HYBRID SPEECH RECOGNITION WITH DEEP BIDIRECTIONAL LSTM

Alex Graves, Navdeep Jaitly and Abdel-rahman Mohamed

University of Toronto
Department of Computer Science
6 King's College Rd. Toronto, M5S 3G4, Canada

ABSTRACT

Deep Bidirectional LSTM (DBLSTM) recurrent neural networks have recently been shown to give state-of-the-art performance on the TIMIT speech database. However, the results in that work relied on recurrent-neural-network-specific objective functions, which are difficult to integrate with existing large vocabulary speech recognition systems. This paper investigates the use of DBLSTM as an acoustic model in a standard neural network-HMM hybrid system. We find that a DBLSTM-HMM hybrid gives equally good results on TIMIT as the previous work. It also outperforms both GMM and deep network benchmarks on a subset of the Wall Street Journal corpus. However the improvement in word error rate over the deep network is modest, despite a great increase in frame-level accuracy. We conclude that the hybrid approach with DBLSTM appears to be well suited for tasks where acoustic modelling predominates. Further investigation needs to be conducted to understand how to better leverage the improvements in frame-level accuracy towards better word error rates.

Index Terms— DBLSTM, HMM-RNN hybrid

1. INTRODUCTION

Deep Bidirectional LSTM was recently introduced to speech recognition, giving the lowest recorded error rates on the TIMIT database [1]. In that work the networks were trained with two end-to-end training methods designed for discriminative sequence transcription with recurrent neural networks, namely Connectionist Temporal Classification [2] and Sequence Transduction [3]. These methods have several advantages: they do not require forced alignments to pre-segment the acoustic data, they directly optimise the probability of the target sequence conditioned on the input sequence, and (especially in the case of Sequence Transduction) they are able to learn an implicit language model from the acoustic training data. However neither method can readily be integrated into existing large vocabulary speech recognition systems, which were designed around the GMM-HMM paradigm. In particular, it is not straightforward to combine them with word-level language models, which play a vital role in real-world tasks.

The standard solution to the problem of training neural networks for speech recognition is to merge them with HMMs in the so-called *hybrid* [4] or *tandem* [5] models. The hybrid approach, in particular, has gained prominence in recent years with the performance improvements yielded by deep networks [6, 7]. In this framework a forced alignment given by a GMM-HMM system is used to create frame-level acoustic targets which the network attempts to classify. Using a forced alignment has the advantage of providing a straightforward objective function (cross-entropy error) for network training. Recognition is performed by combining the acoustic probabilities yielded by the network with the state transition probabilities from the HMM and the word transition probabilities from the language model[1], which can be done efficiently for large vocabulary speech using weighted finite state transducers.

One of the original motivations for the hybrid approach was to increase the amount of context used by the acoustic model. In modern hybrid systems, acoustic context windows of 11 to 21 frames are typically provided to the network. Recurrent Neural Networks (RNNs) can learn how much context to use for each prediction, and are in principle are able to access information from anywhere in the acoustic sequence. It is therefore unsurprising that HMM-RNN hybrids have been considered for almost twenty years [8, 9, 10]. So far however, they have not become a staple of large vocabulary speech recognition.

The two main goals of this paper are to compare the performance of DBLSTM-HMM hybrids with the end-to-end sequence training defined in [1], and to assess the potential of DBLSTM-HMM hybrids for large vocabulary speech recognition.

The network architecture is described in Section 2 and the training method is described in 3. An experimental comparison with end-to-end training on the TIMIT database is given in Section 4, and a comparison with deep networks and GMMs on the Wall Street Journal corpus is provided by Section 5. Section 6 contains discussion of the results and their implications for DBLSTM training and concluding remarks

[1] In practice the HMM state transitions have become less signficant as linguistic and acoustic models have improved, and many current systems ignore them altogether.

are given in Section 7.

2. NETWORK ARCHITECTURE

Given an input sequence $\boldsymbol{x} = (x_1, \ldots, x_T)$, a standard recurrent neural network (RNN) computes the hidden vector sequence $\boldsymbol{h} = (h_1, \ldots, h_T)$ and output vector sequence $\boldsymbol{y} = (y_1, \ldots, y_T)$ by iterating the following equations from $t = 1$ to T:

$$h_t = \mathcal{H}\left(W_{xh}x_t + W_{hh}h_{t-1} + b_h\right) \quad (1)$$

$$y_t = W_{hy}h_t + b_y \quad (2)$$

where the W terms denote weight matrices (e.g. W_{xh} is the input-hidden weight matrix), the b terms denote bias vectors (e.g. b_h is hidden bias vector) and $\mathcal{H}$ is the hidden layer function.

$\mathcal{H}$ is usually an elementwise application of a sigmoid function. However we have found that the Long Short-Term Memory (LSTM) architecture [11], which uses purpose-built *memory cells* to store information, is better at finding and exploiting long range context. Fig. 1 illustrates a single LSTM memory cell. For the version of LSTM used in this paper [12] $\mathcal{H}$ is implemented by the following composite function:

$$i_t = \sigma\left(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci}c_{t-1} + b_i\right) \quad (3)$$

$$f_t = \sigma\left(W_{xf}x_t + W_{hf}h_{t-1} + W_{cf}c_{t-1} + b_f\right) \quad (4)$$

$$c_t = f_t c_{t-1} + i_t \tanh\left(W_{xc}x_t + W_{hc}h_{t-1} + b_c\right) \quad (5)$$

$$o_t = \sigma\left(W_{xo}x_t + W_{ho}h_{t-1} + W_{co}c_t + b_o\right) \quad (6)$$

$$h_t = o_t \tanh(c_t) \quad (7)$$

where σ is the logistic sigmoid function, and i, f, o and c are respectively the *input gate*, *forget gate*, *output gate* and *cell* activation vectors, all of which are the same size as the hidden vector h. The weight matrices from the cell to gate vectors (e.g. W_{si}) are diagonal, so element m in each gate vector only receives input from element m of the cell vector.

One shortcoming of conventional RNNs is that they are only able to make use of previous context. In speech recognition, where whole utterances are transcribed at once, there is no reason not to exploit future context as well. Bidirectional RNNs (BRNNs) [13] do this by processing the data in both directions with two separate hidden layers, which are then fed forwards to the same output layer. As illustrated in Fig. 2, a BRNN computes the *forward* hidden sequence $\overrightarrow{\boldsymbol{h}}$, the *backward* hidden sequence $\overleftarrow{\boldsymbol{h}}$ and the output sequence $\boldsymbol{y}$ by iterating the backward layer from $t = T$ to 1, the forward layer from $t = 1$ to T and then updating the output layer:

$$\overrightarrow{h}_t = \mathcal{H}\left(W_{x\overrightarrow{h}}x_t + W_{\overrightarrow{h}\overrightarrow{h}}\overrightarrow{h}_{t-1} + b_{\overrightarrow{h}}\right) \quad (8)$$

$$\overleftarrow{h}_t = \mathcal{H}\left(W_{x\overleftarrow{h}}x_t + W_{\overleftarrow{h}\overleftarrow{h}}\overleftarrow{h}_{t+1} + b_{\overleftarrow{h}}\right) \quad (9)$$

$$y_t = W_{\overrightarrow{h}y}\overrightarrow{h}_t + W_{\overleftarrow{h}y}\overleftarrow{h}_t + b_y \quad (10)$$

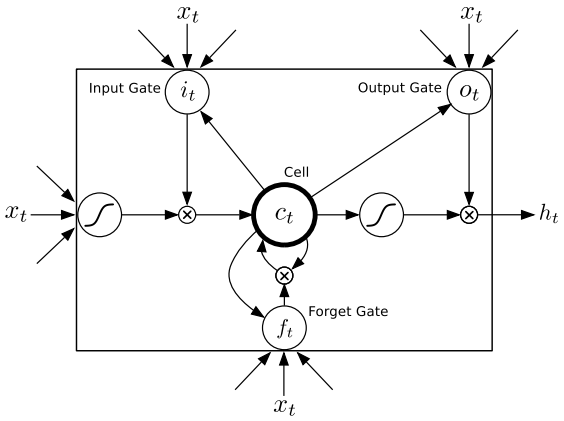

Fig. 1. Long Short-term Memory Cell

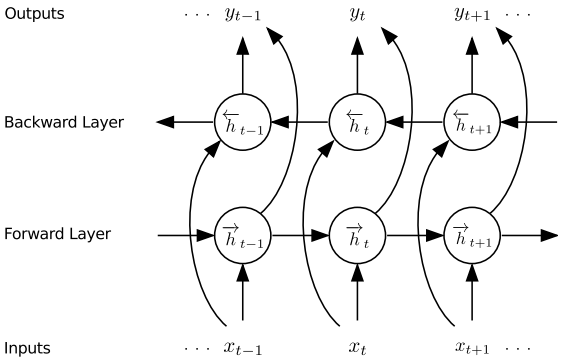

Fig. 2. Bidirectional Recurrent Neural Network

Combing BRNNs with LSTM gives bidirectional LSTM [14], which can access long-range context in both input directions.

A crucial element of the recent success of hybrid systems is the use of *deep* architectures, which are able to build up progressively higher level representations of acoustic data. *Deep RNNs* can be created by stacking multiple RNN hidden layers on top of each other, with the output sequence of one layer forming the input sequence for the next, as shown in Fig. 3. Assuming the same hidden layer function is used for all N layers in the stack, the hidden vector sequences $\boldsymbol{h}^n$ are iteratively computed from $n = 1$ to N and $t = 1$ to T:

$$h_t^n = \mathcal{H}\left(W_{h^{n-1}h^n}h_t^{n-1} + W_{h^n h^n}h_{t-1}^n + b_h^n\right) \quad (11)$$

where we define $\boldsymbol{h}^0 = \boldsymbol{x}$. The network outputs y_t are

$$y_t = W_{h^N y}h_t^N + b_y \quad (12)$$

Deep bidirectional RNNs can be implemented by replacing each hidden sequence $\boldsymbol{h}^n$ with the forward and backward sequences $\overrightarrow{\boldsymbol{h}}^n$ and $\overleftarrow{\boldsymbol{h}}^n$, and ensuring that every hidden layer receives input from both the forward and backward layers at the level below. If LSTM is used for the hidden layers we get deep bidirectional LSTM, as illustrated in Fig. 4.

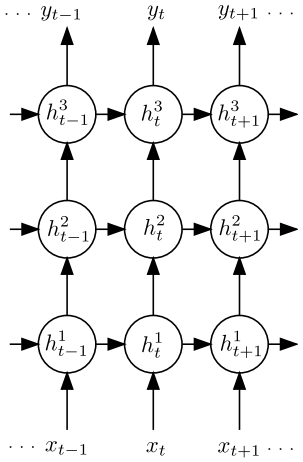

Fig. 3. Deep Recurrent Neural Network

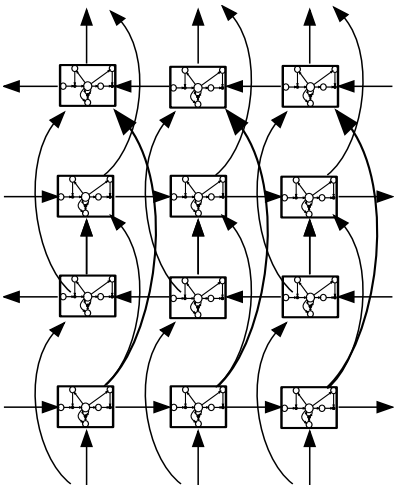

Fig. 4. Deep Bidirectional Long Short-Term Memory Network (DBLSTM)

3. NETWORK TRAINING

Network training follows the standard approach used in hybrid systems [4]. Frame-level state targets are provided on the training set by a forced alignment given by a GMM-HMM system. The network is then trained to minimise the cross-entropy error of the targets using a softmax output layer with as many units as the total number of possible HMM states. At decoding time, the state probabilities yielded by the network are combined with a dictionary and language model to determine the most probable transcription. For a length T acoustic sequence $\boldsymbol{x}$ the network produces a length T output sequence $\boldsymbol{y}$, where each y_t defines a probability distribution over the K possible states: that is, y_t^k (the k^{th} element of y_t) is the network's estimate for the probability of observing state k at time t given $\boldsymbol{x}$. Given a length T state target sequence $\boldsymbol{z}$ the network is trained to minimise the negative log-probability of the target sequence given the input sequence:

$$-\log \Pr(\boldsymbol{z}|\boldsymbol{x}) = -\sum_{t=1}^{T} \log y_t^{z_t} \tag{13}$$

Which leads to the following error derivatives at the output layer

$$-\frac{\partial \log \Pr(\boldsymbol{z}|\boldsymbol{x})}{\partial \hat{y}_t^k} = y_t^k - \delta_{k,z_t} \tag{14}$$

where $\hat{y}_t$ is the vector of output activations before they have been normalised with the softmax function. These derivatives are then fed back through the network using backpropagation through time to determine the weight gradient.

When training deep networks in hybrid systems with stochastic gradient descent it has been found advantageous to select minibatches of frames randomly from the whole training set, rather than using whole utterances as batches. This is impossible with RNN-HMM hybrids because the weight gradients are a function of the entire utterance.

Another difference is that hybrid deep networks are trained with an acoustic context window of frames to either side of the one being classified. This is not necessary for DBLSTM, since it is as able to store past and future context internally, and the data was therefore presented a single frame at a time.

For some of the experiments Gaussian noise was added to the network weights during training [15]. The noise was added once per training sequence, rather than at every timestep. Weight noise tends to 'simplify' neural networks, in the sense of reducing the amount of information required to transmit the parameters [16, 17], which improves generalisation.

4. TIMIT EXPERIMENTS

The first set of experiments were carried out on the TIMIT [18] speech corpus. Their purpose was to see how hybrid training for deep bidirectional LSTM compared with the end-to-end training methods described in [1]. To this end, we ensured that the data preparation, network architecture and training parameters were consistent with those in the previous work. To allow us to test for significance, we also carried out repeated runs of the previous experiments (which were only run once in the original paper). In addition, we ran hybrid experiments using a deep bidirectional RNN with $tanh$ hidden units instead of LSTM.

The standard 462 speaker set with all SA records removed was used for training, and a separate development set of 50 speakers was used for early stopping. Results are reported for the 24-speaker core test set. The audio data was preprocessed using a Fourier-transform-based filterbank with 40 coefficients (plus energy) distributed on a mel-scale, together with their first and second temporal derivatives. Each input

vector was therefore size 123. The data were normalised so that every element of the input vectors had zero mean and unit variance over the training set. All 61 phoneme labels were used during training and decoding (so $K = 61$), then mapped to 39 classes for scoring [19]. All experiments were repeated four times with different random initialisations, and results are quoted as the mean $\pm$ the std. dev.

Table 1 shows the phoneme error rate (PER) for DBLSTM trained with the two methods described in [1]: Connectionist Temporal Classification ('CTC') and Sequence Transduction ('Transducer'). Both networks consisted of five bidirectional hidden levels, each containing two LSTM layers of 250 cells, along with a size 62 softmax output layer (one unit for each phoneme, plus an extra blank unit). The sequence transduction network had an additional phoneme prediction network with a single hidden layer of 250 LSTM cells, and an output network with a single hidden layer of 250 $tanh$ units. The CTC network had approximately 6.8M weights and the Transducer network had approximately 7.4M. All networks were trained using stochastic gradient descent, with learning rate 10^{-4}, momentum 0.9 and random initial weights drawn uniformly from $[-0.1, 0.1]$. The CTC networks were first trained to convergence with no noise, then retrained with weight noise (std. dev. 0.075). The Transducer networks were initialised with the weights of the CTC networks after retraining with noise. The Transducer phoneme error rate of 18.07 $\pm$ 0.24 is consistent with the single result of 17.7 recorded in [1]. Indeed, the single best Transducer run in this paper (the one achieving lowest PER on the development set) also returned 17.7 on the test set.

For hybrid training on TIMIT a phonetic dictionary was used, with three states per phoneme, giving 183 target states in total. A biphone language model was estimated on the training set, and a simple GMM-HMM system was used to provide forced alignments. The posterior state probabilities provided by the networks were not divided by the state occupancy priors, as this has been found to make no difference on TIMIT [6]. Table 2 shows the phoneme error rates for hybrid training with DBLSTM and Deep Bidirectional RNN (DBRNN), along with the frame error rate (FER) and cross-entropy error (CE) in units of nats per frame. The DBLSTM networks had the same architecture as the CTC networks described above, except that the output layer had 183 units (one for each HMM state). As before, each randomly initalised LSTM network was first trained to convergence, then retrained with weight noise. The DBRNN network had 5 bidirectional levels with 500 tanh units in each, giving it approximately the same number of weights as the DBLSTM networks. Retraining with weight noise was not found to be effective for the DBRNN, and the results are only quoted without noise. The best result of 17.99 $\pm$ 0.13 is not significantly different from the best transducer result, which is the best TIMIT result we know of in the literature. The DBLSTM result without weight noise is better than the CTC result without noise, and the DBRNN hybrid result is much better than the DBRNN CTC result of 37.6 quoted in [1].

Table 1. TIMIT Results with End-To-End Training.

TRAINING METHOD	DEV PER	TEST PER
CTC	19.05 ± 0.11	21.57 ± 0.25
CTC (NOISE)	16.34 ± 0.07	18.63 ± 0.16
TRANSDUCER	15.97 ± 0.28	$\mathbf{18.07 \pm 0.24}$

Table 2. TIMIT Results with Hybrid Training.

NETWORK	DEV PER TEST PER	DEV FER TEST FER	DEV CE TEST CE
DBRNN	19.91 ± 0.22 21.92 ± 0.35	30.82 ± 0.31 31.91 ± 0.47	1.07 ± 0.010 1.12 ± 0.014
DBLSTM	17.44 ± 0.156 19.34 ± 0.15	28.43 ± 0.14 29.55 ± 0.31	0.93 ± 0.011 0.98 ± 0.019
DBLSTM (NOISE)	16.11 ± 0.15 $\mathbf{17.99 \pm 0.13}$	26.64 ± 0.08 $\mathbf{27.88 \pm 0.16}$	0.88 ± 0.008 $\mathbf{0.93 \pm 0.004}$

5. WALL STREET JOURNAL EXPERIMENTS

The second set of experiments were carried out on the Wall Street Journal (WSJ) speech corpus. Their main purpose was to gauge the suitability of hybrid DBLSTM-HMM for large vocabulary speech recognition, and in particular to compare the approach with existing deep network and GMM benchmarks.

We trained an sGMM-HMM baseline system on WSJ corpus (available as LDC corpus LDC93S6B and LDC94S13B) using Kaldi recipe s5 [20]. The training set used for the experiments was the 14hour subset train-si84, rather than the full 81 hour set. We used the dataset test-dev93 as the development set. The audio data was preprocessed into 40 dimensional log mel filter-banks, with deltas and accelerations, as with TIMIT. The trigram language model used for the task was provided with the WSJ CD. The forced alignments were generated from Kaldi recipe tri4b, corresponding to LDA preprocessing of data, with MMLT and SAT for adaptation. See Kaldi recipe s5 for further details. There were a total 3385 triphone states in the alignments.

The DBLSTM network had five bidirectional hidden levels, with 500 LSTM cells in each of the forward and backward layers, and a size 3385 softmax output layer, giving a total of 29.9M weights. The training parameters for the DBLSTM network were identical to those used for TIMIT. The deep network (DNN) had a context window of 15 acoustic frames (seven to either side of the centre frame being classified) It had six hidden layers with 2000 sigmoidal units in each, and a size 3385 softmax output layer. The DNN weights were

Table 3. WSJ Results. All results recorded on the dev93 evaluation set. 'WER' is word error rate, 'FER' is frame error rate and 'CE' is cross entropy error in nats per frame.

SYSTEM	WER	FER	CE
DBLSTM	**11.7**	30.0	1.15
DBLSTM (NOISE)	12.0	**28.2**	**1.12**
DNN	12.3	44.6	1.68
SGMM [20]	13.1	–	–

initialized with samples from a mean of 0, standard deviation 0.067 Gaussian. The DNN was trained with stochastic gradient descent, starting with a learning rate of 0.1, and momentum of 0.9. The learning rate was reduced by a factor of 2 at the end of each epoch which failed to produce an improved WER over the previous epoch, on the development set. After six failed attempts, the learning rate was deemed to be low enough that no further annealing was performed. The network was trained for a total of 30 epochs. The posterior probabilities returned by the DNN and DBLSTM were not divided by state priors during decoding.

The DBLSTM networks outperformed both the GMM baseline and the DNN. However the improvement in word error over the DNN was much smaller than the gains in cross entropy error and frame error rate. Furthermore, retraining DBLSTM with noise decreased the cross entropy and FER, but *increased the WER*. The DNN was not pretrained as a Deep Belief Network [6], which may have considerably improved its performance.

6. DISCUSSION

The two sets of experiments in this paper deliver a mixed message about DBLSTM-HMM hybrids. On one hand the TIMIT results show that they can deliver results on a par with end-to-end discriminatively trained DBLSTM, and substantially better than the best deep networks. On the other hand, the Wall Street Journal results suggest that DBLSTM does not perform much better than a randomly initialised deep network, when used as an acoustic model in a hybrid system for large vocabulary speech recognition.

The picture is further complicated by the fact that DBLSTM is much better than deep networks at optimising the objective function used for training (cross entropy error). This is not entirely surprising, since DBLSTM is able to make more use of surrounding context than deep networks, and also able to ensure that it makes similar predictions for consecutive frames (unlike deep networks, where each prediction is made independently). It is therefore difficult to know how to improve the performance of DBLSTM within a hybrid system. Indeed adding a regularisation term reduced the Wall Street Journal cross entropy error still further, but made the word error rate worse.

The fundamental problem is that the frame-level distributions the network is trained to optimise are significantly different from the sequence level distribution that is implicitly defined by the decoding lattice. First of all, the network is trained to model the location of the state-segment boundaries, which may be inaccurate (if the forced alignment used to define them is suboptimal) and which is anyway irrelevant to the word-level distributions defined by the lattice (because the lattice sums over all feasible alignments). Secondly, the cross entropy error does not take into account the language model.

We suspect that the latter effect was the main reason for the disparity betwen cross entropy and word error for DBLSTM on Wall Street Journal. Given that DBLSTM can access context from across the whole sequence, it may be able to learn an implicit word-level language model from the training data which then interferes with the explicit language model used during decoding. For a task like TIMIT, with a weak, biphone language model estimated from the training utterances, the interference is likely to be negligible. For Wall Street Journal, where the language model is trained on a large text corpus and has a profound impact on overall performance, it may be more significant.

7. CONCLUSIONS AND FUTURE WORK

We have applied a hybrid HMM-Deep Bidirectional LSTM system to the TIMIT and Wall Street Journal speech databases. We found that the system gave state-of-the-art results on TIMIT, and outperformed GMM and deep network benchmarks on WSJ.

In the future we would like to apply the system to large vocabulary recognition on corpora with spontaneous speech, such as Switchboard, where the language model is likely to play a less significant role. We would also like to investigate full sequence training as a means of reducing the disparity between the objective function used for training and the performance during decoding.

8. REFERENCES

[1] A. Graves, A. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *Proc ICASSP 2013*, Vancouver, Canada, May 2013.

[2] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist Temporal Classification: Labelling Unsegmented Sequence Data with Recurrent Neural Networks," in *ICML*, Pittsburgh, USA, 2006.

[3] A. Graves, "Sequence transduction with recurrent neural networks," in *ICML Representation Learning Worksop*, 2012.

[4] H.A. Bourlard and N. Morgan, *Connnectionist Speech Recognition: A Hybrid Approach*, Kluwer Academic Publishers, 1994.

[5] Qifeng Zhu, Barry Chen, Nelson Morgan, and Andreas Stolcke, "Tandem connectionist feature extraction for conversational speech recognition," in *International Conference on Machine Learning for Multimodal Interaction*, Berlin, Heidelberg, 2005, MLMI'04, pp. 223–231, Springer-Verlag.

[6] A. 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, jan. 2012.

[7] G. Hinton, Li Deng, Dong 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," *Signal Processing Magazine, IEEE*, vol. 29, no. 6, pp. 82 –97, nov. 2012.

[8] A. J. Robinson, "An Application of Recurrent Nets to Phone Probability Estimation," *IEEE Transactions on Neural Networks*, vol. 5, no. 2, pp. 298–305, 1994.

[9] Oriol Vinyals, Suman Ravuri, and Daniel Povey, "Revisiting Recurrent Neural Networks for Robust ASR," in *ICASSP*, 2012.

[10] A. Maas, Q. Le, T. O'Neil, O. Vinyals, P. Nguyen, and A. Ng, "Recurrent neural networks for noise reduction in robust asr," in *INTERSPEECH*, 2012.

[11] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[12] F. Gers, N. Schraudolph, and J. Schmidhuber, "Learning Precise Timing with LSTM Recurrent Networks," *Journal of Machine Learning Research*, vol. 3, pp. 115–143, 2002.

[13] M. Schuster and K. K. Paliwal, "Bidirectional Recurrent Neural Networks," *IEEE Transactions on Signal Processing*, vol. 45, pp. 2673–2681, 1997.

[14] A. Graves and J. Schmidhuber, "Framewise Phoneme Classification with Bidirectional LSTM and Other Neural Network Architectures," *Neural Networks*, vol. 18, no. 5-6, pp. 602–610, June/July 2005.

[15] Kam-Chuen Jim, C.L. Giles, and B.G. Horne, "An analysis of noise in recurrent neural networks: convergence and generalization," *Neural Networks, IEEE Transactions on*, vol. 7, no. 6, pp. 1424 –1438, nov 1996.

[16] Geoffrey E. Hinton and Drew van Camp, "Keeping the neural networks simple by minimizing the description length of the weights," in *COLT*, 1993, pp. 5–13.

[17] A. Graves, "Practical variational inference for neural networks," in *NIPS*, pp. 2348–2356. 2011.

[18] DARPA-ISTO, *The DARPA TIMIT Acoustic-Phonetic Continuous Speech Corpus (TIMIT)*, speech disc cd1-1.1 edition, 1990.

[19] Kai fu Lee and Hsiao wuen Hon, "Speaker-independent phone recognition using hidden markov models," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 1989.

[20] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, J. Silovsky, G. Stemmer, and K. Vesely, "The kaldi speech recognition toolkit," in *IEEE 2011 Workshop on Automatic Speech Recognition and Understanding*. Dec. 2011, IEEE Signal Processing Society.