INTERSPEECH 2016
September 8–12, 2016, San Francisco, USA

Recurrent Neural Network Language Model With Incremental Updated Context Information Generated Using Bag-of-Words Representation

Md. Akmal Haidar and Mikko Kurimo

Department of Signal Processing and Acoustics, School of Electrical Engineering
Aalto University, Finland

`md.haidar@aalto.fi, mikko.kurimo@aalto.fi`

Abstract

Recurrent neural network language model (RNNLM) is becoming popular in the state-of-the-art speech recognition systems. However, it can not remember long term patterns well due to a so-called vanishing gradient problem. Recently, Bag-of-words (BOW) representation of a word sequence is frequently used as a context feature to improve the performance of a standard feed-forward NNLM. However, the BOW features have not been shown to benefit RNNLM. In this paper, we introduce a technique using BOW features to remember long term dependencies in RNNLM by creating a context feature vector in a separate non-linear context layer during the training of RNNLM. The context information is incrementally updated based on the BOW features and processed further in a non-linear context layer. The output of this layer is used as a context feature vector and fed into the hidden and output layers of the RNNLM. Experiments with Penn Treebank corpus indicate that our approach can provide lower perplexity with fewer parameters and faster training compared to the conventional RNNLM. Moreover, we carried out speech recognition experiments with Wall Street Journal corpus and achieved lower word error rate than RNNLM.

Index Terms: speech recognition, language modeling, recurrent neural networks, bag-of-words

1. Introduction

Statistical n-gram language models (LMs) have been used successfully for speech recognition and other applications. They use local context information by modeling text as a Markovian sequence and capture only the local dependencies between words. They suffer from insufficiencies of the training data, which limit model generalization [1]. Due to limitations of the amount of training data, they encounter a data sparseness problem, which is handled by using backoff smoothing approaches with lower-order language models [2, 3]. Moreover, they cannot capture the long-range information of natural language. Several methods have been investigated to overcome this weakness. A cache-based language model is an earlier approach that is based on the idea that if a word appeared previously in a document it is more likely to occur again [4]. Recently, topic modeling approaches such as latent semantic analysis (LSA) [5], probabilistic LSA (PLSA) [6], and latent Dirichlet allocation (LDA) [7] have been used in language modeling [1, 8, 9, 10, 11].

Neural network-based language model (NNLM) is trained to predict word probabilities and has been used successfully [12, 13]. It avoids the data sparseness problem by representing words in a distributed way, as non-linear combinations of weights in a neural net. The neural net architecture might be feed-forward or recurrent. In feed-forward NNLM (FFNNLM), the recent history with a fixed length window is presented as 1-of-N encoded feature vectors in the input while computing word probabilities. Here, long-term dependency can only be captured with increasing computational cost in linear way. A recurrent NNLM (RNNLM) can capture long-term dependencies by conditioning an input from the hidden state of the previous time step. Here the input contains a word and the state of the hidden layer from the previous time step. The long-range context information are stored in the hidden layer as a memory of the model. However, the RNN is theoretically powerful and it is considered hard to train practically because of the so-called vanishing and exploding gradient problems [14, 15]. The exploding gradient problem can be avoided by a simple efficient strategy of gradient clipping [16] which allowed to train RNN models on large datasets by using simple stochastic gradient descent and backpropagation through time [17, 18, 19].

Nevertheless, the RNN suffers from the gradient vanishing problem as the gradient backpropagated in time and their magnitude shrink close to zero which makes the model to learn longer terms difficult [19]. In [20], long-term context information is captured by using a feature matrix as input into RNNLM where the context information is created by using a pre-trained topic modeling approaches [5, 7] and not learned in the recurrent model. In [19], context information is learned in a context layer during the training of RNNLM by constraining part of the recurrent matrix to close to identity. A more complex model named as long short term memory RNN (LSTM-RNN) LM [21, 22] was investigated to overcome the limitation of RNNLM, where the recurrent hidden units are replaced with LSTM cell incorporating gating units and has shown the state-of-the art results. In [23], BOW context features were applied to FFNNLM [12] and LSTM-RNN LM [22]. They improve the performance over FFNNLM and do not give any improvement over LSTM-RNN LM. However, it is unknown that whether the BOW context features can improve the performance of RNNLM. This motivates us to apply the BOW context features into RNNLM. In this paper, we have applied BOW context features into RNNLM in a different fashion than in [23] and have reported improvement over RNNLM. Here, we have developed a context feature vector in a separate context layer during the training of regular RNNLM.

The rest of this paper is organized as follows. Section 2 and 3 are used for reviewing the RNNLM and the BOW representation of a word sequence respectively. The proposed RNNLM with incremental updated context information is described in section 4. The experimental details are described in section 5. Finally the conclusions and future work are explained in section 6.

2. Recurrent neural network LM

RNNLM [24] contains an input layer, a hidden layer with a recurrent connections that allows the propagation of the previous state information of the hidden layer, an output layer, and their related weight matrices in each connections. In Figure 1, the input vector $w(t)$ encodes an input word at time t using 1-of-N encoding, also known as one-hot representation. It uses an index to each word in the vocabulary of size N and a word is encoded with 1 in its index position and all other coefficients are set to 0. The output layer produces a probability distribution over words at time t given the information in the hidden layer. The output vector $y(t)$ also has the same dimension as the input vector $w(t)$. The hidden layer stores the previous information and act as a memory of the model. The values in the hidden and output layers are calculated as:

$$h(t) = f(Uw(t) + Wh(t-1)) \tag{1}$$

$$y(t) = g(Vh(t)) \tag{2}$$

where

$$f(z) = \frac{1}{1+e^{-z}}, \quad g(z_m) = \frac{e^{z_m}}{\sum_k e^{z_k}} \tag{3}$$

U, W, and V represent the input, recurrent and output weight matrices respectively. $f(z)$ and $g(z_m)$ are the sigmoid and the soft-max function respectively. The soft-max function in the output layer confirms that the output forms a valid probability distribution. To reduce the computation in the output layer, a hierarchical soft-max approach was introduced in [25]. A simple hierarchy of two levels using frequency-based clustering was investigated in [26] and we will mention it when we use it in our experiments. The training of the RNNLM is to learn the weight matrices that maximize the likelihood of the training data. The model is trained by using stochastic gradient descent with back-propagation through time (*BPTT*) algorithm [17, 18, 27]. For further details of the RNNLM, see [24, 26].

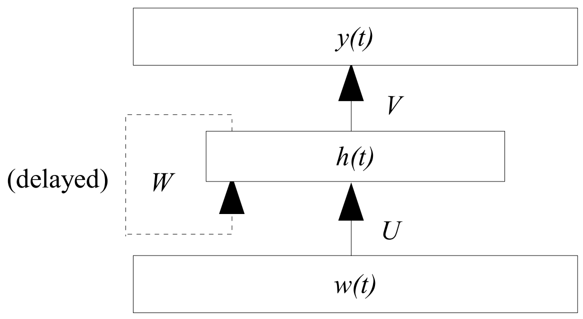

Figure 1: *Recurrent neural network (RNN).*

3. Bag-of-words (BOW) representation of a word sequence

The BOW representation of a word sequence of length L is formed by summing the 1-of-N representation of all words in the sequence. It can be described as [23]:

$$bow(t) = \sum_{i=0}^{L-1} w(t-i) \tag{4}$$

The advantage of adding BOW representation over 1-of-N representation in a FFNNLM is that it can add as many predecessor words as possible by increasing the feature size of only one word. However, BOW loses the order of the word sequence which reduces performance. An exponential decay version of BOW was employed to overcome this problem and it is defined as [23]:

$$bow_{decay}(t) = \sum_{i=0}^{L-1} \gamma^i w(t-i) \tag{5}$$

where γ is a decaying factor in [0, 1] [23]. In [23], the BOW with decay input was applied in FFNNLM [12] and have shown close perplexity results to that of the standard RNNLM [26]. However, the addition of BOW context features does not give any benefit into LSTM-RNN LM [22] and also it was not proved that whether the addition of BOW context features into RNNLM can outperform RNNLM [24, 26]. In this paper, we have applied BOW with decay representation of a sequence into RNNLM in a different fashion than in [23] and have shown improvement over RNNLM [24, 26].

4. RNNLM with incremental updated context information

Theoretically, the state of the hidden layer in the RNNLM represents the memory of the model and it stores the context information for long-time period. However, during RNNLM training, the gradients get propagated back in time, and their magnitude quickly shrink close to zero. This is known as vanishing gradient problem in RNN training which causes to learn longer term difficult [19]. In this section, we introduce context features in a separate non-linear context layer using incremental updated context information, which is formed by using a BOW with decay representation of a word sequence. We have applied context features at each time step after processing L number of words in RNNLM. At time t, we have used a sliding window for a word sequence of length L to form a BOW decay representation (Equation 5) of the sequence that represent the sentence history. Then, context information $context(t)$ for the sliding window is created which is updated incrementally at each time step with a fixed scalar parameter λ in [0, 1]. It is then applied into a non-linear hidden layer defined here as context layer $h_c(t)$. The output of the context layer is used as a context feature vector and applied into the hidden layer $h(t)$ and the output layer $y(t)$. The whole information is described in Figure 2 and we define it here as RNN-BOW LM. The values of the layers are calculated as:

$$h(t) = f(Uw(t) + Wh(t-1) + Gh_c(t)) \tag{6}$$

$$h_c(t) = f(Fcontext(t)) \tag{7}$$

$$y(t) = g(Vh(t) + Dh_c(t)) \tag{8}$$

where, U, W, V, F, G, and D are the input, recurrent, output, context information input, context feature input and context feature output weight matrices respectively. The functions $f(.)$ and $g(.)$ are defined as above. The term $w(t)$ encodes an input word using 1-of-N encoding. The context information $context(t)$ is updated as:

$$context(t) = \lambda context(t-1) + (1-\lambda) bow_{decay}(t) \tag{9}$$

The model is trained by using stochastic gradient descent with backpropagation through time (*BPTT*) algorithm [17, 18, 27].

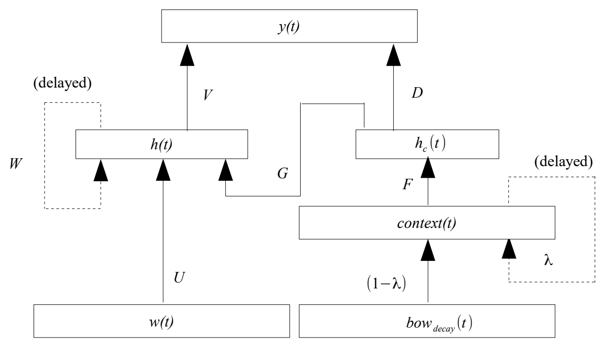

Figure 2: *RNN with incremental updated context information generated using a BOW decay representation.*

5. Experiments

5.1. Data and experimental setup

We have evaluated our approach using a well known Penn Treebank (PTB) corpus [26, 28] and Wall Street Journal (WSJ) corpus [29]. The RNNLM toolkit [30] is used to train the RNNLM and a modified version of the RNNLM toolkit is developed to train the RNN-BOW LM. The SRILM toolkit [31] and the HTK toolkit [32] are used for generating the n-gram LMs and computing the word error rate (WER) respectively. We have used the BOW decay factor γ=0.9, the length of a word sequence L=50 and λ=0.95 to update the context information incrementally throughout the experiments. First, we have reported perplexity results on PTB corpus to compare our approach with different version of the RNNLM. Later, we have performed a speech recognition experiment using WSJ corpus. We have selected one million (1 M) words from '87-89 WSJ text corpus (37 M words) and the transcription data (17 K words) of si_dt_20 folder from CSR-I (WSJ0) corpus [29] as the training and validation set respectively. The validation set is used for early stopping and to control learning rate during training [26]. The WER experiments are evaluated on the evaluation test, which is a total of 333 test utterances (5643 words) from the November 1992 ARPA CSR benchmark test data for non-verbalized vocabularies of 20K words [33, 34]. We have replaced the words that are not appeared in the vocabularies with a token <UNK>. The datasets are described in Table 1. For WER experiments, we have used the acoustic model from [35]. The acoustic model is trained by using all WSJ and TIMIT [36] training data, the 40-phone set of the CMU dictionary [37], approximately 10000 tied-states, 32 Gaussians per state and 64 Gaussians per silence state. The acoustic waveforms are parameterized into a 39-dimensional feature vector consisting of 12 cepstral coefficients plus the 0^{th} cepstral, delta and delta delta coefficients, normalized using cepstral mean subtraction ($MFCC_{0-D-A-Z}$). The cross-word models are evaluated. The values of the beam width, word insertion penalty, and the language model scale factor are 350.0, -4.0, and 15.0 respectively [1, 35].

Table 1: *Datasets*

Corpus	#Words			#Vocabulary
	Train	Valid	Test	
PTB	930 K	74 K	82 K	10 K
WSJ	1 M / 37 M	17 K	5643	17662

5.2. Results on PTB corpus

In Table 2, we have reported validation (Valid) and test perplexity (PPL) results on PTB corpus using various number of hidden and context neurons in the RNN-BOW LM. Here, We have considered hierarchical soft-max in the output layer using frequency-based classes [26]. The baseline model is obtained by an interpolated 5-gram model with modified Kneser-Ney smoothing and no count cutoffs. It is denoted here as KN5. From Table 2, we can note that our approach outperforms both the baseline KN5 model and the RNNLM. The RNN-BOW LM with context neuron size of 10 and 20 gives the best and almost similar performance. Besides, a smaller size of RNN-BOW (e.g., H=90, H_c=10) LM yields better results than the RNNLM with larger size. Moreover, the RNN-BOW LM with larger size (e.g., H=190, H_c=10, and H=280, H_c=20) shows close performance to the standard full RNNLM (123 or 124.7) [20, 26]. Furthermore, it gives approximately similar performance (e.g., H=100, H_c=20) to that of an RNN-LDA LM [20], where the RNN-LDA LM requires a pre-trained topic modeling approaches.

Table 2: *PPL results on PTB corpus using class size of 100, BPTT=4, different number of hidden (H) and context (H_c) neurons.*

Language Model	$H(H_c)$	Valid PPL	Test PPL
KN5	-	148.0	141.2
RNN	100 (-)	148.5	141.0
RNN	200 (-)	142.2	135.2
RNN	300 (-)	141.5	134.5
RNN-BOW	90 (10)	137.8	130.6
RNN-BOW	100 (10)	136.0	127.9
RNN-BOW	100 (20)	134.6	127.5
RNN-BOW	100 (30)	136.9	129.9
RNN-BOW	180 (10)	130.7	125.2
RNN-BOW	180 (20)	131.6	125.6
RNN-BOW	190 (10)	**130.2**	**124.3**
RNN-BOW	190 (20)	130.5	124.3
RNN-BOW	190 (30)	132.9	125.8
RNN-BOW	280 (20)	**130.3**	**123.4**

In Table 3, we have reported test perplexity results on PTB corpus using the full RNNLM and the full RNN-BOW LM, i.e., no hierarchical soft-max in the output layer. Here, we performed experiments for different number of *BPTT* steps. The *BPTT* algorithm is trained in block mode [30]. When *BPTT*=50, the *BPTT* block size is 50 and otherwise it is 10. From Table 3, we can note that with increasing *BPTT*, the performance of RNNLM degrades as the gradient vanishes faster [19] whereas the performance of RNN-BOW LM remains the same. Therefore, the RNN-BOW LM can overcome the vanishing gradient problem and capture the long term dependencies. Moreover, we can see that using the RNN-BOW LM, the best test perplexity is around 115, which is equivalent to the best results on PTB corpus obtained by a more complex LSTM-RNN and an structurally constrained recurrent network (SCRN) reported in [19]. The results in (+KN5) are obtained by interpolating the models with the baseline model using interpolation weight 0.5.

Table 3: *Test PPL results on PTB corpus using different number of BPTT steps, hidden (H) and context (H_c) neurons and without factorizing the output layer.*

LM	$H(H_c)$	$BPTT$	Individual	+KN5
KN5	-	-	141.2	-
RNN [26]	200 (-)	5	123	106
RNN	200 (-)	10	126.0	107.6
RNN	200 (-)	50	139.5	112.7
RNN-BOW	90 (10)	50	119.3	102.9
RNN-BOW	100 (10)	50	118.6	101.9
RNN-BOW	100 (20)	4	118.0	101.5
RNN-BOW	100 (20)	50	118.1	101.5
RNN-BOW	180 (20)	10	**115.2**	**100.0**
RNN-BOW	180 (20)	50	**115.1**	**100.0**
RNN-BOW	190 (10)	50	116.2	101.3

5.3. Complexity analysis of RNNLM and RNN-BOW LM

The time complexities of a full RNNLM and a full RNN-BOW LM for one training step respectively are proportional to [26]:

$$O_{RNN} = (1 + H) \times H \times \tau + H \times N \tag{10}$$

$$\begin{aligned} O_{RNN-BOW} = (1 + H) \times H \times \tau + H \times N + \\ L + L \times H_c + H_c \times H + H_c \times N \end{aligned} \tag{11}$$

where H is the size of the hidden layer, H_c is the size of context layer in RNN-BOW LM, N is the size of the vocabulary, and τ is the amount of *BPTT* steps. When a class layer is used in the output layer to reduce the complexity at a cost of lower performance, the size of vocabulary N in Equation 10 and 11 can be replaced by the class size C [26]. From Equation 10, 11, and Table 2, we can see that even the summation of hidden and context neurons in the RNN-BOW LM equals the hidden layer size of the RNNLM, the RNN-BOW LM requires less parameters than the RNNLM for a fixed τ. From Table 3, we can note that the RNN-BOW LM does not degrades performance with increasing τ. Therefore, from Table 2 and 3, we can conclude that the RNN-BOW LM takes less training time and parameters to outperform the RNNLM [26].

5.4. Results on WSJ corpus

We have created lattices using pruned trigram with modified Kneser-Ney (KN) smoothing, from which we have generated 100-best lists that are used in the rescoring. The baseline n-gram language model for rescoring is a modified KN 5-gram (KN5) model with no count cutoff. The RNNLM and the RNN-BOW LM models are trained using 4 *BPTT* steps. The models are interpolated with the baseline KN5 model with weight 0.75 for the RNNLM or the RNN-BOW LM and 0.25 for the baseline KN5 model. The evaluation test results on 1 M words of WSJ corpus with and without class layer in the output layer are reported in Table 4 and 5 respectively. From Table 4 and Table 5, we can note that the RNN-BOW LM gives 8.6% and 2.5% relative WER reductions over the KN5 and the RNNLM respectively. The independent models without class layer do not give any further WER reductions. However, they yield around 5% perplexity reductions over the models with class layer. Moreover, the interpolation of RNN-BOW LM with KN5 model gives 4.2% (11.8% to 11.3%) and 5.1% (11.7% to 11.1%) WER reductions with and without class layer respectively over the interpolation of RNNLM with KN5 model. Furthermore, we can see that the RNN-BOW LM with class layer gives lower WER than the RNNLM without class layer (i.e. full RNNLM). The significance improvement in WER is done by using a matched-paired t-test where the misrecognized words in each test utterance are counted. We have used the interpolated model using the full RNNLM and the full RNN-BOW LM to perform this test. The p-value of the RNN-BOW+KN5 relative to the RNN+KN5 is 0.0117. At a significance level of 0.05, the proposed RNN-BOW LM is significantly better than the RNNLM.

Table 4: *Test PPL and WER results on 1 M words of WSJ corpus using class size 100.*

Language Model	$H(H_c)$	PPL	WER
KN5	-	248.0	12.8
RNN	200 (-)	226.2	12.0
RNN-BOW	190 (10)	**218.8**	**11.7**
RNN+KN5	200 (-)	191.6	11.8
RNN-BOW+KN5	190 (10)	**183.0**	**11.3**

Table 5: *Test PPL and WER results on 1 M words of WSJ corpus without class layer in the output layer.*

Language Model	$H(H_c)$	PPL	WER
KN5	-	248.0	12.8
RNN	200 (-)	215.6	12.0
RNN-BOW	190 (10)	**207.0**	**11.7**
RNN+KN5	200 (-)	183.4	11.7
RNN-BOW+KN5	190 (10)	**176.6**	**11.1**

6. Conclusions and future work

In this paper, we introduced an RNN architecture with incremental updated context information which is formed by using a BOW decay representation of a word sequence. Our approach overcomes the difficulties of learning long term patterns in RNN which occur due to a so called vanishing gradient problem. The long term dependencies in RNNLM are captured by applying a context feature vector generated in a separate non-linear context layer during the training of RNNLM. We have formed a BOW decay representation of a word sequence using a sliding window. A context information is formed for this window which is updated incrementally. The context information is then passed into a non-linear context layer. The output of this layer is used as a context feature vector and applied into the hidden and output layers of the RNNLM. The training of RNN-BOW LM is comparable to RNNLM training with the same amount of weights. We evaluated our approach on a well known PTB corpus for perplexity evaluation. Our technique requires less training parameters and time to outperform RNNLM significantly. We also carried out 100-best rescoring experiment using WSJ corpus and reported significant WER improvements over RNNLM. For future work, we will evaluate our approach with more training data and various size of context length L. We are also interested to see the performance of RNN-BOW LM using deep context layers. Furthermore, we will apply the proposed approach into LSTM-RNN LM.

7. Acknowledgements

This work was partially funded by the Academy of Finland under the grant number 251170 (Finnish Centre of Excellence Program (2012-2017)). We acknowledge the computational resources provided by Aalto Science-IT project.

8. References

[1] M. A. Haidar and D. O'Shaughnessy, "Unsupervised language model adaptation using LDA-based mixture models and latent semantic marginals," *Computer Speech & Language*, vol. 29, pp. 20-31, 2015.

[2] S. M. Katz, "Estimation of probabilities from sparse data for the language model component of a speech recognizer," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 35, no. 3, pp. 400-401, 1987.

[3] R. Kneser and H. Ney, "Improved backing-off for m-gram language modeling," in *Proc. of IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, vol. 1, Detroit, MI, USA, May 1995, pp. 181-184.

[4] R. Kuhn and R. D. Mori, "A cache-based natural language model for speech recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, no. 6, pp. 570-583, 1990.

[5] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer and R. Harshman, "Indexing by latent semantic analysis," *Journal of the American Society for Information Science*, vol. 41, no. 6, pp. 391-407, 1990.

[6] T. Hofmann, "Probabilistic latent semantic analysis," in *Proc. of the Fifteenth Annual Conference on Uncertainty in Artificial Intelligence (UAI-99)*, Stockholm, Sweden, July 30-August 1, 1999, pp. 289-296.

[7] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent Dirichlet allocation," *Journal of Machine Learning Research*, vol. 3, pp. 993-1022, 2003.

[8] J. R. Bellegarda, "Exploiting latent semantic information in statistical language modeling," *IEEE Transactions on Speech and Audio Processing*, vol. 88, no. 8, pp. 1279-1296, 2000.

[9] D. Gildea and T. Hofmann, "Topic-based language models using EM," in *Proc. of Sixth European Conference on Speech Communication and Technology (EUROSPEECH)*, Budapest, Hungary, September 1999, pp. 2167-2170.

[10] D. Mrva and P. C. Woodland, "A PLSA-based language model for conversational telephone speech," in *Proc. of International Conference on Spoken Language Processing (ICSLP)*, Jeju Island, South Korea, March 2004, pp. 2257-2260.

[11] Y.-C. Tam and T. Schultz, "Unsupervised language model adaptation using latent semantic marginals," in *Proc. of INTERSPEECH*, Pittsburgh, Pennsylvania, September 2006, pp. 2206-2209.

[12] Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin, "A neural probabilistic language model," *Journal of Machine Learning Research*, vol. 3, pp. 1137-1155, 2003.

[13] H. Schwenk, "Continuous space language models," *Computer Speech & language*, vol. 21, no. 3, pp. 492-518, 2007.

[14] S. Hochreiter, "The vanishing gradient problem during learning recurrent neural nets and problem solutions," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* , vol. 6, no. 2, pp. 107-116, 1998.

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

[16] T. Mikolov, "Statistical language models based on neural networks," *PhD thesis, Brno University of Technology,* 2012.

[17] R. J. Williams and D. Zipser, "Gradient-based learning algorithms for recurrent networks and their computational complexity," *Backpropagation: Theory, architectures and applications*, pp. 433-486, 1995.

[18] P. J. Werbos, "Generalization of backpropagation with application to a recurrent gas market model," *Neural Networks*, vol. 1, no. 4, pp. 339-356, 1988.

[19] T. Mikolov, A. Joulin, S. Chopra, M. Mathieu and M. A. Ranzato, "Learning longer memory in recurrent neural networks," *arXiv preprint arXiv2:1412.7753*, 2015.

[20] T. Mikolov and G. Zweig, "Context dependent recurrent neural network language model," in *Proc. of IEEE Workshop on Spoken Language Technology (SLT)*, Miami, FL, USA, 2012, pp. 234-239.

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

[22] M. Sundermeyer, R. Schlüter, and H. Ney, "LSTM neural networks for language modeling," in *Proc. of INTERSPEECH*, Portland, Oregon, USA, September 2012, pp. 194-197.

[23] K. Irie, R. Schlüter, and H. Ney, "Bag-of-Words input for long history representations in neural network-based language models for speech recognition," in *Proc. of INTERSPEECH*, Dresden, Germany, September 2015, pp. 2371-2375.

[24] T. Mikolov, M. Karafiat, L. Burget, J. H. Cernocky, and S. Khudanpur, "Recurrent neural network based language model," in *Proc. of INTERSPEECH*, Makuhari, Chiba, Japan, September 2010, pp. 1045-1048.

[25] J. Goodman, "Classes for fast maximum entropy," in *Proc. of IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, Salt lake city, Utah, USA, May 2001, vol. 1, pp. 561-565.

[26] T. Mikolov, S. Kombrink, L. Burget, J. H. Cernocky, and S. Khudanpur, "Extensions of recurrent neural network language model," in *Proc. of IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, Prague, Czech Republic, May 2011, pp. 5528-5531.

[27] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning internal representations by error propagation," *Technical report, DTIC Document*, 1985.

[28] M. P. Marcus, M. A. Marcinkiewicz, and B. Santorini, "Building a large annotated corpus of english: the penn treebank," *Computational Linguistics*, vol. 19, no. 2, pp. 313-330, 1993.

[29] J. Garofolo, D. Graff, D. Paul, and D. Pallett, "CSR-I (WSJ0) complete LDC93S6A," *Linguistic Data Consortium*, Philadelphia, 1993.

[30] T Mikolov, S. Kombrink, A. Deoras, L. Burget, and J. Cernocky, "RNNLM-recurrent neural network language modeling toolkit," in *Proc. of IEEE Workshop on Automatic Speech Recognition & Understanding (ASRU)*, Hilton Waikoloa Village Resort Waikoloa, HI, USA, December 2011, pp. 196-201.

[31] A. Stolcke, "SRILM-an extensible language modeling toolkit," in *Proc. of International Conference on Spoken Language Processing (ICSLP)*, Denver, Colorado, September 2002, pp. 901-904.

[32] S. Young, P. Woodland, G. Evermann, and M. Gales, "The HTK toolkit 3.4.1," http://htk.eng.cam.ac.uk/, 2013.

[33] D. B. Paul and J. M. Baker, "The design for the wall street journal-based CSR corpus," in *Proc. of International Conference on Spoken Language Processing (ICSLP)*, Banff, Alberta, Canada, October 1992, pp. 899-902.

[34] P.C. Woodland, J.J. Odell, V. Valtchev and S.J. Young, "Large vocabulary continuous speech recognition using HTK," in *Proc. of IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, Adelaide, Australia, April 1994, pp. II:125-128.

[35] K. Vertanen, "HTK wall street journal training recipe," http://www.inference.phy.cam.ac.uk/kv227/htk/

[36] J. Garofolo, L. Lamel, W. Fisher, J. Fiscus, D. Pallett, N. Dahlgren, and V. Zue, "TIMIT acoustic-phonetic continuous speech corpus LDC93S1," *Linguistic Data Consortium*, Philadelphia, 1993.

[37] -, "The Carnegie Mellon University (CMU) pronouncing dictionary," http://www.speech.cs.cmu.edu/cgi-bin/cmudict