Lattice Based Transcription Loss for End-to-End Speech Recognition

Jian Kang[1], *Wei-Qiang Zhang*[1] *and Jia Liu*[1]

[1]Tsinghua National Laboratory for Information Science and Technology
Department of Electronic Engineering, Tsinghua University, Beijing 100084, China

`kangj13@mails.tsinghua.edu.cn`

Abstract

End-to-end speech recognition systems have been successfully implemented and have become competitive replacements for hybrid systems. A common loss function to train end-to-end systems is connectionist temporal classification (CTC). This method maximizes the log likelihood between the feature sequence and the associated transcription sequence. However there are some weaknesses with CTC training. The main weakness is that the training criterion is different from the test criterion, since the training criterion is log likelihood, while the test criterion is word error rate. In this work, we introduce a new lattice based transcription loss function to address this deficiency of CTC training. Compared to the CTC function, our new method optimizes the model directly using the transcription loss. We evaluate this new algorithm in both a small speech recognition task, the Wall Street Journal (WSJ) dataset and a large vocabulary speech recognition task, the Switchboard dataset. Results demonstrate that our algorithm outperforms a traditional CTC criterion, and achieving 7% WER relative reduction. In addition, we compare our new algorithm to some discriminative training algorithms, such as state-level minimum Bayes risk (SMBR) and minimum word error (MWE), showing that our algorithm is more convenient and contains more varieties for speech recognition.

Index Terms: lattice, transcription loss, end-to-end system, connectionist temporal classification

1. Introduction

In recent years, deep neural networks (DNNs) combined with hidden Markov models (HMM) have become the dominant approach in acoustic modeling [1, 2, 3]. In these hybrid systems, DNNs compute the emission probabilities of HMM states. Recurrent neural networks (RNNs) add to this by incorporating self connections from the previous time step. The hidden state contains a dynamic history information instead of a fixed-size window on the input features sequence. This architecture has performed better than traditional DNNs , and the use of temporal connections has made the use of RNNs as well as Long Short-Term Memory RNNs (LSTM RNNs) [4] more suitable for sequence tasks [5, 6, 7, 8, 9, 10]. These approaches as well as increased computation power and amounts of training data have resulted in substantial reduction of error rate for speech recognition tasks.

However, traditional neural networks based hybrid systems are also restrict the application of speech recognition systems. They require extra resources such as dictionaries, which may be impractical in real applications. In addition, they rely on Gaussian Mixture Models (GMM) to obtain initial frame-level labels. The procedure of building GMM system involves a number of different training complexities which make it difficult to implement. In recent years, another variant of systems called end-to-end system [11, 12] has been proposed. Such systems build the relationship between acoustic features sequence and their transcription sequence directly, without any requirement of intermediate components. In [13, 14, 15, 16, 17, 18], the end-to-end systems are composed of layers of bidirectional Long Short-term Memory (LSTM) network which are trained by connectionist temporal classification (CTC). With CTC, deep bidirectional LSTMs can be trained directly by using the log likelihood between input feature sequence and their transcript sequence. Such systems have achieved better results than comparative hybrid systems on datasets having thousands of hours English training data.

Although the CTC based end-to-end systems have achieved strong performance, the current CTC systems have several weaknesses. With the CTC function, incorrect transcriptions are ignored, which implies all CTC paths are equally bad. In addition, the CTC criterion function is not consistent with the test criterion. The CTC training function maximizes the log likelihood between input and output sequences. However, the test criterion for speech recognition tasks is given by the word error rate. We would like to use a more direct criterion to train the model, so that the gap between the train and test criterion can be reduced.

In this paper, we propose a new lattice based transcription loss function. This function corresponds better to the target test results. We evaluate our algorithm on both a small scaled Wall Street Journal (WSJ) task and a large vocabulary continuous speech recognition Switchboard task. From our results, we can conclude that our lattice based transcription loss function obtains about 7% relative error reduction on SWB task and 6% relative error reduction on WSJ task. In addition, we compare and analyze the differences between our algorithm and some early works focus on task loss estimation [19, 20] as well as discriminative sequence training algorithms [21, 22, 23, 24].

The remaining parts of this paper are organized as follows: In Section 2, we briefly review the CTC function. In Section 3, we propose our lattice based transcription loss function and compare it to some early works in [20, 21, 22, 23, 24]. We report our experimental results in detail in Section 4, with conclusions in Section 5.

2. Connectionist Temporal Classification

Traditional neural network based acoustic models are typically trained using frame-level labels and a cross entropy criterion. The frame-level labels are often generated by an early GMM-HMM system. In contrast to this, connectionist temporal

This work is supported by National Natural Science Foundation of China under Grant No. 61370034, No. 61273268 and No. 61403224.

classification (CTC) [25] maps the input frames to the associated transcription sequence directly, maximizing the log probability of the likelihood between all possible alignments and a target transcription. No prior knowledge about the relationship between the input and output sequence is required, nor is any initial auxiliary system.

We denote K as the number of units, and add an extra symbol ϕ as the blank unit, corresponding to the null emissions. Assume $\mathbf{x} = (x_{1:T})$ is the input sequence and $\mathbf{z} = (\boldsymbol{z}_{1:U})$ is its output transcription sequence. CTC aims to maximize $lnPr(\mathbf{z}|\mathbf{x})$, the log probability of the output text sequence given its inputs.

Given these two sequences, it is difficult to calculate the above probability directly, since the length of output sequence $\mathbf{z}$ and input sequence $\mathbf{x}$ are not normally equal. To solve this difficulty, a CTC path $\pi = (\boldsymbol{\pi}_{1:T})$ is proposed, representing a frame level label sequence. Compared to $\mathbf{z}$, this allows repetitions of realistic units and occurrences of blank unit. The additional blank unit means that no labels are emitted. Since the repetition operations are arbitrary, there are multiple CTC paths corresponding to the same transcription sequence. We define the CTC paths set of a transcription sequence $\mathbf{z}$ as $B^{-1}(\mathbf{z})$. The likelihood of this sequence $\mathbf{z}$ can be calculated as the sum of probability of all its CTC paths.

$$Pr(\mathbf{z}|\mathbf{x}) = \sum_{\pi \in B^{-1}(\mathbf{z})} Pr(\pi|\mathbf{x}). \tag{1}$$

Each likelihood $Pr(\pi|\mathbf{x})$ is decomposed using the output posterior probability of the softmax layer.

$$Pr(\pi|\mathbf{x}) = \prod_{i=1}^{T} y^t_{\pi_t}. \tag{2}$$

where, y^t_k is the posterior probability of the label k at the t frame.

After this process, the log likelihood can theoretical be calculated. However, traversing all possible paths in Eq. (1) is computationally intractable. In practice, Eq. (1) can be efficiently calculated using a dynamic programming algorithm. We define α^t_u as the total probability of all CTC paths ending with $\boldsymbol{z}_u$ at the t frame. Similarly, we define β^t_u as the total probability of all CTC paths starting with $\boldsymbol{z}_u$ at the t frame. By using these two variants, the likelihood probability can be computed as:

$$Pr(\mathbf{z}|\mathbf{x}) = \sum_u \frac{\alpha^t_u \beta^t_u}{y^t_{\boldsymbol{z}_u}}. \tag{3}$$

The derivative of equation (3) with respect to the softmax layer output y^t_k can be calculated as:

$$\frac{\partial lnPr(\mathbf{z}|\mathbf{x})}{\partial y^t_k} = -\frac{1}{Pr(\mathbf{z}|\mathbf{x})(y^t_k)^2} \sum_{u \in \{u|\boldsymbol{z}_u = k\}} \alpha^t_u \beta^t_u. \tag{4}$$

3. Lattice Based Transcription Loss

The CTC function maximizes the log probability of the output correct transcription sequence given its inputs. Although this function considers the sequence information compared to the traditional cross entropy criterion, there are still some deficiencies to be considered. First, during optimization, the incorrect transcriptions are ignored, which implies all CTC paths are equal. We hope that candidates with higher correctness are more probable than those with lower. Second, the CTC function does not directly correspond with the test criterion. The CTC function uses the derivate of the log likelihood to optimize the parameters of the neural network. However, in speech recognition task, the most usual and standard measure is the word error rate, which is defined as the edit distance between the candidate transcription sequences and the correct transcription sequence. In this view, both the cross entropy criterion and the CTC function are surrogate loss functions. We would like to use a more direct criterion to train the model, so the gap between the train and test criterion can be reduced. There have been some previous works on this, such as [19, 20]. In this section, we will follow these thoughts and propose a new lattice based transcription loss function and an associated optimization procedure.

We denote the input acoustic features sequence, the candidate transcription sequences and the correct transcription sequence as $\mathbf{x}$, $\mathbf{y}$, $\mathbf{z}$, respectively. Given these sequences, the transcription loss $\mathbf{L}(x)$ is defined as:

$$\mathbf{L}(x) = \sum_y Pr(\mathbf{y}|\mathbf{x})\mathfrak{L}(\mathbf{y}, \mathbf{z}). \tag{5}$$

Here, $\mathfrak{L}(\mathbf{y}, \mathbf{z})$ calculates the edit distance between sequence $\mathbf{y}$ and $\mathbf{z}$.

Then, substituting Eq. (1) into Eq. (5) :

$$\mathbf{L}(x) = \sum_y \sum_{\pi \in B^{-1}(\mathbf{y})} Pr(\pi|\mathbf{x})\mathfrak{L}(\mathbf{y}, \mathbf{z}) \tag{6}$$

$$= \sum_\pi Pr(\pi|\mathbf{x})\mathfrak{L}(B(\pi), \mathbf{z}) \tag{7}$$

In [20], they use Monte-Carlo sampling to approximate both $\mathbf{L}(x)$ and their gradients. However, we decompose above equation further, adopting lattice states as the basic variants of the function.

A lattice consists of a set of nodes representing points in time and a set of spanning arcs representing unit hypotheses. A simple lattice example is shown in Fig.1. Each node and its spanning arc comprises a state. By combining states serially, we can generate different transcription candidates, i.e. different CTC paths. We define two property functions for a state in the lattice. For each state s, we define $T(s)$ as the time index of state s, and define $C(s)$ as the unit class index of state s. For instance, as Fig.1 shows, $T(S6) = t3$ and $C(S6) = c$.

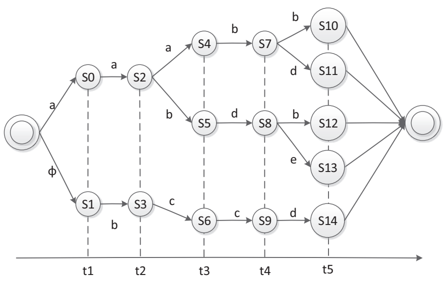

Figure 1: A simple lattice example.

For each state s, we can use a dynamic programming algorithm to access the best forward sub-sequence $\alpha(s)$ and backward sub-sequence $\beta(s)$. Here, the forward sub-sequence and backward sub-sequence for state s represent the output of the arc sequence ending up with the previous state for s

and starting from s, respectively. For instance, as the Fig.1 shows, $\alpha(S6) = \phi b$ and $\beta(S6) = cd$. A CTC path can be generated by connecting $\alpha(s)$, a unit k and $\beta(s)$. We define J_k as mapping the state s to unit k in a CTC path π. Here, because state s can be mapped to every unit, if k equals $C(s)$, the transcription sequence is a major transcription for state s, otherwise the sequence is a minor transcription for state s. For comparison, the mapping from state to unit in discriminative learning algorithm [21, 22, 23, 24] is fixed, and then observing whether the mapped unit and their corresponding alignment are same or not.

We substitute the above relationship into above Eq. (7) :

$$\mathbf{L}(x) = \sum_{\pi}\sum_{k}\sum_{s\in J_k^{-1}(\pi)} Pr(\pi|\mathbf{x})\mathfrak{L}(B(J_k(s)),\mathbf{z}) \quad (8)$$

$$= \sum_{s}\sum_{k} Pr(s|\mathbf{x})\mathfrak{L}(B(J_k(s)),\mathbf{z}) \quad (9)$$

To differentiate the loss function $\mathbf{L}(x)$ with respect to the softmax layer output $Pr(k,t|\mathbf{x})$, we first use chain rule to decompose the derivative:

$$\frac{\partial \mathbf{L}(x)}{\partial Pr(k,t|\mathbf{x})} = \sum_{s} \frac{\partial \mathbf{L}(x)}{\partial s}\frac{\partial s}{\partial Pr(k,t|\mathbf{x})}. \quad (10)$$

For the former terms, this can be calculated as:

$$\frac{\partial \mathbf{L}(x)}{\partial s} = \sum_{k} Pr(s|\mathbf{x})\mathfrak{L}(B(J_k(s)),\mathbf{z}). \quad (11)$$

For the latter terms, the gradient is :

$$\frac{\partial s}{\partial Pr(k,t|\mathbf{x})} = I(T(s) == t)I(C(s) == k). \quad (12)$$

Here, $I(cond)$ is an indicator function, whose value is equal to 1 if the input variant $cond$ is true, else equal to 0.

Combining these two parts, the Eq. (10) can be rewritten as:

$$\frac{\partial \mathbf{L}(x)}{\partial Pr(k,t|\mathbf{x})} = \sum_{s} Pr(s|\mathbf{x})\mathfrak{L}(B(J_k(s)),\mathbf{z})I(T(s) == t) \\ I(C(s) == k). \quad (13)$$

Because the state s in the lattice is certain, we can set all states having the same probability, so the probability $Pr(s|\mathbf{x})$ equals 1. In the future, this probability can be set unequal with each other according to their confidence.

Next, the derivatives passed through the softmax layer can be computed as:

$$\frac{\partial \mathbf{L}(x)}{\partial y_k^t} = \sum_{k'} \frac{\partial \mathbf{L}(x)}{\partial Pr(k',t|\mathbf{x})}\frac{\partial Pr(k',t|\mathbf{x})}{\partial y_k^t} \quad (14)$$

$$= Pr(k,t|\mathbf{x})\left(\frac{\partial \mathbf{L}(x)}{\partial Pr(k,t|\mathbf{x})} - \sum_{k'} Pr(k',t|\mathbf{x})\frac{\partial \mathbf{L}(x)}{\partial Pr(k',t|\mathbf{x})}\right) \quad (15)$$

As we can see, in our algorithm, different candidate transcriptions have different loss. In addition, the loss function is directly related to the word error rate. These two improvements address shortcomings of the CTC training approach.

Compared to [20], our algorithm adopts lattice based methods instead of Monte-Carlo sampling to obtain candidate transcriptions. This makes the algorithm more suitable for speech recognition. Furthermore, the lattice contains more candidate transcription, and they are stored in a compact way. So the estimate is more precise and the computational complexity is less.

Compared to [21, 22, 23, 24], our algorithm calculate the loss directly using the transcription, however, traditional sequence training algorithms have to generate a fixed frame level alignments, i.e. numerator alignments. So our algorithm is more convenient. Furthermore, the mapping between state and unit in sequence training algorithms is fixed, defining by an HMM topology in hybrid systems [23, 24] or just the central phone of state in CTC system [21, 22]. But for our algorithm, this mapping, i.e. the function $J_k(s)$ in Eq. (9), is arbitrary. This makes the algorithm containing more variant candidates for speech recognition.

4. Experimental results

4.1. Experiment setup

In order to evaluate our model, we implement experiments on Wall Street Journal (WSJ) dataset and switchboard dataset. For WSJ task, there are 81 hours speech in total. We randomly select 95% as the training data, and the remaining 5% as development data. The input features are 40-dim fBank features with zero mean and unit variance as well as second order delta and delta-delta coefficients derivatives. In order to compare with previous works [14, 17, 19, 20], we use $eval02$ as the evaluation data. For SWBD task, we select both the 110-hour subset and the full 310-hour set as the training data and another 3-hour subset as the validation data. We report our results on the NIST 2000 Hub5. For fair comparison, we construct CTC baseline system using EESEN [17], an open-source tool based on Kaldi [26].

Our new lattice based system is initialized by a trained CTC system. The baseline CTC systems for the WSJ task have 4 bidirectional LSTM layers. The baseline CTC systems for the 110-hour SWBD task and the full 310-hour SWBD task have 4 and 5 bidirectional LSTM layers, respectively. At each layer, there are 320 memory cells for both forward and backward layers. For the baseline CTC systems, the learning rate is 0.00004 initially. At the end of every epoch, the learning rate is reduced by a factor of 2 if the frame accuracy on the development set drops. After the CTC training has finished, we decode the training set, generating the lattices and calculating the derivative using the above algorithm. Next, the new lattice based system is initialized with the previous best CTC system and the starting learning rate is 0.000004. After every epoch, we will evaluate on the test set, and if the WER increase, the learning rate is reduced by 2.

4.2. Experimental results for WSJ task

In these experiments, we evaluate our algorithm on the WSJ task. As mentioned in [17], the CTC based system can be used to model phoneme or character. In order to confirm our algorithm, we apply our lattice based algorithm to model both phoneme and character. For the phoneme-based system, we adopt the CMU dictionary as the lexicon. We select 72 labels from the lexicon. In addition, we add noise, short pause marks as well as a blank. The total number of units for the phoneme-based system is 75. For the character-based system, we also extract labels from the word list of the CMU dictionary. The total number of units is 59, including letters, digits, punctuation marks and oov marks. As our new algorithm based systems are initialized by the best CTC system, the architecture of our modified system is the same as that for the CTC systems mentioned above.

In order to compare to the discriminative learning algorithms, we apply sequence training algorithm to phoneme based CTC systems. The details about discriminative sequence training algorithms for CTC systems are in [21, 22].

Results are shown in Table 1. In order to have a broad comparison, we also show the results reported in [14, 17, 19, 20].

Table 1: WER results across systems for WSJ task. Here, 'Lattice based' represents our new algorithm, 'SMBR' represents sequence training algorithm, 'phn' represents systems for phoneme and 'char' represents systems for character.

Model	WER%	System unit
Lattice based	**6.85**	phn
Eesen[17]	7.28	phn
SMBR	7.06	phn
Lattice based	**6.37**	char
Eesen[17]	6.70	char
Graves *et al.*[20]	8.20	char
Hannun *et al.*[14]	14.1	char
Bahdanau *et al.*[19]	18.0	char

Here, the task loss function algorithm [19] and sampling based expected transcription loss algorithm [20] are similar to our algorithm, all of which have the goal of finding a more suitable training function to reduce the gap between the training criterion and testing criterion. From the results, we can see that our lattice based transcription loss function systems can achieve better WER than result achieved by [19, 20] as well as [14] when we model the character. Compared to baseline system Eesen, we can achieve about 6% relative WER reduction in both phoneme and character modeling task. In addition, we observe that our new algorithm can outperform sequence training algorithm, achieving 3% relative WER.

4.3. Experimental results for SWBD task

In these experiments, we evaluate our algorithm on the SWBD task. We test our algorithm to model both phonemes and characters. For the phoneme-based system, we select 45 labels from the lexicon including noise, short pause, laugh marks as well as blank. For the character-based system, the total number of units is 46. The architecture of our modified system is as mentioned above. The set for sequence training algorithm is the same as above.

Results are shown in Table 2. Compared to the baseline Eesen system, we achieve about 7% relative WER reduction in both phoneme and character modeling tasks. Comparing to sequence training algorithm, we achieve about 4% relative WER reduction.

4.4. Experimental results for varying parameters in the new algorithm

In this section, we evaluate the impact of hyperparameters in our algorithm. The most valuable factor in our algorithm is size

Table 2: WER results across systems for SWBD task. Here, '110h' represents systems for 110-hour subset task, 'full' represents systems for full set task, 'phn' represents systems for phoneme, and 'char' represents systems for character.

Model	WER%	Training data duration	System unit
Lattice based	**18.5**	110h	phn
Eesen[27]	19.9	110h	phn
Hybrid LSTM[28]	19.2	110h	phn
SMBR	19.1	110h	phn
Lattice based	**22.8**	110h	char
Eesen	24.4	110h	char
Lattice based	**14.1**	full	phn
Eesen[27]	15.0	full	phn
Hybrid LSTM[28]	15.8	full	phn

of the lattice. In order to describe this property, we calculate the average number of states per frames to measure the size of the lattice. When decoding, we select different beam size to determine the lattice size. We choose four beam size conditions, decoding and calculating the average number of states per frames. For these comparisons, we choose phoneme based systems for WSJ and 110-hour SWBD subset as the experiment environments and measure the WER.

Table 3: WER results across systems with different lattice sizes. Here, '110h-SWBD' represents systems for the 110-hour subset task, and 'WSJ' represents systems for the WSJ task.

Model	Avg #states per frames			
	1.6	3.3	4.5	6
110h-SWBD	19.4	19.0	18.7	**18.5**
WSJ	7.10	6.99	6.92	**6.85**

Results are shown in Table 3. From the results, we can clearly see that when the size of the lattice becomes larger, the WER declines at first and then becomes stable. However, the time cost increases almost linearly with respect to the total number of states in the lattice. So we can select different beam size in practice, which balancing between system performance and calculation time.

5. Conclusions

In this paper, we apply a new lattice based transcription loss function to train end-to-end systems. Compared to the traditional CTC function, this approach reduces the gap between the training criterion and test criterion, giving extra attention to incorrect transcriptions. In addition, the procedures are less complex compared to prior works. Our new algorithm replaces the Monte-Carlo sampling method and is closer to the traditional method in speech recognition. We evaluate this new algorithm on the WSJ and SWBD datasets. Results show that our new algorithm obtains about more than 6% relative error reduction. In addition, our new algorithm achieve about 3% relative WER reduction, compared to system based on SMBR criterion.

6. References

[1] F. Seide, G. Li, and D. Yu, “Conversational speech transcription using context-dependent deep neural networks.” in *Interspeech*, 2011, pp. 437–440.

[2] F. Seide, G. Li, X. Chen, and D. Yu, “Feature engineering in context-dependent deep neural networks for conversational speech transcription,” in *Automatic Speech Recognition and Understanding (ASRU), 2011 IEEE Workshop on*. IEEE, 2011, pp. 24–29.

[3] 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.

[4] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[5] T. Mikolov, M. Karafiát, L. Burget, J. Cernockỳ, and S. Khudanpur, “Recurrent neural network based language model.” in *Interspeech*, vol. 2, 2010, p. 3.

[6] A. Graves, N. Jaitly, and A.-r. Mohamed, “Hybrid speech recognition with deep bidirectional lstm,” in *Automatic Speech Recognition and Understanding (ASRU), 2013 IEEE Workshop on*. IEEE, 2013, pp. 273–278.

[7] H. Sak, A. W. Senior, and F. Beaufays, “Long short-term memory recurrent neural network architectures for large scale acoustic modeling.” in *INTERSPEECH*, 2014, pp. 338–342.

[8] T. N. Sainath, O. Vinyals, A. Senior, and H. Sak, “Convolutional, long short-term memory, fully connected deep neural networks,” in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2015, pp. 4580–4584.

[9] J. T. Geiger, Z. Zhang, F. Weninger, B. Schuller, and G. Rigoll, “Robust speech recognition using long short-term memory recurrent neural networks for hybrid acoustic modelling.” in *INTERSPEECH*, 2014, pp. 631–635.

[10] A. Senior, H. Sak, and I. Shafran, “Context dependent phone models for lstm rnn acoustic modelling,” in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2015, pp. 4585–4589.

[11] A. Graves, A.-r. Mohamed, and G. Hinton, “Speech recognition with deep recurrent neural networks,” in *2013 IEEE international conference on acoustics, speech and signal processing*. IEEE, 2013, pp. 6645–6649.

[12] J. Chorowski, D. Bahdanau, K. Cho, and Y. Bengio, “End-to-end continuous speech recognition using attention-based recurrent nn: first results,” *arXiv preprint arXiv:1412.1602*, 2014.

[13] H. Sak, A. Senior, K. Rao, O. Irsoy, A. Graves, F. Beaufays, and J. Schalkwyk, “Learning acoustic frame labeling for speech recognition with recurrent neural networks,” in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2015, pp. 4280–4284.

[14] A. Y. Hannun, A. L. Maas, D. Jurafsky, and A. Y. Ng, “First-pass large vocabulary continuous speech recognition using bi-directional recurrent dnns,” *arXiv preprint arXiv:1408.2873*, 2014.

[15] A. Hannun, C. Case, J. Casper, B. Catanzaro, G. Diamos, E. Elsen, R. Prenger, S. Satheesh, S. Sengupta, A. Coates *et al.*, “Deep speech: Scaling up end-to-end speech recognition,” *arXiv preprint arXiv:1412.5567*, 2014.

[16] D. Amodei, R. Anubhai, E. Battenberg, C. Case, J. Casper, B. Catanzaro, J. Chen, M. Chrzanowski, A. Coates, G. Diamos *et al.*, “Deep speech 2: End-to-end speech recognition in english and mandarin,” *arXiv preprint arXiv:1512.02595*, 2015.

[17] Y. Miao, M. Gowayyed, and F. Metze, “Eesen: End-to-end speech recognition using deep rnn models and wfst-based decoding,” in *2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*. IEEE, 2015, pp. 167–174.

[18] J. Li, H. Zhang, X. Cai, and B. Xu, “Towards end-to-end speech recognition for chinese mandarin using long short-term memory recurrent neural networks,” in *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.

[19] D. Bahdanau, D. Serdyuk, P. Brakel, N. R. Ke, J. Chorowski, A. Courville, and Y. Bengio, “Task loss estimation for sequence prediction,” *arXiv preprint arXiv:1511.06456*, 2015.

[20] A. Graves and N. Jaitly, “Towards end-to-end speech recognition with recurrent neural networks.” in *ICML*, vol. 14, 2014, pp. 1764–1772.

[21] H. Sak, A. Senior, K. Rao, and F. Beaufays, “Fast and accurate recurrent neural network acoustic models for speech recognition,” *arXiv preprint arXiv:1507.06947*, 2015.

[22] T. Sainath, K. Rao *et al.*, “Acoustic modelling with cd-ctc-smbr lstm rnns,” in *2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*. IEEE, 2015, pp. 604–609.

[23] B. Kingsbury, “Lattice-based optimization of sequence classification criteria for neural-network acoustic modeling,” in *2009 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2009, pp. 3761–3764.

[24] D. Povey and P. C. Woodland, “Minimum phone error and i-smoothing for improved discriminative training,” in *Acoustics, Speech, and Signal Processing (ICASSP), 2002 IEEE International Conference on*, vol. 1. IEEE, 2002, pp. I–105.

[25] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, “Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks,” in *Proceedings of the 23rd international conference on Machine learning*. ACM, 2006, pp. 369–376.

[26] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz *et al.*, “The kaldi speech recognition toolkit,” in *IEEE 2011 workshop on automatic speech recognition and understanding*, no. EPFL-CONF-192584. IEEE Signal Processing Society, 2011.

[27] Y. Miao, M. Gowayyed, X. Na, T. Ko, F. Metze, and A. Waibel, “An empirical exploration of ctc acoustic models,” in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2016, pp. 2623–2627.

[28] Y. Miao and F. Metze, “On speaker adaptation of long short-term memory recurrent neural networks,” in *Sixteenth Annual Conference of the International Speech Communication Association (INTERSPEECH). ISCA*, 2015.