Deep Neural Network Training Emphasizing Central Frames

Gakuto Kurata[1], *Daniel Willett*[2]

[1]IBM Research
[2]Nuance Communications

gakuto@jp.ibm.com, Daniel.Willett@nuance.com

Abstract

It is common practice to concatenate several consecutive frames of acoustic features as input of a Deep Neural Network (DNN) for speech recognition. A DNN is trained to map the concatenated frames as a whole to the HMM state corresponding to the center frame while the side frames close to both ends of the concatenated frames and the remaining central frames are treated as equally important. Though the side frames are relevant to the HMM state of the center frame, this relationship may not be fully generalized to unseen data. Thus putting more emphasis on the central frames than on the side frames avoids overfitting to the DNN training data. We propose a new DNN training method to emphasize the central frames. We first conduct pre-training and fine-tuning with only the central frames and then conduct fine-tuning with all of the concatenated frames. In large vocabulary continuous speech recognition experiments with more than 1,000 hours of data for DNN training, we obtained a relative error rate reduction of 1.68%, which was statistically significant.

Index Terms: Deep Neural Network (DNN), Concatenated Frames, Bottleneck Feature (BNF), Large Vocabulary Continuous Speech Recognition (LVCSR)

1. Introduction

A Deep Neural Network (DNN) can be used as a feature extractor for Gaussian Mixture Model-Hidden Markov Model (GMM-HMM) systems and Acoustic Models (AMs) for DNN-HMM systems in speech recognition [1, 2, 3, 4, 5, 6, 7]. A DNN for speech recognition typically consists of an input layer that takes several concatenated frames of acoustic features, many hidden layers, and an output layer that predicts the HMM state corresponding to the *center frame* at the center of the concatenated input frames.

In speech recognition research, concatenating consecutive frames is a widely-used approach, referred to as acoustic context expansion. For example, in feature-space discriminative training, the posterior probabilities for preceding and succeeding frames are considered [8]. However, several of the preceding and succeeding frames are averaged out and this causes the center frame to be relatively emphasized.

In DNN for speech recognition, a fixed number of frames are concatenated for input and the concatenated frames are treated equally, as shown in Figure 1. Here, we call the few frames from both ends of the concatenated frames the *side frames* and the remaining frames the *central frames*. Considering that the output layer predicts the HMM state corresponding to the center frame, the concatenated frames as a whole are associated with the HMM state of the center frame. However, though the acoustic features of the side frames are related to the HMM state of the center frame, they may also contain irrelevant information. The relationship between the side frames and the HMM state of the center frame may not be fully generalized to unseen data. Therefore relying too much on the side frames rather than on the central frames has a risk of over-fitting to the DNN training data. In other words, emphasizing the central frames can reduce the risk of over-fitting.

In this paper, we propose a new method to emphasize the central frames in DNN training. We first conduct pre-training and fine-tuning with only the central frames. The resulting DNN relies on the central frames to predict the target HMM state. Then starting from this DNN, we conduct fine-tuning with all of the concatenated frames. Since the DNN after the first stage of fine-tuning can predict the target HMM state to some extent from just the central frames, the weights between the side frames and the units in the second layer become small after the second stage of fine-tuning, which reduces the risk of over-fitting to the training data.

The rest of this paper is organized as follows. Section 2 describes the proposed method in detail after summarizing the normal DNN training procedures. Then Section 3 explains the experiments that we conducted to show the advantages of the proposed method, followed by some investigations on the trained DNN. Finally Section 4 concludes this paper.

2. Proposed method

We first revisit a typical training flow of DNN for speech recognition. Then we describe our proposed method, *two-stage fine-tuning*, to emphasize the central frames. We also explain another approach to emphasize the central frames in which we introduce regularization on the weight updates between the side frames in the input layer and the units in the second layer. Finally, we introduce a related work that has a similar motivation.

2.1. Normal DNN training

Typical DNN training consists of two steps. The first step, pre-training, stacks the layers up to a pre-defined DNN topology with initializing weights. Recent advances in deep learning owe a lot to better weight initialization with unsupervised generative approaches, originally by Restricted Bolzmann Machines (RBMs) [9] and more recently by a type of autoencoder [10]. Instead of using unsupervised generative pre-training, supervised discriminative pre-training is also widely used in DNN training for speech recognition [11, 12, 13, 14]. Once the layers are stacked and the weights are initialized in the pre-training, fine-tuning that discriminatively updates the weights with error backpropagation using the cross-entropy objective function follows [15]. To further boost the speech recognition accuracy, sequential training has been added recently, which we don't address in this paper [16, 17, 18, 19].
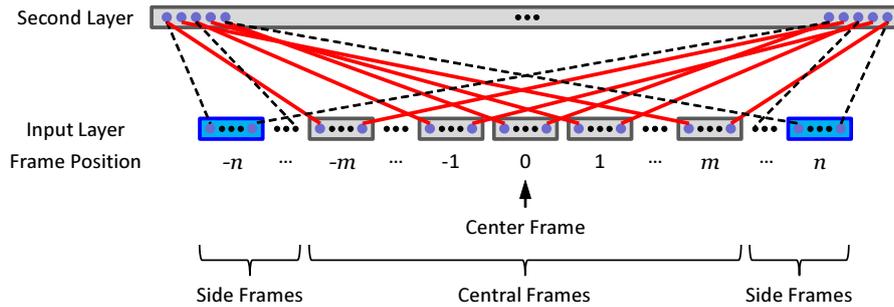
Figure 1: *Connections between input and second layers. We call the frame that is at the center of the concatenated frames the **center frame** whose associated HMM state is predicted in the DNN. We call the frames at positions from $-m$ to m the **central frames** and the remaining frames the **side frames**. The connections between the central frames and the units in the second layer are emphasized in the proposed method. Please note that only a fraction of the connections are depicted to reduce the complexity.*
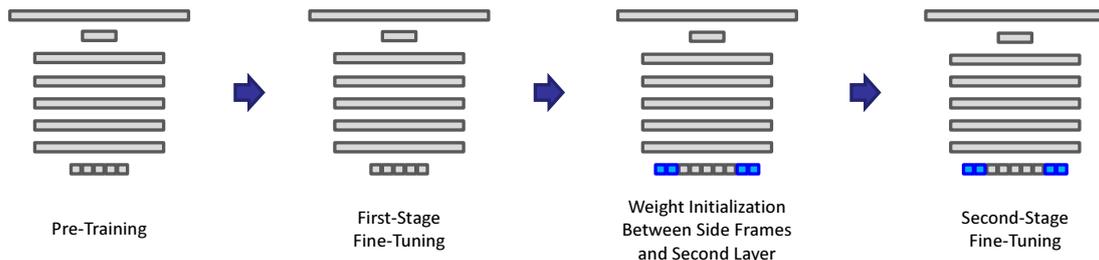

Figure 2: *Flow of two-stage fine-tuning. We first conduct pre-training and fine-tuning with only the central frames and then conduct fine-tuning with all of the concatenated frames.*

2.2. Two-stage fine-tuning

We propose a new DNN training method to put emphasis on the central frames, which corresponds to putting larger weights on the bold red connections in Figure 1. The motivation of the proposed method is that though the DNN represents the mapping between the concatenated frames and the HMM states of the center frame with treating all of the frames equally, the relationship between the side frames and the HMM state of the center frame may not be fully generalized to the unseen data and thus relying too much on the side frames has the risk of over-fitting to the DNN training data.

Figure 2 shows the overall procedure of the proposed two-stage fine-tuning. In the following explanation, we assume that $2n+1$ frames are used for the DNN input and the central $2m+1$ frames are emphasized as shown in Figure 1 where $m < n$.

Pre-training: We conduct pre-training by setting up a DNN with a topology that takes $2m + 1$ concatenated frames at positions from $-m$ to m as input. Please note that before pre-training, we initialized the weights with random values according to the normalized initialization introduced in [20].

First-stage fine-tuning: We conduct first-stage fine-tuning with $2m + 1$ concatenated frames.

Weight initialization: We modify the DNN topology so that it takes $2n + 1$ concatenated frames at positions from $-n$ to n as input. In this process, we initialize the weights of the connections between the side frames and the second layer (which are depicted with dashed lines in Figure 1) by the same method used in the pre-training [20] while we keep the weights between the central frames and the second layer estimated in the first-stage fine-tuning. For other layers above the second layer, we keep the weights estimated in the first-stage fine-tuning.

Second-stage fine-tuning: We conduct second-stage fine-tuning with $2n + 1$ concatenated frames while using the same training dataset as was used in the first-stage fine-tuning. Please note that all of the weights in the DNN are tuned in this step.

Since the second-stage fine-tuning starts from the DNN that predicts the target HMM state from the central frames to some extent, the side frames work secondarily in the DNN after the second-stage fine-tuning.

2.3. Regularization on side frames

We propose an alternative training method to put emphasis on the central frames instead of two-stage fine-tuning. The idea is to put regularization terms during the weight update of the connections between the side frames and the second layer. The weight w in the DNN is typically updated with a back-propagation algorithm as $w = w - \alpha \frac{1}{s} \Delta w$,where α is a learning rate and s is the number of utterances in each mini-batch. In the proposed method, while we continue using the equation above for the center frame, we introduce a regularization term λ for the connections between the other frames in the input layer and all of the units in the second layer as $w = w - \alpha(\frac{1}{s} \Delta w + \lambda w)$,where larger λ is used for the frames further from the center frame.

This method uses $2n + 1$ frames as input for the DNN from the beginning and doesn't require fine-tuning twice.

2.4. Related work

[21] also pointed out that mapping the concatenated input frames to the HMM state of the center frame has drawbacks and proposed a method to predict the HMM states of the multiple frames rather than the center frame from the concatenated input frames. Though the motivation is similar, [21] takes care of the output side of the DNN while we focus on the input side of the DNN.

Table 1: *Two-stage fine-tuning and regularization on side frames with 50 hours of training data. (Character Error Rate (CER), CER Reduction (CERR) from normal training, Kana Error Rate (KER) and KER Reduction (KERR) from normal training are shown.)*

DNN training	Input frames	CER (CERR) [%]			KER (KERR) [%]		
		Task 1	Task 2	Average	Task 1	Task 2	Average
Normal	11	20.45	31.67	26.06	12.25	15.21	13.73
Two-stage fine-tuning	$1 \to 11\ (m = 0)$	20.44 (0.06)	32.22 (-1.71)	26.33 (-0.83)	12.25 (0.00)	15.35 (-0.93)	13.80 (-0.44)
	$3 \to 11\ (m = 1)$	20.17 (1.39)	31.12 (1.74)	25.64 (1.57)	12.11 (1.16)	14.95 (1.70)	13.53 (1.43)
	$\mathbf{5 \to 11\ (m = 2)}$	20.12 (1.61)	31.01 (2.09)	**25.57 (1.85)**	12.09 (2.32)	14.71 (3.30)	**13.40 (2.29)**
	$7 \to 11 (m = 3)$	19.96 (2.40)	31.44 (0.73)	25.70 (1.57)	11.97 (3.55)	14.97 (1.60)	13.47 (1.95)
	$9 \to 11\ (m = 4)$	20.21 (1.20)	31.42 (0.80)	25.81 (1.00)	12.15 (1.33)	14.92 (1.95)	13.53 (1.38)
Regularization	11	20.08 (1.79)	31.39 (0.88)	25.74 (1.34)	12.04 (1.69)	14.79 (2.75)	13.42 (2.22)

Table 2: *Two-stage fine-tuning with more than 1,000 hours of training data. (Character Error Rate (CER), CER Reduction (CERR) from normal training, Kana Error Rate (KER) and KER Reduction (KERR) from normal training are shown.)*

DNN training	CER (CERR) [%]			KER (KERR) [%]		
	Task 1	Task 2	Average	Task 1	Task 2	Average
Normal	16.28	26.80	21.54	9.14	11.48	10.31
Two-stage fine-tuning	16.12 (1.00)	26.42 (1.41)	**21.27 (1.20)**	9.05 (1.00)	11.20 (2.37)	**10.12 (1.68)**

3. Experiments

We conducted Large Vocabulary Continuous Speech Recognition (LVCSR) experiments in Japanese to verify whether speech recognition accuracy is improved by using the proposed DNN training methods[1]. First we conducted preliminary experiments with 50 hours of training data with various configurations. Then we conducted large-scale experiments with more than 1,000 hours of training data to confirm the advantages of the proposed method. Finally we investigated the weight magnitudes of the DNNs trained with the normal training method and the proposed method.

3.1. Experimental setup

We used 11 concatenated frames of 31 dimensional Mel-Frequency Cepstral Coefficient (MFCC) features for the input of the DNN. The DNN topology we used was an input layer of 341 ($= 11 \times 31$) units, 5 hidden layers of 1,024 units, a bottleneck layer of 40 units, and an output layer of 3,000 units. We used the senone (tied triphone HMM states) for the target in the DNN training [22]. For the pre-training and fine-tuning, both used error backpropagation with a cross-entropy objective function. During DNN training, we prepared a held-out set and monitored the Phone Error Rate (PER) on it while controlling the learning rate using a recipe similar as [23].

After the DNN training was completed, we used the trained DNN for bottleneck feature (BNF) extraction [4, 5] and trained the GMM with the extracted 40-dimension BNFs from the bottleneck layer. For the GMM training, we conducted Maximum Likelihood (ML) training in the preliminary experiments in Section 3.2. In the large-scale experiments in Section 3.3, we conducted ML training followed by feature-space and model-space discriminative training [24, 25].

The training data and the evaluation data were recorded using mobile phones. The evaluation data has 2 tasks: Tasks 1 and 2 contain dictation type and voice search type utterances, respectively, with both sets containing more than 10,000 utterances.

We prepared the language models (LMs) and the lexicons for each task. The LM training corpora for each task consist of a total of more than 1 billion words and we linearly interpolated

a word 4-gram model and a class 3-gram model [26]. The size of the lexicon for each task was almost 1 million.

The evaluation metrics we used were the Character Error Rate (CER) and Kana Error Rate (KER). For the CER, the reference transcripts and the recognized results were split into characters and compared character by character. For the KER, the reference transcripts and the recognized results were converted into Kana expressions and compared Kana by Kana. Please note that the Kana are Japanese characters that express approximate pronunciations. Since Japanese has ambiguity in the word segmentation and word spelling, we used CER and KER for Japanese LVCSR evaluation instead of the Word Error Rate (WER) that is widely used for English and some other languages.

3.2. Preliminary experiments

We conducted preliminary experiments using 50 hours of training data for the DNN and GMM training. For the baseline system, we used the normal training method that consists of pre-training and fine-tuning using all 11 concatenated frames.

The purpose of the preliminary experiments was to investigate how many central frames need to be trained in the pre-training and first-stage fine-tuning under the framework of the two-stage fine-tuning proposed in Section 2.2. Thus we first trained the DNNs with various numbers of central frames and then with all 11 concatenated frames. Using the expression in Figure 1, we kept n to 5 and changed m from 0 to 4. Please note that the topologies of the DNNs from the normal training method and the proposed training method are exactly the same.

The results are shown in Table 1. Except for the case of $m = 0$, we found accuracy improvements with the proposed two-stage fine-tuning. The best result was obtained when we set $m = 2$, so we first trained with the central 5 frames and then with all 11 frames, resulting in 1.85% CER reduction and 2.29% KER reduction on average for the 2 tasks. We conducted the large-scale experiment that appears in the next subsection with this configuration.

Please note that in both the first-stage and second-stage fine-tuning, we iterated the training until the PER on the held-out set was well saturated. Thus this improvement is not due to simply increasing the accumulated number of epochs.

Another purpose of the preliminary experiments was to confirm the effects of the regularization approach described in
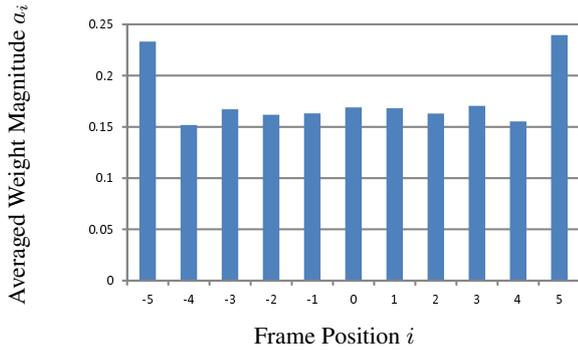
---

[1]Please note that our proposed method doesn't contain any Japanese specific techniques.

Figure 3: *Averaged weight magnitudes between each frame in the input layer and the second layer after normal DNN training.*

Figure 4: *Averaged weight magnitudes between each frame in the input layer and the second layer after two-stage fine-tuning ($m = 2, n = 5$).*

Section 2.3. We tried various numbers of regularization terms and the best result is shown in Table 1 when the regularization terms were set to $1e-2$ for the frames at ± 5, to $1e-3$ for the frames at ± 4, to $1e-4$ for the frames at ± 3, to $1e-5$ for the frames at ± 2, and to $1e-6$ for the frames at ± 1. We reduced the CER by 1.34% and the KER by 2.22% on average for the 2 tasks. However, this improvement was smaller than the improvement from the two-stage fine-tuning. Thus, we used the two-stage fine-tuning in the large-scale experiments in the next subsection.

3.3. Large-scale experiments

We conducted large-scale experiments using more than 1,000 hours of training data for the DNN and GMM training. For the baseline system, we used the normal training method that consists of pre-training and fine-tuning using all 11 concatenated frames. For the proposed two-stage fine-tuning, we first trained with the central 5 frames and then with all 11 frames, which was the best configuration in the preliminary experiment.

The result is shown in Table 2. We reduced the CER by 1.20% and the KER by 1.68% on average for the 2 tasks while obtaining consistent improvements for both tasks. These improvements were statistically significant and support the advantages of the proposed two-stage fine-tuning.

3.4. Analysis on connection weights

We investigated the connection weights between the input layer and the second layer in the DNNs trained with the normal training method and the proposed method in the large-scale experiments described in Section 3.3.

We calculated a_i, the average of the weight magnitudes of the connections between all of the coefficients in the frame at position i and all of the units in the second layer. The definition of a_i is

$$a_i = \frac{\sum_{d=1,\cdots,31} \sum_{u=1,\cdots,1024} |w_{du}^i|}{31 \times 1024} ,$$

where w_{du}^i is the weight between the d-th coefficient of the input frame at position i and the u-th unit in the second layer. Please note that the dimension of the input MFCC is 31, the frame position i is from -5 to 5, and the number of units in the second layer is 1,024.

The average weight magnitudes for the DNNs trained with the normal training method and the proposed method are shown in Figure 3 and Figure 4, respectively. By comparing the two figures, we can see th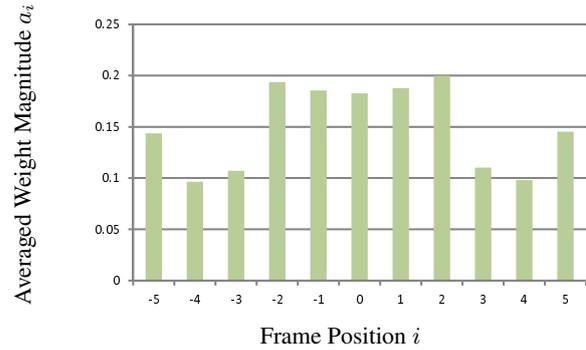at the proposed method increased the weight magnitudes for the central 5 frames at the positions from -2 to $+2$ and suppressed the weight magnitudes for the 3 side frames on both sides at the positions from -5 to -3 and $+3$ to $+5$. Our backgrounding idea to emphasize the central frames was realized by the proposed training method, as expected.

We noticed that the weight magnitudes at both ends were large in the original training method. By using the proposed method, the weight magnitudes at both ends were suppressed, but still large. There are two reasons for this. The first reason is that the frames at both ends contain the information of the more distant frames at the positions $\pm o$ where $o > 5$. The other reason is that the delta feature is important in speech recognition [27]. We only used the static features as the input for the DNN. The DNN automatically learns the delta features and the resulting weight magnitudes at both ends become large.

4. Conclusion

In this paper, we proposed a method to train a DNN while emphasizing the central frames. This was motivated by the idea that relying too much on the side frames in DNN training has a risk of over-fitting to the training data.

From our LVCSR experiments, we can conclude that the two-stage fine-tuning in which we first train the DNN with the central frames alone and then with all of the concatenated frames improved the speech recognition accuracy. Investigation of the weight magnitudes between the input and second layers confirmed that the proposed method increased the weight magnitudes for the central frames while it suppressed the weight magnitudes for the side frames.

We limited the number of fine-tuning stages to 2 in this paper. Gradually expanding the central frames such as training with 5, 7, 9, and then 11 frames in order might be worth trying. In addition, generally speaking, increasing the number of frames for DNN input improves speech recognition accuracy. Increasing the number of input frames with emphasizing central frames should also be explored.

We conducted experiments in which a trained DNN is used for BNF extraction. During the DNN training, we didn't use any idea that is specific to BNF extraction. Thus our idea should be also applicable in DNN training for DNN-HMM systems.

We tried frame-wise cross-entropy training in this paper. Starting sequential training from better model improved by our proposed method might be beneficial, which is our future work.

5. Acknowledgment

6. References

[1] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, and B. Kingsbury, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 82–97, 2012.

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

[3] A. Mohamed, G. E. Dahl, and G. Hinton, "Acoustic modeling using deep belief networks," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 1, pp. 14–22, 2012.

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

[5] J. Gehring, Y. Miao, F. Metze, and A. Waibel, "Extracting deep bottleneck features using stacked auto-encoders," in *Proc. ICASSP*, 2013, pp. 3377–3381.

[6] L. Deng and D. Yu, "Deep learning: methods and applications," *Foundations and Trends in Signal Processing*, vol. 7, no. 3–4, pp. 197–387, 2014.

[7] D. Yu and L. Deng, *Automatic Speech Recognition - A Deep Learning Approach*, Springer.

[8] 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.

[9] G. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural computation*, vol. 18, no. 7, pp. 1527–1554, 2006.

[10] C. Plahl, T. N. Sainath, B. Ramabhadran, and D. Nahamoo, "Improved pre-training of deep belief networks using sparse encoding symmetric machines," in *Proc. ICASSP*, 2012, pp. 4165–4168.

[11] H. Soltau, H. Kuo, L. Mangu, G. Saon, and T. Beran, "Neural network acoustic models for the DARPA RATS program.," in *Proc. Interspeech*, 2013, pp. 3092–3096.

[12] T. N. Sainath, B. Kingsbury, and B. Ramabhadran, "Improving training time of deep belief networks through hybrid pre-training and larger batch sizes," in *Proc. NIPS Workshop on Log-linear Models*, 2012.

[13] F. Seide, G. Li, X. Chen, and D. Yu, "Feature engineering in context-dependent deep neural networks for conversational speech transcription," in *Proc. ASRU*, 2011, pp. 24–29.

[14] H. Larochelle, Y. Bengio, J. Louradour, and P. Lamblin, "Exploring strategies for training deep neural networks," *The Journal of Machine Learning Research*, vol. 10, pp. 1–40, 2009.

[15] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Cognitive modeling*, 1988.

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

[17] A. Mohamed, D. Yu, and L. Deng, "Investigation of full-sequence training of deep belief networks for speech recognition.," in *Proc. Interspeech*, 2010, pp. 2846–2849.

[18] 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.

[19] K. Veselỳ, A. Ghoshal, L. Burget, and D. Povey, "Sequence-discriminative training of deep neural networks.," in *Proc. Interspeech*, 2013, pp. 2345–2349.

[20] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proc AISTATS*, 2010, pp. 249–256.

[21] N. Jaitly, V. Vanhoucke, and G. Hinton, "Autoregressive product of multi-frame predictions can improve the accuracy of hybrid models," in *Proc. Interspeech*, 2014.

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

[23] T. N. Sainath, B. Kingsbury, B. Ramabhadran, P. Fousek, P. Novak, and A. Mohamed, "Making deep belief networks effective for large vocabulary continuous speech recognition," in *Proc. ASRU*, 2011, pp. 30–35.

[24] S. F. Chen, B. Kingsbury, L. Mangu, D. Povey, G. Saon, H. Soltau, and G. Zweig, "Advances in speech transcription at IBM under the DARPA EARS program," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 14, pp. 1596–1608, 2006.

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

[26] S. F. Chen and J. Goodman, "An empirical study of smoothing techniques for language modeling," *Computer Speech & Language*, vol. 13, no. 4, pp. 359–393, 1999.

[27] S. Furui, "Cepstral analysis technique for automatic speaker verification," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 29, no. 2, pp. 254–272, 1981.