INVESTIGATING BIDIRECTIONAL RECURRENT NEURAL NETWORK LANGUAGE MODELS FOR SPEECH RECOGNITION

X. Chen[1], A. Ragni[1], X. Liu[2], M.J.F. Gales[1]

University of Cambridge Engineering Department [1], Chinese University of Hong Kong [2]
{xc257,ar527,xl207,mjfg}@eng.cam.ac.uk

ABSTRACT

Recurrent neural network language models (RNNLMs) are powerful language modeling techniques. Significant performance improvements have been reported in a range of tasks including speech recognition compared to n-gram language models. Conventional n-gram and neural network language models are trained to predict the probability of the next word given its preceding context history. In contrast, bidirectional recurrent neural network based language models consider the context from future words as well. This complicates the inference process, but has theoretical benefits for tasks such as speech recognition as additional context information can be used. However to date, very limited or no gains in speech recognition performance have been reported with this form of model. This paper examines the issues of training bidirectional recurrent neural network language models (bi-RNNLMs) for speech recognition. A bi-RNNLM probability smoothing technique is proposed, that addresses the very sharp posteriors that are often observed in these models. The performance of the bi-RNNLMs is evaluated on three speech recognition tasks: broadcast news; meeting transcription (AMI); and low-resource systems (Babel data). On all tasks gains are observed by applying the smoothing technique to the bi-RNNLM. In addition consistent performance gains can be obtained by combining bi-RNNLMs with n-gram and uni-directional RNNLMs.

***Index Terms*—** language model, bidirectional recurrent neural network, speech recognition, interpolation

1. INTRODUCTION

Language models (LM) are crucial components in many application areas including speech recognition. They aim to estimate the probability of any given word sequence $\mathcal{W} =< w_1, w_2, ..., w_L >$. The sequence probability can be computed using

$$P(\mathcal{W}) = P(w_1, w_2, ..., w_L) = \prod_{t=1}^{L} P(w_t|w_{t-1}, ..., w_1) \quad (1)$$

This research was partly funded under the ALTA Institute, University of Cambridge. Thanks to Cambridge English, University of Cambridge, for supporting this research. This work was also supported by the Intelligence Advanced Research Projects Activity (IARPA) via Department of Defense U. S. Army Research Laboratory (DoD/ARL) contract number W911NF-12-C-0012. The U. S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. Disclaimer: The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of IARPA, DoD/ARL, or the U. S. Government. Xunying Liu is funded by MSRA grant no. 6904412 and CUHK grant no. 4055065

The task of language model then becomes that of calculating the probability of word w_t given its previous words $w_1^{t-1} =< w_1, ..., w_{t-1} >$. A variety of statistical language models have been proposed to compute $P(w_t|w_1^{t-1})$, including n-gram LMs [1] and neural network LMs [2, 3]. n-gram LMs have been the dominant language modeling approach for several decades due to good performance and efficient implementation. Recently, significant improvements have been reported with recurrent neural network (RNN) LMs over standard n-gram LMs in many fields including speech recognition [4, 5, 6, 7]. Long short-term memory (LSTM) based LMs [8] can further improve performance by handling gradient vanishing issue existed in sigmoid activation RNNLMs.

More recently, bidirectional RNNs (bi-RNNs) [9] have outperformed unidirectional RNNs (uni-RNNs) in application areas ranging from acoustic modeling [10] to machine translation [11]. Bi-RNNs incorporate both the previous and future information to improve prediction. However to date, in the field of language modeling, very limited or no gains in speech recognition performance have been reported with bi-RNNLMs over uni-RNNLMs. Several alternative approaches have attempted to use succeeding words for language modeling [12, 13, 14, 15]. The backward RNNLMs were interpolated with the forward RNNLMs for speech recognition in [12] and [15]. [14] investigated the training of bi-RNNLMs using noise contrastive estimation for NLP tasks. [13] applied bi-RNNLMs on a broadcast news transcription task and it was reported that sigmoid bi-RNNLMs gave small gains, while no improvements were obtained from bi-LSTM LMs. This paper investigates the use of bi-RNNLMs for speech recognition, following the work in [13]. The issues in training these bidirectional models is discussed, and possible combination approaches. This paper further investigates the use of these bi-directional RNNLMs, showing that by appropriately normalising the language posteriors of these models and combining them with other state-of-art language models consistent gains can be obtained.

This paper is organised as: Section 2 gives a brief review of RNNLMs, including unidirectional and bidirectional RNNLMs. The interpolation of RNNLMs and n-gram LMs is discussed in Section 3, followed by the proposed smoothing method for bidirectional RNNLMs in Section 4. Experimental results are presented in Section 5 and conclusion drawn in Section 6.

2. RECURRENT NEURAL NETWORK LMS

Traditionally, language models are trained to predict the word probability based on the current history, $P(w_t|w_1^{t-1})$. For these form of model, whether an n-gram or uni-directional RNNLM, the probability of word sequence $\mathcal{W}$ can then be computed using Equation (1). This form of language model will be referred to as unidirectional LMs (uni-LMs).

Incorporating future information into the prediction of the current word complicates the calculation of the sentence probability. The simple decomposition shown in Equation (1) cannot be used with these bidirectional LMs (bi-LMs). The probability of a word now takes the form $P(w_t|w_1^{t-1}, w_{t+1}^L)$. To address this issues, the individual word probabilities are combined within a Product of Experts (PoE) framework to yield the total sentence probability. Thus

$$P_{bi}(\mathcal{W}) = \frac{1}{Z_{bi}} \prod_{i=1}^{L} P(w_t|w_1^{t-1}, w_{t+1}^L) = \frac{1}{Z_{bi}} \hat{P}_{bi}(\mathcal{W}) \quad (2)$$

where $\hat{P}_{bi}(\mathcal{W})$ is the product of word probabilities from bi-RNNLMs over sequence $\mathcal{W}$ and Z_{bi} is a sentence-level normalisation term,

$$Z_{bi} = \sum_{\mathcal{W} \in \Theta} \hat{P}_{bi}(\mathcal{W}) \quad (3)$$

where Θ denotes the set of all possible word sequences. Unfortunately, it is impractical to calculate Z_{bi}, complicating the calculation of perplexity for bi-LMs.

2.1. Unidirectional RNNLMs

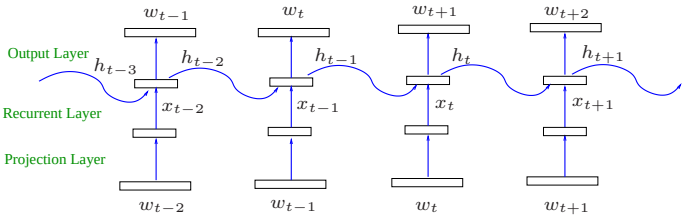

Fig. 1. *An example unidirectional RNNLM.*

Figure 1 shows a typical unidirectional RNNLM. The operation of this model is as follows. First, each word in the input layer is projected to a low-dimensional, continuous, space via a linear projection layer. This projected, word vector x_{t-1} is then combined with the history vector h_{t-2}, which represents the word history w_1^{t-2} to form a new history vector h_{t-2}. This is then fed to a softmax function to yield the probability distribution over the word at time t. Thus the prediction of the current word, w_t, is dependent on a representation of the complete history of words w_1^{t-1}. A range of non-linear functions have been used in the recurrent layer. Sigmoid [3] and long short-term memory (LSTM) [16] activations are two popular choices for language modeling.

2.2. Bidirectional RNNLMs

To incorporate future information into the word prediction an additional hidden unit is incorporated into the model. This topology is shown in Figure 2. This second history vector encodes the complete future history w_{t+1}^L, and allows the bidirectional RNNLM, w_t, $P(w_t|w_1^{t-1}, w_{t+1}^L)$, to be computed.

The training of bi-RNNLMs is more complicated than uni-RNNLMs as the future information must be taken into account. In [13], all sentences in the training corpus were concatenated to form a single sequence. This sequence was then "chopped" into sub-sequences with the averaged sentence length. Bi-RNNLMs were trained with minibatch mode on GPU by processing multiple sequences. This allows bi-RNNLMs to be efficiently trained. However, consistency issues can arise when not cutting at sentence

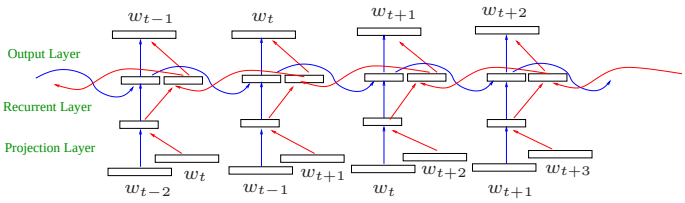

Fig. 2. *An example bidirectional RNNLMs.*

boundaries as history vectors are reset in the middle of a sentence. In this paper, the bi-RNNLMs are trained in a more consistent fashion. Multiple sentences are aligned from left to right to form minibatches during bi-RNNLM training. In order to handle issues caused by variable sentence lengths, `NULL` tokens are appended to the end of sentences to ensure that the aligned sentences have the same length. These `NULL` tokens are not used for parameter update.

Although the perplexity of bi-RNNLMs is difficult to obtain, the log-likelihood of $P(w_t|w_1^{t-1}, w_{t+1}^L)$ can still be used as objective function during training. An unnormalised, "pseudo", PPL can also be calculated by using $P(w_t|w_1^{t-1}, w_{t+1}^L)$ in a similar fashion to the uni-LMs. However, the pseudo PPL of bi-RNNLMs cannot be directly compared with PPL of uni-LMs as the unnormalised probability is used in bi-RNNLMs. Note, in this paper, (uni- and bi-) RNNLMs with an unclustered, full output layer are trained efficiently on GPU [17] with a modified version of the CUED-RNNLM toolkit [18].

3. INTERPOLATION OF RNNLMS AND N-GRAM LMS

n-gram and uni-RNN LMs have different, complementary, modeling ability [19]. Improved performance is possible by interpolating these two models together. The same should be true for bi-LMs. In this section, two possible interpolation methods for bi-LMs are briefly described.

3.1. Linear Interpolation

linear interpolation of two uni-LMs (e.g. n-gram and RNN LMs) is commonly used. Here

$$P(w_t|w_1^{t-1}) = \lambda P_{rnn}(w_t|w_1^{t-1}) + (1-\lambda) P_{ng}(w_t|w_1^{t-1}) \quad (4)$$

where λ is the interpolation weight of RNNLM. The resulting interpolated probability is a valid probability mass function (PMF) and can be to calculate sequence, sentence, probabilities. Applying the same approach to bi-LMs would yield

$$P(w_t|w_1^{t-1}, w_{t+1}^L) = \quad (5)$$
$$\lambda P_{uni}(w_t|w_1^{t-1}) + \frac{1}{Z_{bi}}(1-\lambda) P_{bi}(w_t|w_1^{t-1}, w_{t+1}^L)$$

As previously discussed it is not practical to compute the normalisation term Z_{bi}. Thus linear interpolation is challenging for bi-LMs.

3.2. Log-linear Interpolation

An alternative approach is to apply linear interpolation in the log domain, log-linear interpolation [20]. For two uni-LMs this yields

$$P(w_t|w_1^{t-1}) = \quad (6)$$
$$\frac{1}{Z(w_1^{t-1})} P_{rnn}(w_t|w_1^{t-1})^{\lambda} P_{ng}(w_t|w_1^{t-1})^{1-\lambda}$$

where $Z(w_1^{t-1})$ is the history-dependent normalisation term. It can be computed by summing over the vocabulary $\mathcal{V}$,

$$Z(w_1^{t-1}) = \sum_{w \in \mathcal{V}} P_{rnn}(w|w_1^{t-1})^{\lambda} P_{ng}(w|w_1^{t-1})^{1-\lambda} \quad (7)$$

Log-linear model combination with bi-LMs is again more complicated due to the need to compute normalisation terms with future words. To address this the models can be combined at the word sequence $\hat{\mathcal{W}}$ level.Thus considering a uni-LM and bi-LM

$$\begin{aligned} P(\mathcal{W}) &= \frac{1}{Z} P_{uni}(\mathcal{W})^{\lambda} P_{bi}(\mathcal{W})^{1-\lambda} \\ &= \frac{1}{\bar{Z}} P_{uni}(\mathcal{W})^{\lambda} \hat{P}_{bi}(\mathcal{W})^{1-\lambda} \end{aligned} \quad (8)$$

where $\bar{Z}$ is the sentence-level normalisation term and $\hat{P}_{bi}(\mathcal{W})$ is defined in Equation (2). The log of LM probabilities are normally used in speech recognition, thus Equation 8 becomes

$$\log P(\mathcal{W}) = C + \lambda \log P_{uni}(\mathcal{W}) + (1-\lambda) \log \hat{P}_{bi}(\mathcal{W}) \quad (9)$$

where C is a constant and does not alter the rank ordering of hypotheses. Thus, sentence level log-linear interpolation of uni-LMs and bi-LMs is valid, and the unnormalised form (with C) used for speech recognition. Hence, though the performance of bi-RNNLMs cannot be evaluated using perplexity, it can be evaluated with WER. It is also worth noting that the sentence level log-linear interpolation can be re-expressed as word level log-linear interpolation of the two models.

4. BIDIRECTIONAL RNNLMS PROBABILITY SMOOTHING

To investigate the performance of bi-RNNLMs, an initial contrast of the pseudo-perplexity of the bi-RNNLM was compared to the perplexity of a uni-RNNLM, both trained on the AMI data. Table 1 shows the results. Though the two numbers are not directly comparable, the dramatically lower pseudo-perplexity of the bi-RNNLM indicates that the individual predicted word probabilities of the bi-RNNLM are much higher than that of uni-RNNLM. Thus the predicted word probability distribution of bi-RNNLMs is expected to much sharper (lower entropy) than that of uni-RNNLMs. A similar trend was also observed on other tasks.

RNNLM	Dir.	(Pseudo) PPL
sigm	uni	85.2
	bi	27.8

Table 1. (Pseudo) PPLs of uni-RNN and bi-RNN LMs on AMI data

It is interesting to consider the impact of having a significantly lower entropy on the word predictions of the bi-RNNLM. The held-out evaluation data will not have exactly the same form as the training data. Furthermore, speech recognition systems will make errors, for example due to unusual acoustic data or word sequence. The bi-RNLLM will give low probability to these error-full, unusual, words sequences. In standard systems this issue would be addressed by optimising the language model scale factor. An alternative approach adopted here is to introduce an additional tunable parameter, α to smooth the probability distribution. Thus word probability is

$$P(w_i|w_1^{t-1}, w_{t+1}^{L}) = \frac{\exp(\alpha y_i)}{\sum_j^{\mathcal{V}} \exp(\alpha y_j)} \quad (10)$$

where y_i is the activation before softmax function in the output layer. α is set empirically set after training the bi-RNNLM as previously The output distribution is smoothed and flattened when α is less than 1. In this paper, α was set to 0.7 for all tasks.

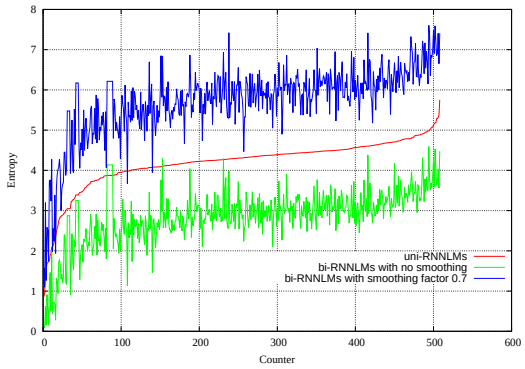

Fig. 3. *Entropy of uni-RNN, bi-RNN and smoothed bi-RNN LMs on AMI. The counter is ordered by the entropy values of uni-RNNLM.*

To further illustrate the sharpness, low entropy, of the individual word probabilities, the average sentence word-prediction entropy was computed for the for uni-RNNLM, bi-RNNLM and smoothed bi-RNNLM ($\alpha = 0.7$) on held out data from the AMI corpus. These averages are shown in Figure 3. To help interpretability the sentences are ordered by the average entropy of the uni-RNNLM. All plots follow the same general trend. The unsmoothed bi-RNNLM has a significantly lower average entropy than the other systems. It is interesting that the operating point ($\alpha = 0.7$) has a higher average entropy than the uni-RNNLM. This is not unsurprising as combining models in a PoE framework (to yield the bi-RNNLM sentence probabilities) is known to "over-sharpen" the distributions.

5. EXPERIMENTS

The performance of the bi-RNNLMs was evaluated on three corpora: broadcast news (BN); AMI meeting data (multiple distant microphone (MDM) configuration); and Dholuo (IARPA-babel403b-v1.0b) from the Babel project. A DNN-based hybrid system with sequence training [21] was built for BN; joint decoding of Tandem and DNN-HMM systems [22] were used on the AMI and Babel tasks. CMLLR-based speaker adaptation were applied for all tasks. The vocabularies of BN, AMI and Babel are 59K, 41K and 18K respectively. The sizes of words for n-gram LM training in BN, AMI and Babel are 1.5G, 2G and 467K; the number of word for RNNLMs training are 15M, 2.4M and 467K. A 4-gram LM was trained for AMI and 3-gram LMs for BN and Babel. All RNNLMs were trained with a modified version of CUED-RNNLM Toolkit [18]. For uni-RNNLMs, the hidden layer sizes of BN, AMI and Babel corpora were 512, 256 and 100 hidden, while for bi-RNNLMs, 256, 128 and 50 hidden nodes were used for each direction. In this paper, linear interpolation was used between n-gram LMs and uni-RNNLMs[1], and log-linear interpolation between uni-LMs and bi-LMs.

It is not easy to do lattice rescoring for bi-RNNLMs, though it is feasible for uni-RNNLMs as shown in [23]. For simplicity, 100-best rescoring was used for speech recognition for all RNNLMs. The oracle WERs of 100-best list for the tasks are shown in Table 2.

[1]log-linear interpolation did not outperform linear interpolation according to our experiments for uni-LMs

WER	BN		AMI		Babel
	dev	eval	dev	eval	
1-best	12.8	11.8	30.4	31.0	47.9
100-best oracle	10.0	9.5	17.7	17.7	32.6

Table 2. Oracle WERs of 100-best list on three corpora

Task	RNNLM	PPL	WER	
			dev	eval
BN	—	151.9	12.8	11.8
	sigm	112.8	12.1	11.2
	LSTM	104.4	11.9	11.0
AMI	—	182.1	30.4	31.0
	sigm	143.3	29.1	29.6
Babel	—	115.3	47.9	—
	sigm	81.0	46.6	—

Table 3. PPL and WER results of uni-RNNLMs on three tasks

Table 3 shows the PPL and WER results for n-gram LMs and uni-RNNLMs on the tasks. In addition to the baseline sigmoid-based RNNLMs, the performance of an LSTM-based system on the BN task is shown. Note more training data was available for BN compared to AMI and Babel where no gains over the sigmoid-based baseline were obtained. For these baseline, uni-LM, systems, linear interpolation was used ($\lambda = 0.5$). The uni-RNNLMs shows significant gains in terms of both PPLs and WERs. The LSTM LM yields moderate additional gains over the sigmoid RNNLM on BN.

Task	RNNLM	Dir.	WER	
			dev	eval
BN	sigm	uni	12.1	11.2
		bi	12.3	11.3
		uni+bi	11.7	10.8
	LSTM	uni	11.9	11.0
		bi	11.8	11.0
		uni+bi	11.5	10.5
AMI	sigm	uni	29.1	29.6
		bi	29.7	30.2
		uni+bi	29.0	29.6
Babel	sigm	uni	46.6	—
		bi	47.5	—
		uni+bi	46.5	—

Table 4. WERs of bi-RNNLM on three tasks

Table 4 shows WER results of bi-RNNLMs, compared to the uni-RNNLM: "uni" is the baseline for linear interpolation of n-gram and uni-RNN LMs from Table 3; "bi" indicates log-linear interpolation of n-gram and bi-RNN LMs ($\lambda = 0.5$)); and "uni+bi" indicates n-gram LM is linearly interpolated with the uni-RNNLM followed by log-linearly interpolated with the bi-RNNLM (the interpolation weights of n-gram, uni-RNN, bi-RNN were 0.35, 0.35 and 0.3 respectively). From Table 4, bi-RNNLMs yield comparable performance to uni-RNNLMs on the BN data. This is consistent with the observations in [13]. The combination of n-gram LMs, uni-RNNLMs and bi-RNNLMs yields 0.4%-0.5% WER reduction over the baseline uni-RNNLMs. However, for the other tasks, AMI and Babel, which have higher WERs, only marginal improvements were obtained. One possible reason is the sensitivity of the word probability predictions to errors when no bi-RNNLM smoothing is applied.

Task	RNNLM	Dir.	WER	
			dev	eval
BN	sigm	uni	12.1	11.2
		bi	12.1	11.1
		uni+bi	11.8	10.9
	LSTM	uni	11.9	11.0
		bi	11.9	10.9
		uni+bi	11.6	10.6
AMI	sigm	uni	29.1	29.6
		bi	29.0	29.6
		uni+bi	28.5	29.2
Babel	sigm	uni	46.6	—
		bi	46.5	—
		uni+bi	46.2	—

Table 5. WERs of smoothed bi-RNNLM on three tasks

The final experiments examine the impact of bi-RNNLM smoothing ($\alpha = 0.7$), discussed in Section 4, on performance. The results can be found in Table 5. Using this simple smoothing approach, consistent and significant performance gain can be achieved on all three tasks by using bi-RNNLMs. Compared to the baseline system (n-gram + uni-RNN LMs), bi-RNNLMs provide an additional 0.4% to 0.6% WER reduction. It is also worth noting that a fixed language model scale was used for each task in all experiments. Tuning the language model scale factor did not alter the performance. Note the use of smoothing with uni-RNNLMs, despite optimising α, yielded no performance gains.

6. CONCLUSION

In this paper, bidirectional RNNLMs are investigated for speech recognition. Compared to previous work [13], where the bi-RNNLMs were examined on a broadcast news transcription task with a low word error rate and combined with n-gram LMs only, a more complete study was performed: three corpora with different WERs were investigated; and the combination of bi-RNNLMs, n-gram and uni-RNN LMs was investigated. Bi-RNNLMs yielded good performance improvements on broadcast data, which has a relatively low WER. However, for the other two corpora which have higher WERs, only marginal improvements were obtained. In order to overcome the sensitivity issue of bi-RNNLMs to recognition errors, a simple smoothing method was proposed. Consistent and significant WER improvements can be obtained on the all three tasks.

It is clear that succeeding words are helpful to improve speech recognition performance. However, there are issues that need to be addressed to make these forms of bi-directional models generally applicable: better distribution smoothing approaches; improved combination schemes; more efficient training approaches; and lattice rescoring techniques. These will be explored in future work.

7. ACKNOWLEDGEMENT

The authors would like to thank Ebru Arisoy, Abhinav Sethy and Bhuvana Ramabhadran from IBM for discussions and valuable suggestions on bidirectional RNNLMs.

8. REFERENCES

[1] Roni Rosenfield, "Two decades of statistical language modeling: Where do we go from here?," *Proceedings of the IEEE*, 2000.

[2] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin, "A neural probabilistic language model," Journal of Machine Learning Research, vol. 3, pp. 1137–1155, 2003.

[3] Tomas Mikolov, Martin Karafiát, Lukas Burget, Jan Cernockỳ, and Sanjeev Khudanpur, "Recurrent neural network based language model.," in *Proc. ISCA INTERSPEECH*, 2010.

[4] Tomas Mikolov, Stefan Kombrink, Lukas Burget, J.H. Cernocky, and Sanjeev Khudanpur, "Extensions of recurrent neural network language model," in *Proc. ICASSP*. IEEE, 2011.

[5] Jacob Devlin, Rabih Zbib, Zhongqiang Huang, Thomas Lamar, Richard Schwartz, and John Makhoul, "Fast and robust neural network joint models for statistical machine translation," in *Proc. ACL*, 2014.

[6] Wim De Mulder, Steven Bethard, and Marie-Francine Moens, "A survey on the application of recurrent neural networks to statistical language modeling," *Computer Speech & Language*, vol. 30, no. 1, pp. 61–98, 2015.

[7] Stefan Kombrink, Tomas Mikolov, Martin Karafiat, and Lukas Burget, "Recurrent neural network based language modeling in meeting recognition.," in *Proc. ISCA INTERSPEECH*, 2011.

[8] Martin Sundermeyer, Ralf Schlüter, and Hermann Ney, "LSTM neural networks for language modeling," in *Proc. ISCA INTERSPEECH*, 2012.

[9] Mike Schuster and Kuldip K Paliwal, "Bidirectional recurrent neural networks," *Signal Processing, IEEE Transactions on*, vol. 45, no. 11, pp. 2673–2681, 1997.

[10] Hasim Sak, Andrew Senior, and Françoise Beaufays, "Long short-term memory recurrent neural network architectures for large scale acoustic modeling," in *Proc. ICSA INTERSPEECH*, 2014.

[11] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.

[12] Yangyang Shi, Martha Larson, Pascal Wiggers, and Catholijn Jonker, "Exploiting the succeeding words in recurrent neural network language models.," in *Proc. ICSA INTERSPEECH*, 2013.

[13] Ebru Arisoy, Abhinav Sethy, Bhuvana Ramabhadran, and Stanley Chen, "Bidirectional recurrent neural network language models for automatic speech recognition," in *Proc. ICASSP*. IEEE, 2015, pp. 5421–5425.

[14] Tianxing He, Yu Zhang, Jasha Droppo, and Kai Yu, "On training bi-directional neural network language model with noise contrastive estimation," *arXiv preprint arXiv:1602.06064*, 2016.

[15] W Xiong, J Droppo, X Huang, F Seide, M Seltzer, A Stolcke, D Yu, and G Zweig, "The Microsoft 2016 conversational speech recognition system," *Proc. ICASSP*, 2017.

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

[17] Xie Chen, Xunying Liu, Yongqiang Wang, Mark Gales, and Phil Woodland, "Efficient training and evaluation of recurrent neural network language models for automatic speech recognition," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 2016.

[18] Xie Chen, Xunying Liu, Mark Gales, and Phil Woodland, "CUED-RNNLM an open-source toolkit for efficient training and evaluation of recurrent neural network language models," in *Proc. ICASSP*. IEEE, 2015.

[19] Xie Chen, Xunying Liu, Gales Mark, and Phil Woodland, "Investigation of back-off based interpolation between recurrent neural network and n-gram language models.," in *ASRU, IEEE Workshop on*, 2015.

[20] Alexander Gutkin, "Log-linear interpolation of language models," *Mémoire de DEA, University of Cambridge*, 2000.

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

[22] Haipeng Wang, Anton Ragni, Mark Gales, Kate Knill, Phil Woodland, and Chao Zhang, "Joint decoding of tandem and hybrid systems for improved keyword spotting on low resource languages," in *Proc. ISCA INTERSPEECH*, 2015.

[23] Xunying Liu, Xie Chen, Yongqiang Wang, Mark Gales, and Phil Woodland, "Two efficient lattice rescoring methods using recurrent neural network language models," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 24, no. 8, pp. 1438–1449, 2016.