SEQUENCE DISTILLATION FOR PURELY SEQUENCE TRAINED ACOUSTIC MODELS

Naoyuki Kanda, Yusuke Fujita, Kenji Nagamatsu

Hitachi Ltd., Japan

{naoyuki.kanda.kn,yusuke.fujita.su,kenji.nagamatsu.dm}@hitachi.com

ABSTRACT

This paper presents our exploration into teacher-student (TS) training for acoustic models (AMs) based on the lattice-free maximum mutual information technique. Whereas most previous studies of TS training used a frame-level distance between teacher and student models' distributions, we propose using the sequence-level temperatured Kullback-Leibler divergence as a metric for TS training. In our experiment on the AMI meeting corpus, we prepared a strong teacher model consisting of a convolutional neural network, time delay neural network, and long short-term memory, which had 47.7M parameters and achieved a state-of-the-art word error rate (WER) of 18.05%. Whereas the small student AM (10.8M params. and 19.72% WER) trained by a frame-level TS training was able to fill only 43% of the WER gap between teacher and student AMs, the student AM trained by the proposed method achieved a 18.23% WER, filling 89% of the WER gap from the teacher AM. We also show that the frame-level TS training sometimes even degrades the performance of the student model whereas the proposed method consistently improved the accuracy.

Index Terms— Deep learning, acoustic model, distillation, lattice-free MMI

1. INTRODUCTION

Teacher-student (TS) training is a technique to train a small student model that imitates a large and accurate teacher model [1, 2]. Because recent research into deep neural network (DNN)-based acoustic models (AMs) has suggested that very deep and complicated models achieve good results in many cases (such as deep convolutional neural networks (CNNs) [3], a combination of a CNN, long short-term memory (LSTM), and DNN [4, 5], or a combination of a time delay neural network (TDNN) and LSTM [6–8]), model compression techniques like TS training are worth pursuing for small-footprint and fast decoding.

Most studies on TS training have used frame-level Kullback-Leibler (KL)-divergence between teacher and student AMs as a training metric [1, 2, 9–12]. This is reasonable when the AM is trained on the basis of a frame-level criterion, such as the cross entropy (CE) loss. However, the best AMs are normally achieved by a sequence-level criterion, such as maximum mutual information (MMI), state-level minimum Bayes risk (sMBR) [13, 14], or recently proposed lattice-free MMI (LFMMI) [15]. To imitate the sequence-level quality of the teacher model, a sequence-level criterion should be used for TS training as well. The idea of using sequence-level KL divergence for TS training was first proposed by Kim and Rush for neural machine translation [16]. At around the same time, Wong and Gales proposed the same idea for MMI and sMBR DNN-HMM AMs [17]. Although they showed very promising results, their method relied on several approximations with additional N-best generation. Very recently, we showed a simpler, non-approximated form of error calculation of the sequence-level KL divergence based TS training [18]. In that work, we also presented an implementation that does not require any N-best or lattice generation by using a technique devised for LFMMI. In our experiment, our method achieved a good gain by TS training for LFMMI AMs, but an unignorable gap between teacher and student AMs still remained after TS training.

To overcome the above challenges, this paper presents our recent efforts on TS training based on a sequence-level criterion. First, we extend our previous work [18] by incorporating the temperature parameter into the sequence-level KL divergence. The temperature parameter was first introduced for TS training by Hinton et al. [2] to make the softmax output softer (i.e., more informative), which is known as "knowledge distillation." Their method was for frame-level criterion, so we extended the idea to the sequence-level KL divergence and explored the effectiveness of the temperature. [1] We also conducted a comprehensive exploration into TS training on LFMMI AMs by changing various parameters, such as training criteria, learning rates, initialization techniques, and student model sizes. Especially, we found that the frame-level TS training was sometimes even harmful, which has not been mentioned in previous literature.

Our experiments were based on LFMMI (purely sequence training) because of its state-of-the-art accuracy in many scenarios [8, 15, 19]. Because of its unique property that the AM does not have softmax output, some additional considerations are required for TS training in LFMMI. We discuss these details in Section 3.1.

2. LFMMI TRAINING OF AM

In MMI training, a neural-network parameter θ is estimated to maximize the criterion as follows.

$$\mathcal{F}^{MMI}(\theta) = \sum_u \log P_\theta(\mathbf{S}_u|\mathbf{X}_u). \quad (1)$$

Here, u indicates the index of a training utterance. The term $\mathbf{S}_u$ and $\mathbf{X}_u$ indicate the reference state sequence and the acoustic features of training utterance u, respectively. The error signal w.r.t. the final layer's output $y(u,t)$ [2] of the AM at the time frame t of utterance u is calculated as follows.

$$\frac{\partial \mathcal{F}^{MMI}(\theta)}{\partial y(u,t)} = \delta_{\mathbf{S}_u:y(u,t)} - \gamma^{DEN}_{\theta,y(u,t)}, \quad (2)$$

where $\delta_{\mathbf{S}_u:y(u,t)}$ is a delta function, which is 1 if the state corresponding to $y(u,t)$ is in $\mathbf{S}_u$, and 0 otherwise. The second term

[1] We noticed that Wong and Gales [17] suggested using the temperature as the acoustic and language model scaling but did not actually explore it.

[2] In the conventional DNN-HMM hybrid, y is the activation of the final softmax layer [14, 20]. On the other hand, LFMMI assumed that the AM directly outputs the pseudo-likelihood and that y is the output of the final layer.

$\gamma^{DEN}_{\theta,y(u,t)}$ represents a posterior probability of being in a state corresponding to $y(u,t)$, calculated on the basis of the current model parameter θ as follows,

$$\begin{aligned}\gamma^{DEN}_{\theta,y(u,t)} &= \sum_{\mathbf{S}} \delta_{\mathbf{S}:y(u,t)} P_\theta(\mathbf{S}|\mathbf{X}_u) \\ &= \frac{\sum_{\mathbf{S}} \delta_{\mathbf{S}:y(u,t)} P_\theta(\mathbf{X}_u|\mathbf{S}) P(\mathbf{S})}{\sum_{\mathbf{S}'} P_\theta(\mathbf{X}_u|\mathbf{S}') P(\mathbf{S}')}. \end{aligned} \quad (3)$$

In earlier studies, the decoded lattices created using CE-trained AMs were used to constrain the hypothesis space in Eq. (3) [13, 14, 20]. However, this requires redundant CE training of AMs. In addition, the accuracy of MMI training could be limited because the parameters could fall into the local optimum near the CE-AM.

Recently, Povey et al. [15] proposed LFMMI, which can avoid the redundant lattice-creation procedure. Instead of lattice-based error calculation, LFMMI uses forward-backward calculation on the phone 4-gram space to calculate Eq. (3). Various techniques (e.g., $l2$-regularization, CE-regularization, connectionist-temporal-classification (CTC)-like topology) were introduced to realize MMI training of AMs without relying on the CE-AM-based lattices. As an important difference from the conventional DNN-HMM, it is assumed in LFMMI modeling that the AM directly outputs the pseudo log-likelihood instead of the posterior probability of each state; i.e., the softmax activation function is no longer applied at the output layer. This requires us an additional consideration about TS training, which we will discuss in Section 3.1.

Although LFMMI achieved a large accuracy gain on the conventional methods, Povey et al. [15] also showed that the sMBR training on LFMMI AM could further improve the accuracy. In our experiments, we basically train an AM first by LFMMI and then switched to sMBR training starting from the LFMMI-trained AM.

3. TS TRAINING WITH SEQUENCE-LEVEL TEMPERATURED KL DIVERGENCE

3.1. Conventional method: $l2$-norm-based training

TS training is a technique to train a small student model that imitates a large and accurate teacher model. For conventional CE-based DNN-HMM AMs, most literature realized the TS training by minimizing the frame-level KL divergence between teacher and student models' outputs [1]. Hinton et al. proposed using temperatured softmax to make the output distribution softer (i.e., more informative), which is known as "knowledge distillation" [2]. However, these techniques assumed that the softmax activation function is used at the output layer. Because the softmax activation function is not used in LFMMI modeling, the KL-based formalization and the temperatured softmax are both no longer able to be applied.

Another simple way to realize TS training is to minimize the $l2$-norm between teacher and student models' outputs. In this case, the training criterion (to maximize) is the inverse of squared $l2$-norm, as follows.

$$\mathcal{F}^{l2}(\theta||\theta^*) = \sum_u \sum_t -\frac{1}{2}||y(u,t) - r(u,t)||_2^2 \quad (4)$$

Here, $r(u,t)$ is the output of the teacher model at the time frame t of utterance u. The error signal w.r.t. the final layer's output of the student model is as follows.

$$\frac{\partial \mathcal{F}^{l2}}{\partial y(u,t)} = r(u,t) - y(u,t) \quad (5)$$

Note that the same error is approximately derived when the temperatured softmax activation is used with a high temperature [2]. In our experiment with LFMMI AMs, we evaluated $l2$-norm based TS training as the conventional method of the frame-level TS training.

3.2. Proposed method: Sequence-level temperatured Kullback-Leibler divergence-based training

We define the training criterion to maximize as the inverse of KL divergence between the sequence-level temperatured posterior on the basis of teacher model parameter θ^* and student model parameter θ.

$$\begin{aligned}\mathcal{F}^{SeqKL}(\theta||\theta^*;T) &= -\sum_u \sum_{\mathbf{S}} P_{T,\theta^*}(\mathbf{S}|\mathbf{X}_u) \log \frac{P_{T,\theta^*}(\mathbf{S}|\mathbf{X}_u)}{P_{T,\theta}(\mathbf{S}|\mathbf{X}_u)} \\ &\propto \sum_u \sum_{\mathbf{S}} P_{T,\theta^*}(\mathbf{S}|\mathbf{X}_u) \log P_{T,\theta}(\mathbf{S}|\mathbf{X}_u) \end{aligned} \quad (6)$$

Here, $P_{T,\theta}(\mathbf{S}|\mathbf{X}_u)$ represents the temperatured posterior of sequence $\mathbf{S}$ given $\mathbf{X}$ with model parameter θ and temperature parameter T as follows.

$$P_{T,\theta}(\mathbf{S}|\mathbf{X}) = \frac{[P_\theta(\mathbf{X}|\mathbf{S})P(\mathbf{S})]^{1/T}}{\sum_{\mathbf{S}'}[P_\theta(\mathbf{X}|\mathbf{S}')P(\mathbf{S}')]^{1/T}}, \quad (7)$$

If $T = 1$, it is equal to the conventional sequence-level posterior. By applying T larger than 1.0, the posterior becomes close to the uniform distribution (i.e., softer distribution). [3]

Then, the error signal w.r.t. the final layer's $y(u,t)$ of the student model can be derived as follows. [4]

$$\begin{aligned}\frac{\partial \mathcal{F}^{SeqKL}}{\partial y(u,t)} &= \sum_{\mathbf{S}} P_{T,\theta^*}(\mathbf{S}|\mathbf{X}_u) \frac{\partial \log P_{T,\theta}(\mathbf{S}|\mathbf{X}_u)}{\partial y(u,t)} \\ &= \sum_{\mathbf{S}} P_{T,\theta^*}(\mathbf{S}|\mathbf{X}_u) \frac{1}{T}(\delta_{\mathbf{S}:y(u,t)} - \gamma^{DEN}_{T,\theta,y(u,t)}) \\ &= \frac{1}{T}(\gamma^{DEN}_{T,\theta^*,y(u,t)} - \gamma^{DEN}_{T,\theta,y(u,t)}) \end{aligned} \quad (8)$$

Here, $\gamma^{DEN}_{T,\theta,y(u,t)} = \sum_{\mathbf{S}} \delta_{\mathbf{S}:y(u,t)} P_{T,\theta}(\mathbf{S}|\mathbf{X}_u)$ is a temperatured posterior probability of being a state corresponding to $y(u,t)$ calculated on the basis of the student parameter θ (i.e., a temperatured version of (3)). Similarly, $\gamma^{DEN}_{T,\theta^*,y(u,t)} = \sum_{\mathbf{S}} \delta_{\mathbf{S}:y(u,t)} P_{T,\theta^*}(\mathbf{S}|\mathbf{X}_u)$ is a temperatured posterior probability of being a state corresponding to $y(u,t)$ calculated on the basis of the teacher's parameter θ^*. By comparing Eqs. (2) and (8), the differences from the normal supervised training are (i) the use of the posterior $\gamma^{DEN}_{T,\theta^*,y(u,t)}$ instead of the delta function and (ii) the use of the temperature parameter. Both $\gamma^{DEN}_{T,\theta,y(u,t)}$ and $\gamma^{DEN}_{T,\theta^*,y(u,t)}$ can be estimated using the forward-backward calculation over the (temperatured) phone 4-gram space, the same as the estimation of $\gamma^{DEN}_{\theta,y(u,t)}$.

4. EXPERIMENT

4.1. Experimental settings

Our experiment was conducted on the individual headset microphone (IHM) data of the AMI meeting corpus [21]. We conducted

[3] The temperature parameter is different from the AM scaling parameter in that it is applied to both AM and language model (LM) scores whereas the AM scaling parameter is applied only to the AM score in order to coordinate the scale of the LM score and the AM score. Note that, different from the conventional DNN-HMM, the LM score and the AM score are combined without any scaling in LFMMI training.

[4] From the first equation to the second equation, almost the same derivation as Eq. (2) is used by replacing $\mathbf{S}_u$ with $\mathbf{S}$. From the second to the third, we used $\sum_{\mathbf{S}} P_{T,\theta^*}(\mathbf{S}|\mathbf{X}_u) = 1$.

our experiments on the basis of the Kaldi toolkit [22]. The training and evaluation datasets were prepared accordance with the instructions in Kaldi. The training data totaled 77 h and was augmented 6 times using speed perturbation (x3) [23] and noise/reverberation perturbation (x2) [24]. The development and evaluation data totaled 8.9 h and 8.7 h, respectively. A 3-gram LM trained by AMI transcription (49K vocabulary) was used for decoding. We tuned all parameters by using the development data, and the best setting was used for decoding the evaluation data.

We trained two types of AMs, the architectures of which are shown in Fig. 1. One is a big and accurate teacher-AM, which consisted of CNN, TDNN, and LSTM. The other is a much smaller student-AM, which was a combination of TDNN and LSTM. The output layer had 4,654 nodes, which corresponded to the clustered context-dependent phoneme HMM states. As explained in Section 2, softmax activation was not used at the output layer. Input features for the network were 40-dim Mel-frequency cepstral coefficients (MFCCs) and 40-dim log-Mel-filterbank (FBANK) both without normalization. In addition, we extracted a 100-dim iVector every 100 msec and appended it to the input features for online speaker/environment normalization [25]. Instead of delaying reference labels, we advanced the input features by five frames, eliminating the need to consider the output delay in model distillation. Both models were first trained by LFMMI and then further trained by sMBR criterion. We applied $l2$-regularization and cross-entropy-regularization proposed by Povey et al. [15] with scales of 0.00005 and 0.1, respectively. Batch normalization [26] was applied after each convolutional layer. [5] In addition, backstitch technique [8] with the backstitch scale 1.0 and backstitch interval 4 was used in LFMMI training. Parameter size and the real time factor (RTF) for decoding (evaluated on Intel(R) Xeon(R) CPU E5-2670) for two AMs are shown in Table 1.

Table 1. *Summary of teacher and student models.*

Architecture	# of params.	RTF
CNN-TDNN-LSTM (teacher)	47.7M	1.30
TDNN-LSTM (student)	10.8M	0.29

4.2. Baseline model results

Word error rates (WERs) of the teacher and student AMs are shown in Table 2. As shown in the table, additional sMBR training after LFMMI training consistently gave us the best results. CNN-TDNN-LSTM achieved much better WERs than TDNN-LSTM. [6] However, CNN-TDNN-LSTM is so large that it cannot be used for real-time decoding (as shown in Table 1). Therefore, in the next experiment, we conducted TS training to make the small TDNN-LSTM mimic the large CNN-TDNN-LSTM.

Table 2. *WERs of teacher and student models.*

Model	Criterion	dev	eval
CNN-TDNN-LSTM	LFMMI	19.32	18.76
	LFMMI → sMBR	**18.83**	**18.05**
TDNN-LSTM	LFMMI	20.82	20.33
	LFMMI → sMBR	**20.24**	**19.72**

[5] Batch normalization in other places slightly degraded the performance in our preliminary experiment.

[6] We actually could not find any better AMI-IHM results than those of CNN-TDNN-LSTM in the previous literature.

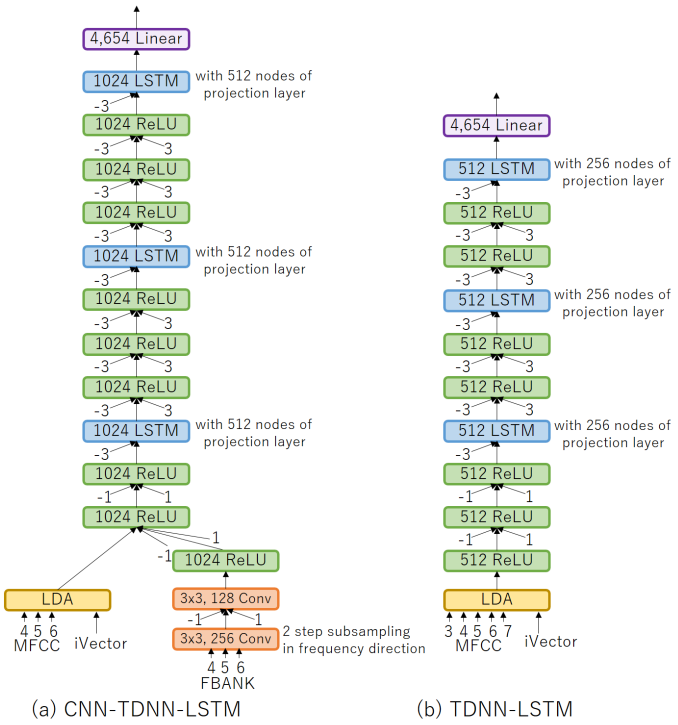

Fig. 1. *Model architectures of (a) teacher and (b) student models. A number with an arrow indicates a time splicing index, which forms a basis of TDNN [27].*

4.3. Comparison of TS training with $l2$-norm and sequence KL

We then evaluated TS training with conventional $l2$-norm based method and sequence KL-based method. In this experiment, we set the temperature parameter $T = 1.0$. We conducted TS training starting from sMBR-trained TDNN-LSTM and conducted four epochs of TS training with various learning rates. Backstitch training with the backstitch scale 1.0 and backstitch interval 4 was used, which slightly improved WERs. Neither $l2$-regularization nor CE-regularization was applied.

Results are shown in Table 3. In this table, initial learning rates are presented. The learning rate was exponentially decayed from the initial value presented in the table to the one-tenth the initial learning rate at the end of the training. As shown in the table, $l2$-norm based TS training improved the WER from 19.72% to 19.01%, which corresponds to filling only 43% of the WER gap between the student and teacher AMs. On the other hand, sequence KL-based TS training achieved a 18.45% WER, filling 76% of the WER gap between the student and teacher AMs.

4.4. Effect of initialization of student model before TS training

As a supplemental experiment, we compared different initialization schemes of the student AM before TS training. Table 4 shows the results of TS training starting from (i) a randomly initialized student model, (ii) a LFMMI-trained student model, and (iii) a sMBR trained student model. The sequence KL-based training with $T = 1.0$ was conducted with the initial learning rate of 0.005. As shown in the table, the better the initial model, the better the results after TS training. In the rest of the experiments, we conducted TS training starting from the sMBR trained student model.

Table 3. *WER of student AM (TDNN-LSTM) with different TS training criterion.*

Criterion	Learning Rate	dev	eval
LFMMI → sMBR	-	**20.24**	**19.72**
TS ($l2$-norm)	0.0001	20.14	19.78
	0.00005	19.70	19.14
	0.00002	**19.65**	**19.01**
	0.00001	19.72	19.13
	0.000005	20.00	19.46
TS (SeqKL; $T = 1.0$)	0.02	19.99	19.37
	0.01	19.44	18.72
	0.005	**19.08**	**18.45**
	0.002	19.15	18.54
	0.001	19.41	18.84
Teacher Model (CNN-TDNN-LSTM)		**18.83**	**18.05**

Table 4. *Effect of initialization of student model for TS training.*

Initial Student Model	WER before TS		WER after TS	
	dev	eval	dev	eval
Random	-	-	19.20	18.79
LFMMI	20.82	20.33	19.14	18.55
LFMMI → sMBR	20.24	19.72	**19.08**	**18.45**

4.5. Effect of temperature in sequence KL-based TS training

We then evaluated the effect of the temperature parameter T in the proposed sequence KL-based TS training. WERs with different temperature values for development and evaluation sets are shown in Fig. 2. As shown in the figure, a temperature of 1.2-1.4 slightly but consistently improved WERs. The best result was achieved at the temperature of 1.2, where we achieved a 18.23% WER for the evaluation set. The difference from the teacher model was only 0.18% of WER (filling 89% of the WER gap between teacher and student models) although the student model was 4.4 times smaller and can be decoded 4.5 times faster.

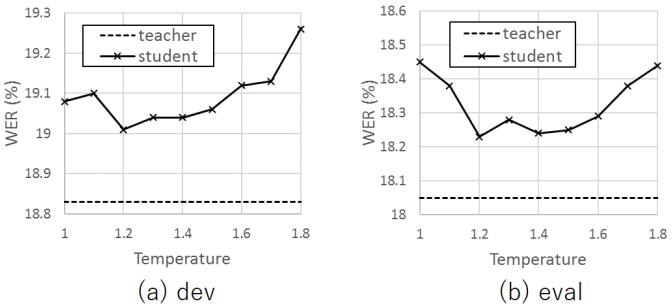

Fig. 2. *Effect of temperature in TS training based on the sequence-KL divergence.*

4.6. Effect of student model size

Finally, we evaluated the case where an even smaller student model was used for TS training. In this experiment, we used the same teacher model (CNN-TDNN-LSTM) but shrank the student TDNN-LSTM by reducing the node size. We prepared two types of additional student TDNN-LSTMs. One had the same architecture as that in Fig. 1 (b), but all 512-dim ReLU layers were replaced with 256-dim ReLu layers. This model had 6.4M parameters, and decoding RTF was 0.17. The other is the same as the first, but the parameters were further reduced by replacing all 512-dim LSTM layers into 256-dim LSTM with 128-dim projection layers. The latter model had 3.3M parameters, and decoding RTF was 0.10.

WERs for evaluation data with three student models are shown in Fig. 3. The left figure plots the results in accordance with the model size, and the right figure plots the same results in accordance with the decoding RTF. All training parameters were set to the best parameters in the previous experiments. As shown in the figure, the proposed sequence-KL-based TS training consistently achieved much better WERs than the conventional $l2$-norm based TS training.

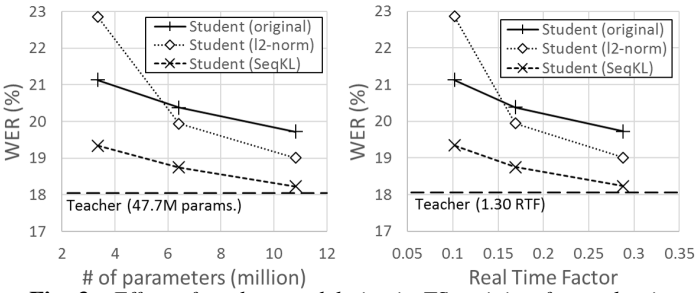

Fig. 3. *Effect of student model size in TS training for evaluation data (left: params., right: RTF). "Student (original)" indicates the student model before TS training.*

One important observation here is that the smaller the student model, the larger the difference between $l2$-norm based TS training and the proposed sequence-KL-based TS training. Especially, the $l2$-norm based TS training even degraded the WER from the original student model when the model was very small. It should be emphasized that this degradation was not caused by the parameter divergence in network training because we found that the training criterion $\mathcal{F}^{l2}$ was successfully improved even for the smallest model. Instead, we interpret that this phenomena occurs because the incomplete imitation of frame-level output (due to too small capacity of the model) deteriorated the sequence-level quality as the sequence-trained AMs. Different from the $l2$-norm based method, our proposed method consistently achieved improvements even for very small AMs.

5. CONCLUSION

In this paper, we proposed using the sequence-level temperatured Kullback-Leibler divergence as a metric for TS training. In our experiment on the AMI meeting corpus, the proposed method filled 89% of the WER gap between teacher and student AMs, whereas the frame-level TS training was able to fill only 43% of the WER gap. We also showed that the smaller the student model, the more superior the proposed method. The frame-level TS training sometimes even degraded the performance, whereas the proposed method consistently improved the accuracy.

6. REFERENCES

[1] Jinyu Li, Rui Zhao, Jui-Ting Huang, and Yifan Gong, "Learning small-size DNN with output-distribution-based criteria," in *Proc. INTERSPEECH*, 2014, pp. 1910–1914.

[2] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean, "Distilling the knowledge in a neural network," *arXiv preprint arXiv:1503.02531*, 2015.

[3] George Saon, Gakuto Kurata, Tom Sercu, Kartik Audhkhasi, Samuel Thomas, Dimitrios Dimitriadis, Xiaodong Cui, Bhuvana Ramabhadran, Michael Picheny, Lynn-Li Lim, et al., "English conversational telephone speech recognition by humans and machines," *arXiv preprint arXiv:1703.02136*, 2017.

[4] Sepp Hochreiter and Jürgen Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[5] Tara N Sainath, Oriol Vinyals, Andrew Senior, and Hasim Sak, "Convolutional, long short-term memory, fully connected deep neural networks," in *Proc. ICASSP*, 2015, pp. 4580–4584.

[6] Alexander Waibel, Toshiyuki Hanazawa, Geoffrey Hinton, Kiyohiro Shikano, and Kevin J Lang, "Phoneme recognition using time-delay neural networks," *IEEE Trans. ASSP*, vol. 37, no. 3, pp. 328–339, 1989.

[7] Vijayaditya Peddinti, Yiming Wang, Daniel Povey, and Sanjeev Khudanpur, "Low latency modeling of temporal contexts," *IEEE Signal Processing Letters*, 2017.

[8] Yiming Wang, Vijayaditya Peddinti, Hainan Xu, Xiaohui Zhang, Daniel Povey, and Sanjeev Khudanpur, "Backstitch: Counteracting finite-sample bias via negative steps," *Proc. INTERSPEECH*, pp. 1631–1635, 2017.

[9] William Chan, Nan Rosemary Ke, and Ian Lane, "Transferring knowledge from a RNN to a DNN," *Proc. INTERSPEECH*, pp. 3264–3268, 2015.

[10] Yevgen Chebotar and Austin Waters, "Distilling knowledge from ensembles of neural networks for speech recognition.," in *Proc. INTERSPEECH*, 2016, pp. 3439–3443.

[11] Liang Lu, Michelle Guo, and Steve Renals, "Knowledge distillation for small-footprint highway networks," in *Proc. ICASSP*, 2017, pp. 4820–4824.

[12] Shinji Watanabe, Takaaki Hori, Jonathan Le Roux, and John R Hershey, "Student-teacher network learning with enhanced features," in *Proc. ICASSP*, 2017, pp. 5275–5279.

[13] Hang Su, Gang Li, Dong Yu, and Frank Seide, "Error back propagation for sequence training of context-dependent deep networks for conversational speech transcription.," in *Proc. ICASSP*, 2013, pp. 6664–6668.

[14] Karel Veselỳ, Arnab Ghoshal, Lukás Burget, and Daniel Povey, "Sequence-discriminative training of deep neural networks.," in *Proc. INTERSPEECH*, 2013, pp. 2345–2349.

[15] Daniel Povey, Vijayaditya Peddinti, Daniel Galvez, Pegah Ghahrmani, Vimal Manohar, Xingyu Na, Yiming Wang, and Sanjeev Khudanpur, "Purely sequence-trained neural networks for asr based on lattice-free mmi," *Proc. INTERSPEECH*, pp. 2751–2755, 2016.

[16] Yoon Kim and Alexander M Rush, "Sequence-level knowledge distillation," *arXiv preprint arXiv:1606.07947*, 2016.

[17] Jeremy HM Wong and Mark JF Gales, "Sequence student-teacher training of deep neural networks," in *Proc. INTERSPEECH*, 2016, pp. 2761–2765.

[18] Naoyuki Kanda, Yusuke Fujita, and Kenji Nagamatsu, "Investigation of lattice-free maximum mutual information-based acoustic models with sequence-level Kullback-Leibler divergence," *Proc. ASRU*, pp. 69–76, 2017.

[19] Wayne Xiong, Jasha Droppo, Xuedong Huang, Frank Seide, Mike Seltzer, Andreas Stolcke, Dong Yu, and Geoffrey Zweig, "Achieving human parity in conversational speech recognition," *arXiv preprint arXiv:1610.05256*, 2016.

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

[21] Jean Carletta, Simone Ashby, Sebastien Bourban, Mike Flynn, Mael Guillemot, Thomas Hain, Jaroslav Kadlec, Vasilis Karaiskos, Wessel Kraaij, Melissa Kronenthal, et al., "The ami meeting corpus: A pre-announcement," in *International Workshop on Machine Learning for Multimodal Interaction*. Springer, 2005, pp. 28–39.

[22] Daniel Povey, Arnab Ghoshal, Gilles Boulianne, Lukáš Burget, Ondřej Glembek, Nagendra Goel, Mirko Hannemann, Petr Motlíček, Yanmin Qian, Petr Schwarz, et al., "The Kaldi speech recognition toolkit," in *Proc. ASRU*, 2011.

[23] Tom Ko, Vijayaditya Peddinti, Daniel Povey, and Sanjeev Khudanpur, "Audio augmentation for speech recognition.," in *Proc. INTERSPEECH*, 2015, pp. 3586–3589.

[24] Tom Ko, Vijayaditya Peddinti, Daniel Povey, Michael L Seltzer, and Sanjeev Khudanpur, "A study on data augmentation of reverberant speech for robust speech recognition," in *Proc. ICASSP*, 2017, pp. 5220–5224.

[25] George Saon, Hagen Soltau, David Nahamoo, and Michael Picheny, "Speaker adaptation of neural network acoustic models using i-vectors.," in *Proc. ASRU*, 2013, pp. 55–59.

[26] Sergey Ioffe and Christian Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. ICML*, 2015, pp. 448–456.

[27] Vijayaditya Peddinti, Daniel Povey, and Sanjeev Khudanpur, "A time delay neural network architecture for efficient modeling of long temporal contexts.," in *Proc. INTERSPEECH*, 2015, pp. 3214–3218.