free optimization," in *Proc.INTERSPEECH*, 2012.

[17] F. Grezl, M. Karafiat, S. Kontar, and J. Cernocky, "Probabilistic and bottle-neck features for LVCSR of meetings," in *Proc. ICASSP*, 2007, pp. 757–760.

[18] H. B. Hu and S. A. Zahorian, "A neural network based nonlinear feature transformation for speech recognition," in *Proc. INTERSPEECH*, 2008, pp. 1533–1536.

[19] D. Yu and M. L. Seltzer, "Improved bottleneck features using pretrained deep neural networks," in *Proc. INTERSPEECH*, 2011, pp. 237–240.

[20] T. N. Sainath, B. Kingsbury, and B. Ramabhadran, "Autoencoder bottleneck features using deep belief networks," in *Proc. ICASSP*, 2012, pp. 4153–4156.

[21] M. Paulik, "Lattice-based training of bottleneck feature extraction neural networks," in *Proc. INTERSPEECH*, 2013, pp. 89–93.

[22] Y. B. Bao, H. Jiang, L. R. Dai, and C. Liu, "Incoherent training of deep neural networks to de-correlate bottleneck features for speech recognition," in *Proc. ICASSP*, 2013, pp. 6892–6894.