# Towards End-to-End Speech Recognition for Chinese Mandarin using Long Short-Term Memory Recurrent Neural Networks

*Jie Li, Heng Zhang, Xinyuan Cai, Bo Xu*

Interactive Digital Media Technology Research Center,
Institute of Automation, Chinese Academy of Sciences, Beijing, P.R.China
{jie.li, heng.zhang, xinyuan.cai, xubo}@ia.ac.cn

## Abstract

End-to-end speech recognition systems have been successfully designed for English. Taking into account the distinctive characteristics between Chinese Mandarin and English, it is worthy to do some additional work to transfer these approaches to Chinese. In this paper, we attempt to build a Chinese speech recognition system using end-to-end learning method. The system is based on a combination of deep Long Short-Term Memory Projected (LSTMP) network architecture and the Connectionist Temporal Classification objective function (CTC). The Chinese characters (the number is about 6,000) are used as the output labels directly. To integrate language model information during decoding, the CTC Beam Search method is adopted and optimized to make it more effective and more efficient. We present the first-pass decoding results which are obtained by decoding from scratch using CTC-trained network and language model. Although these results are not as good as the performance of DNN-HMMs hybrid system, they indicate that it is feasible to choose Chinese characters as the output alphabet in the end-to-end speech recognition system.

**Index Terms**: Long Short-Term Memory, End-to-end, Connectionist Temporal Classification, speech recognition

## 1. Introduction

It is a complex and domain-specific task to build modern large vocabulary continuous speech recognition (LVCSR) systems, which rely on heavily engineered processing stages, including input features, acoustic models, language models and Hidden Markov Models (HMMs). While speech recognition has benefited a lot from the introduction of deep learning algorithms [1, 2, 3, 4, 5, 6], these algorithms usually focus on improving acoustic models which is only a single component in the complex pipeline. Neural networks for speech recognition are typically trained to classify individual frames of acoustic data using cross-entropy criteria as the objective function, which is substantially different from the true performance measure (sequence-level transcription accuracy). This inconsistency leads to a phenomenon that a great increase in frame accuracy may translate to negligible improvement, or even deterioration of transcription accuracy. To solve this problem, sequence-discriminative criteria such as maximum mutual information (MMI) [7], boosted MMI (BMMI) [8], and minimum Bayes risk (MBR) [9] have been used to further optimize a well-trained Deep Neural Network (DNN)-HMMs hybrid system [10, 11, 12]. However, these methods are only effective when they are used to retrain a system already trained on frame-level. In a way, this makes the pipelines of speech recognition systems even more sophisticated. Another problem exists when cross-entropy is used as the objective function. That is, the frame-level training targets must be obtained in advance. This is often done by inferring from the alignments determined by an already well-trained GMM-HMMs speech recognition system. This also increases the difficulty of building modern LVCSR systems.

With the goal of building a system where as much of the speech pipeline as possible is replaced by a single recurrent neural network (RNN), [13] presents a speech recognition system that directly transcribes audio data with text, without requiring an intermediate phonetic representation. This end-to-end learning approach treats speech recognition as a direct sequence transduction problem and discards many of the assumptions present in modern HMM-based LVCSR systems. The system in [13] is based on a combination of the deep bidirectional Long Short-Term Memory (LSTM) network [14] and the Connectionist Temporal Classification (CTC) [15] objective function. With CTC function, a RNN can be trained for sequence transcription tasks without requiring any prior alignment between the input and target sequences. Inspired by [13], [16] presents a state-of-the-art speech recognition system called DeepSpeech, which is developed using end-to-end deep learning. In DeepSpeech, Bi-Directional Recurrent Deep Neural Networks (BRDNN) is chosen and trained using thousands of hours of data to directly generate English text transcriptions. The output alphabet consists of 26 letters, punctuation marks, as well as tokens for 'blank' symbol and space. The decoding algorithm in DeepSpeech, described in [17] in detail, is a first-pass decoding approach and incorporates a language model constraint whenever the algorithm proposes appending a space character.

The works listed above all focus on English speech recognition. It is not so straightforward to apply approaches described above to end-to-end speech recognition for Chinese Mandarin because several distinctive characteristics exist between these two languages. Firstly, there are no spaces between Chinese words, which makes the decoding algorithm described in [17] not suitable for this task. What's more, Chinese words are graphemically made up of characters. The set of most common 6,000 Chinese characters has a sufficient coverage for most user scenarios. It is reasonable to choose Chinese characters as the output alphabet directly. However, the end-to-end speech recognition for English contains just dozens of output labels. Thus it is necessary to discuss whether the method can perform well with thousands of labels, such as the 6,000 Chinese characters. Additionally, the Chinese language has a closed set of Constant-Vowel structured syllables, the number of which is about 400 as

The work was supported by 973 program in china, grant No. Y3A2011D81.

for the toneless ones. It provides another choice of output alphabet other than characters.

In this paper, we attempt to build an end-to-end speech recognition system for Chinese mandarin. The core of our system is a LSTM Projected (LSTMP) network [4, 5] to ingest speech features and generate Chinese transcriptions. To make a better understanding of the end-to-end method, comparative experiments are performed for different amounts of training data and the depth of the model is also explored in this paper. The CTC beam search algorithm [13] is also optimized to make the decoding more effective and more efficient.

The remainder is organized as follows. Section 2 describes the network architecture. A brief introduction to CTC objective function is given in Section 3 and the decoding algorithms are introduced in Section 4. We report our experimental results in Section 5 and conclude this work in Section 6.

## 2. Network Architecture

In contrast to the fixed contextual windows of inputs to DNNs, RNNs use a dynamically changing contextual window of all sequence history. This capability makes RNNs better suited for sequence modeling tasks. However, it is difficult to train conventional RNNs due to the vanishing gradient and exploding gradient problems [18]. In practice this shortcoming makes it hard for RNNs to learn tasks containing delays of more than about 10 timesteps between relevant input and target events [19]. To address these problems, LSTM has been designed as an elegant RNN architecture.

In the standard architecture of LSTM networks, there are an input layer, a LSTM recurrent layer and an output layer. To address the computational complexity of training LSTM models, LSTMP is proposed in [4], which has a separate linear projection layer on top of the LSTM layer. Experimental results in [4, 5] show that this projection layer can make better use of the model parameters and is important for the nice performance of LSTM-HMMs hybrid systems. We believe that this will also hold for the end-to-end speech recognition. Thus LSTMP is chosen as our model architecture and is shown in Fig. 1(a).

Given an input sequence $\mathbf{x} = (x_1, ..., x_T)$, a LSTM-P network computes a mapping to an output sequence $\mathbf{y} = (y_1, ..., y_T)$ by calculating the network activations using the following equations iteratively from $t = 1$ to $T$:

$$i_t = \sigma\left(W_{ix}x_t + W_{ip}p_{t-1} + W_{ic}c_{t-1} + b_i\right) \quad (1)$$
$$f_t = \sigma\left(W_{fx}x_t + W_{fp}p_{t-1} + W_{fc}c_{t-1} + b_f\right) \quad (2)$$
$$a_t = g\left(W_{cx}x_t + W_{cp}p_{t-1} + b_c\right) \quad (3)$$
$$c_t = f_t c_{t-1} + i_t a_t \quad (4)$$
$$o_t = \sigma\left(W_{ox}x_t + W_{op}p_{t-1} + W_{oc}c_t + b_o\right) \quad (5)$$
$$p_t = W_{pm}\left(o_t h\left(c_t\right)\right) \quad (6)$$
$$y_t = softmax\left(W_{yp}p_t + b_y\right) \quad (7)$$

where $\sigma$ is the logistic sigmoid function, and $i$, $f$, $o$, $a$ and $c$ are respectively the input gate, forget gate, output gate, cell input activation and cell state vectors. The $W$ terms and $b$ terms denote weight matrices and bias vectors respectively. $W_{ic}$, $W_{fc}$ and $W_{oc}$ are diagonal weight matrices for peephole connections. $g$ and $h$ are the cell input and cell output activation functions, generally $\tanh$ in this paper.

Fig. 1(b) shows the architecture of deep LSTMP (DLSTM-P) where multiple LSTMP layers are stacked on top of each other. This model is introduced for acoustic modeling in [5]

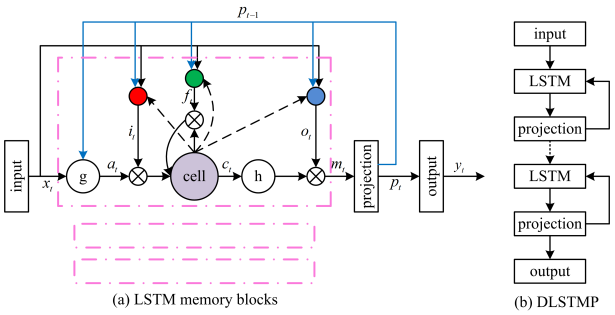


Figure 1: *Network architecture.*

and shows a significant performance improvement compared with the shallow one. DLSTMP is also examined for end-to-end speech recognition in this paper. We choose unidirectional LSTM instead of bidirectional one because [20] shows that bidirectional LSTM only has a slight advantage yet has a shortcoming of unsuitable for using on-line.

## 3. Connectionist Temporal Classification

Connectionist Temporal Classification (CTC) [15, 21] is an objective function which amounts to maximizing the likelihood of an output sequence by efficiently summing over all possible input-output sequence alignments. It uses a softmax output layer to define a separate output distribution $P\left(k|t\right)$ at every step $t$ along the input sequence for all the transcription labels plus an extra 'blank' symbol which represents a non-output. Intuitively the network decides whether to emit any label, or no label, at every time-step. Given a length $T$ input sequence $\mathbf{x}$ and the output vectors $y_t$, the probability of emitting the label or blank with index $k$ at time $t$ is given as:

$$P\left(k|t, \mathbf{x}\right) = \frac{\exp\left(y_t^k\right)}{\sum_{k'} \exp\left(y_t^{k'}\right)} \quad (8)$$

where $y_t^k$ is element $k$ of $y_t$. A CTC path (or alignment) $\boldsymbol{\pi}$ is a length $T$ sequence of blank and label indices. The probability $P\left(\boldsymbol{\pi}|\mathbf{x}\right)$ is the product of the emission probabilities at every time-step:

$$P\left(\boldsymbol{\pi}|\mathbf{x}\right) = \prod_{t=1}^{T} P\left(\pi_t|t, \mathbf{x}\right) \quad (9)$$

For a given transcription sequence, there are as many possible paths as there different ways of separating the labels with blanks. To map from these paths to the transcription, a many-to-one map $\mathcal{B}$ can be defined that removes first the repeated labels and then the blanks from paths. The conditional probability of an output transcription $\mathbf{y}$ can be calculated by summing the probabilities of all the paths mapped onto it by $\mathcal{B}$:

$$P\left(\mathbf{y}|\mathbf{x}\right) = \sum_{\boldsymbol{\pi} \in \mathcal{B}^{-1}(\mathbf{y})} P\left(\boldsymbol{\pi}|\mathbf{x}\right) \quad (10)$$

This 'collapsing together' of different paths onto the same transcription is what allows CTC to use unsegmented data, because it removes the requirement of knowing where in the input sequence the labels occur. Given a target transcription $\mathbf{y}^*$, the network can then be trained to minimise the CTC objective function:

$$CTC\left(\mathbf{x}\right) = -\log P\left(\mathbf{y}^*|\mathbf{x}\right) \quad (11)$$

# 4. Decoding

Decoding a CTC network means to find the most probable output transcription $\mathbf{y}$ for a given input sequence $\mathbf{x}$. Two approaches are employed in this paper.

## 4.1. Best Path Decoding

The first method, which refer to as best path decoding, is based on the assumption that the most probable path corresponds to the most probable transcription:

$$\mathbf{y}^* \approx \mathcal{B}(\boldsymbol{\pi}^*) \tag{12}$$

$$\text{where } \boldsymbol{\pi}^* = \arg\max_{\boldsymbol{\pi}} P(\boldsymbol{\pi}|\mathbf{x})$$

Best path decoding is trivial to compute, since $\boldsymbol{\pi}^*$ is just the concatenation of the most active outputs at every time-step. Sequence $\mathbf{y}^*$ can be scored against the reference transcription using Label Error Rate (LER). When Chinese characters are used as output labels, LER is just Character Error Rate (CER), which is the performance metric used for Chinese speech recognition.

## 4.2. Constrained Beam Search Decoding

The CTC Beam Search algorithm proposed in [13] allows the integration of a dictionary and language model. However, in [13], this algorithm is only tested when just a dictionary included during decoding. To integrate language model information, instead of using the beam search method, the authors use their CTC-trained neural network to rescore an n-best list extracted from a state-of-the-art DNN-HMMs system. It is partly due to implementation difficulties as mentioned in [13]. This makes the system still relying on HMMs speech recognition infrastructure.

In this paper, we adopt this beam search algorithm and present first-pass decoding results using a neural network and language model to decode from scratch, rather than re-ranking an existing set of hypotheses. What's more, to make the decoding more efficient and more effective, the algorithm is optimized in two ways:

- A hyper-parameter $\alpha$ that accounts for language model weight, is added into the expression of extension probability which is defined as follows:

$$P(k, \mathbf{y}, t) = P(k|t, \mathbf{x})\, P^{\alpha}(k|\mathbf{y}) \begin{cases} P^{-}(\mathbf{y}, t-1) \text{ if } \mathbf{y}^e = k \\ P(\mathbf{y}, t-1) \text{ otherwise} \end{cases} \tag{13}$$

  where $P(k, \mathbf{y}, t)$ is the extension probability of $\mathbf{y}$ by label $k$ at time $t$, $P(k|t, \mathbf{x})$ is the CTC emission probability of $k$ at $t$, $P(k|\mathbf{y})$ is the transition probability from $\mathbf{y}$ to $\mathbf{y} + k$ that can be used to integrate prior linguistic information into the search and can be re-weighted with parameter $\alpha$.

- During decoding, two sets $B_{\text{prev}}$ and $B_{\text{next}}$ are kept to maintain a list of active prefixes at the previous time step and proposed prefixes at the next time step respectively. In the original algorithm, the size of $B_{\text{next}}$ is equal to $W(K+1)$, where $W$ is the beam width and $K$ is the number of labels. In this paper, just like $B_{\text{prev}}$, the size of $B_{\text{next}}$ is also constrained to be never larger than $W$. A proposed prefix will **not** be pushed into $B_{\text{next}}$ if its path score is less than the minimum score in $B_{\text{next}}$. This trick makes the decoding much faster and taking up less memory.

# 5. Experiments

Experiments are carried out on a 130-hour Chinese Mandarin transcription task, of which the training set contains 142592 utterances (about 130 hours). A held-out development set (about 2 hours) is used to monitor the model training process and to find a good setting of the parameter $\alpha$ during decoding. The performance is evaluated on another 2-hour test set, consisting of 2300 utterances.

## 5.1. Baseline Systems

Two hybrid baseline systems are built in this paper, namely DNN-HMMs and LSTMP-HMMs. To get the alignments needed by these two systems, tied-state cross-word tri-phone GMM-HMMs are first trained with maximum likelihood estimation (MLE) using the regular 42-dimension features, which consist of 13-dimensional PLP and smoothed F0 appended with the first and second order derivatives. The GMM-HMMs model contains 5120 tied tri-phone states, with an average of 40 Gaussian components per state.

### *5.1.1. DNN-HMMs*

The DNN model in the hybrid DNN-HMMs system containing 5 hidden layers with 2048 units in each layer and an output layer with 5120 senones, is first pre-trained using RBM-based greedy layer-wise pre-training and then fine-tuned using state labels obtained by forced alignment of the trained GMM-HMMs. Concatenations of 11 frames (5+1+5) of 42-dimensional PLP features are used as input of the network.

### *5.1.2. LSTMP-HMMs*

For the LSTMP RNN architecture, there are two ways to utilize the information from the future frames to make better decisions for the current frame. The first one is to add a time-window of future context to the network input, and the second one is to introduce a delay between the inputs and the targets. In [4, 5, 22], the second method is adopted while in this paper, we choose the first one since it performs better in our experiments. Thus, the inputs to LSTMP network are concatenated features, which are produced by concatenating the current frame with 5 frames in its right context (0+1+5).

The deep LSTMP architecture with two layers each with 800 cells and 512 recurrent projection units is first pre-trained using the discriminative pre-training algorithm [23]. In the training procedure, a time step of 40 ($T_{bptt}$) is used to forward-propagate the activations and backward-propagate the gradients using the truncated BPTT learning algorithm [24]. For computational efficiency, one GPU operates in parallel on 80 subsequences from different utterances at a time. To further increase the training speed, the model is trained on our asynchronous stochastic gradient descent (ASGD) training platform [25]. The learning rate is decreased exponentially with an initial learning rate of 1e-04.

The performance of these two baseline systems is evaluated on the test set and shown in Table 1.

## 5.2. End-to-end Systems

The LSTMP model in the end-to-end system has the same architecture with that in LSTMP-HMMs hybrid system, except for the output layer, which has 6,725 units (6,724 units for Chinese characters set and 1 for 'blank' symbol). Discriminative pre-training is first preformed followed by fine-tuning on

the ASGD platform. Initial learning rate ranged from 1e-05 to 6e-05 are set specific to a network architecture for stable convergence of each network.

Two sets of language models (LMs) with different n-gram orders are trained using transcription of 130-hour training set. One set is based on word unit and the other character unit. The hybrid baseline systems decode with both sets of LMs while for end-to-end system only character-based LMs are used since the CTC beam search algorithm can not integrate word-based LMs directly.

#### 5.2.1. Varying modeling power of LMs

We first compare the performance of these systems on the test set when decoding with different language models by varying the n-gram order. The results are shown in Table 1.

Table 1: *System performance with different LMs [CER %]. 'No LM' in second line means best path decoding for CTC system.*

| LM Order | Character LM | | | Word LM | |
|---|---|---|---|---|---|
| | LSTMP-CTC | DNN-HMMs | LSTMP-HMMs | DNN-HMMs | LSTMP-HMMs |
| No LM | 39.4 | — | — | — | — |
| 2 | 29.5 | 20.3 | 17.1 | 15.9 | 13.0 |
| 3 | 19.2 | 12.7 | 11.4 | **11.6** | 9.8 |
| 4 | **16.4** | 12.6 | 11.9 | 11.7 | **9.5** |
| 5 | 17.4 | **12.3** | 11.5 | 11.7 | 9.6 |
| 6 | 19.6 | 12.4 | **11.3** | 11.9 | 9.6 |

Several observations can be made from Table 1. **First**, comparing the results of two hybrid systems, LSTMP-HMMs always perform better than DNN-HMMs when the same language model is used. This indicates that LSTMP is a more powerful acoustic model than DNN. **Second**, for end-to-end system, CER is quite high without any sort of language constraint (39.4% in the second line). This is consistent with our observation that the best path decoding results are quite similar with true reference in pronunciation while different in grapheme. When language model information is integrated, CER decreases a lot, from 39.4% to 16.4% when decoding using character-based 4-gram LM with decoding parameters $W = 300, \alpha = 1.4$. **Third**, with the increased power of character-based LMs, the performance of hybrid systems tends to converge. However, for end-to-end system, too large LM order will lead to performance loss. A possible explanation is that too long character history will skew the inter-label transitions learned directly from the data during CTC training.

#### 5.2.2. Increasing depth of LSTMP

The results in Table 1 show that the two layer LSTMP-CTC network performs worse than the baseline hybrid systems. A potential method to increase modeling capabilities is to make the model deeper. In this subsection, experiments are carried out to explore effects of model depth in end-to-end system. The results are shown in Table 2 and indicate that depth is very important in the end-to-end system which coincides with previous findings for deep neural networks [1] in hybrid systems. With model depth increased from 1 to 3, the performance is improved from 19.7% to 13.9%. Although the final result is not as good as the baseline system, it can be concluded that it is feasible to build an end-to-end speech recognition system for Chinese Mandarin with characters as the output alphabet directly.

Table 2: *End-to-end system performance with different depths of LSTMP models [CER %].*

| System | Depth | Character LM Order | | |
|---|---|---|---|---|
| | | 4 | 5 | No LM |
| LSTMP-CTC | 01 | 19.7 | 21.0 | 47.8 |
| | 02 | 16.4 | 17.4 | 39.4 |
| | 03 | **13.9** | 14.3 | 32.2 |
| DNN-HMMs | | 12.6 | **12.3** | — |

#### 5.2.3. Varying training data size of LSTMP

The authors in [13] found that for English, when transcribed speech is insufficient, it is hard for the RNN to learn how to 'spell' words correctly whereas it may be easier to learn to identify phonemes, thus DNN-HMMs performs much better than RNN-CTC in this case. We doubt that this may not hold for Mandarin since in Chinese each character is independent and words are not 'spelled' by characters. The RNN only needs to learn local relationships among characters, without the need for learning the 'spelling' rules. In this subsection, comparative experiments are conducted with different amounts of training data. LSTMP contains two layers and DNN has the same architecture as the baseline system. Table 3 shows the results.

Table 3: *System performance with different hours of training data [CER %].*

| Train Data (Hour) | LSTMP-CTC | | | DNN-HMMs | |
|---|---|---|---|---|---|
| | No LM | 4-gram | 5-gram | 4-gram | 5-gram |
| 20 | 59.3 | 31.9 | 34.1 | 33.2 | 28.5 |
| 70 | 50.2 | 25.3 | 26.8 | 21.0 | 20.2 |
| 130 | 39.4 | 16.4 | 17.4 | 12.6 | 12.3 |

It can be seen that with the increase of training data, the performance of both systems is improved a lot. What's more, performance improvement trends for these two systems are similar. When training data is inadequate (e.g. 20-hour), DNN-HMMs does not perform much better than LSTMP-CTC, compared with the situation where relatively sufficient data is provided (e.g. 130-hour). This phenomenon differs with the finding in [13] which can be attributed to linguistic difference between Mandarin and English.

## 6. Conclusions

In this work, we attempt to build an end-to-end speech recognition system for Chinese Mandarin. The system is based on a combination of LSTMP network and CTC objective function. Chinese characters are used as the output labels directly. To decode from scratch with language models, CTC beam search algorithm is adopted and optimized. Although the resulting system is not as good as the baseline, it indicates that the 6,000 Chinese characters can be modeled directly. We also find that depth is very important and when training data is insufficient, different with findings in English, the hybrid system does not perform much better than the end-to-end system for Mandarin.

In the future, more research will be done on decoding algorithm with the aim to make it suitable for word-based language models. Other output labels, such as syllables and tonal syllables, will also be used as the targets.

# 7. References


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

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

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

[4] H. Sak, A. Senior, and F. Beaufays, "Long short-term memory based recurrent neural network architectures for large vocabulary speech recognition," *arXiv preprint arXiv:1402.1128*, 2014.

[5] ——, "Long short-term memory recurrent neural network architectures for large scale acoustic modeling," in *Proceedings of the Annual Conference of International Speech Communication Association (INTERSPEECH)*, 2014.

[6] H. Sak, O. Vinyals, G. Heigold, A. Senior, E. McDermott, R. Monga, and M. Mao, "Sequence discriminative distributed training of long short-term memory recurrent neural networks," *entropy*, vol. 15, no. 16, pp. 17–18, 2014.

[7] Y. Normandin, "Maximum mutual information estimation of hidden markov models," in *Automatic Speech and Speaker Recognition*. Springer, 1996, pp. 57–81.

[8] D. Povey, D. Kanevsky, B. Kingsbury, B. Ramabhadran, G. Saon, and K. Visweswariah, "Boosted mmi for model and feature-space discriminative training," in *Acoustics, Speech and Signal Processing, 2008. ICASSP 2008. IEEE International Conference on*. IEEE, 2008, pp. 4057–4060.

[9] M. Gibson and T. Hain, "Hypothesis spaces for minimum bayes risk training in large vocabulary speech recognition." in *INTERSPEECH*. Citeseer, 2006.

[10] T. S. Brian Kingsbury and H. Soltau, "Scalable minimum bayes risk training of deep neural network acoustic models using distributed hessian-free optimization." in *INTERSPEECH*, 2012.

[11] H. Su, G. Li, D. Yu, and F. Seide, "Error back propagation for sequence training of context-dependent deep networks for conversational speech transcription," in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*. IEEE, 2013, pp. 6664–6668.

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

[13] A. Graves and N. Jaitly, "Towards end-to-end speech recognition with recurrent neural networks," in *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, 2014, pp. 1764–1772.

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

[15] 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.

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

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

[18] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *Neural Networks, IEEE Transactions on*, vol. 5, no. 2, pp. 157–166, 1994.

[19] S. Hochreiter, Y. Bengio, P. Frasconi, and J. Schmidhuber, "Gradient flow in recurrent nets: the difficulty of learning long-term dependencies," 2001.

[20] A. Graves, A.-R. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*. IEEE, 2013, pp. 6645–6649.

[21] A. Graves, *Supervised sequence labelling with recurrent neural networks*. Springer, 2012, vol. 385.

[22] X. Li and X. Wu, "Constructing long short-term memory based deep recurrent neural networks for large vocabulary speech recognition," *CoRR*, vol. abs/1410.4281, 2014. [Online]. Available: http://arxiv.org/abs/1410.4281

[23] 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.

[24] R. J. Williams and J. Peng, "An efficient gradient-based algorithm for on-line training of recurrent network trajectories," *Neural computation*, vol. 2, no. 4, pp. 490–501, 1990.

[25] S. Zhang, C. Zhang, Z. You, R. Zheng, and B. Xu, "Asynchronous stochastic gradient descent for dnn training," in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*. IEEE, 2013, pp. 6660–6663.