PARAPHRASTIC RECURRENT NEURAL NETWORK LANGUAGE MODELS

X. Liu, X. Chen, M. J. F. Gales & P. C. Woodland

Cambridge University Engineering Dept,
Trumpington St., Cambridge, CB2 1PZ U.K.
Email: {*xl207,xc257,mjfg,pcw*}*@eng.cam.ac.uk*

ABSTRACT

Recurrent neural network language models (RNNLM) have become an increasingly popular choice for state-of-the-art speech recognition systems. Linguistic factors influencing the realization of surface word sequences, for example, expressive richness, are only implicitly learned by RNNLMs. Observed sentences and their associated alternative paraphrases representing the same meaning are not explicitly related during training. In order to improve context coverage and generalization, paraphrastic RNNLMs are investigated in this paper. Multiple paraphrase variants were automatically generated and used in paraphrastic RNNLM training. Using a paraphrastic multi-level RNNLM modelling both word and phrase sequences, significant error rate reductions of 0.6% absolute and perplexity reduction of 10% relative were obtained over the baseline RNNLM on a large vocabulary conversational telephone speech recognition system trained on 2000 hours of audio and 545 million words of texts. The overall improvement over the baseline n-gram LM was increased from 8.4% to 11.6% relative.

Index Terms: recurrent neural network, language model, paraphrase, speech recognition

1. INTRODUCTION

In order to handle the data sparsity problem associated with conventional back-off n-gram language models (LMs), language modelling techniques that represent preceding history contexts in a continuous and lower dimensional vector space, such as neural network language models (NNLMs) [2, 27, 25, 19, 29, 11], can be used. NNLMs are widely used in state-of-the-art speech recognition systems due to their inherently strong generalization performance. Depending on the network architecture, NNLMs can be categorised into two major types: feedforward NNLMs [2, 27, 25, 11], which use a vector representation of preceding contexts of a fixed number of words, and recurrent NNLMs (RNNLMs) [19, 20, 29], which use a recurrent vector representation of variable length full histories. In recent years RNNLMs have been shown to give significant improvements over conventional back-off n-gram LMs and feedforward NNLMs, thus drawing increasing research interest [19, 20, 29, 5, 28, 30, 9, 10].

A crucial set of generalization patterns that all statistical language models, including RNNLMs, are expected to acquire during training, are the linguistic factors influencing the realization of surface word sequences, for example, expressive richness. Natural languages are well known for this important attribute. Multiple surface realizations that are mutually paraphrastic can be used to represent the same meaning. In order to handle such variability, both the surface word sequences and their associated alternative expressions need to be considered during model training. In conventional RNNLMs, the relationship between observed words, phrases and sentences and their paraphrase alternatives are only implicitly learned via the similarity between their respective vector space representations [22], but not explicitly used during training. This can lead to insufficient context coverage and generalization.

In order to address this issue, paraphrastic LMs, were previously proposed [14, 17] to learn the rich paraphrastic relationship between longer span syntactic structures, such as phrases, without manually deriving the associated expert semantic labelling. A phrase level generative model statistically estimated from standard text data is used to explicitly generate multiple paraphrase variants for each training data sentence. Maximizing the marginal probability of these variants produces automatically smoothed statistics that are re-distributed over multiple surface realizations. This intuitively and interpretable discounting method can be exploited by many different forms of LMs that do not explicitly model the expressive richness of natural languages. In previous research, this technique were used to improve the performance of back-off n-gram LMs [14] and feedforward NNLMs [15].

In this paper, paraphrastic recurrent neural network language models are investigated. Multiple paraphrase variants were explicitly generated and used in paraphrastic RNNLM training. The rest of the paper is organized as follows. Recurrent neural network LMs are reviewed in section 2. Paraphrastic LMs are introduced in section 3. Paraphrastic RNNLMs are proposed in section 4. In section 5 paraphrastic RNNLMs are evaluated on a state-of-the-art conversational telephone speech transcription task. Section 6 is the conclusion and future work.

2. RECURRENT NEURAL NETWORK LMS

In contrast to feedforward NNLMs, recurrent NNLMs [19] represent the full, non-truncated history $h_1^{i-1} = <w_{i-1}, \ldots, w_1>$ for word w_i using the 1-of-k encoding of the most recent preceding word w_{i-1} and a continuous vector $\boldsymbol{v}_{i-2}$ for the remaining context. For an empty history, this is initialized, for example, to a vector of all ones. The topology of the recurrent neural network used to compute LM probabilities $P_{\text{RNN}}(w_i|w_{i-1}, \boldsymbol{v}_{i-2})$ consists of three layers, as is shown in figure 1. The full history vector, obtained by concatenating the those of w_{i-1} and $\boldsymbol{v}_{i-2}$, is fed into the input layer. The hidden layer compresses the information of these two inputs and computes a new representation $\boldsymbol{v}_{i-1}$ using a sigmoid activation to

The research leading to these results was supported by EPSRC grant EP/I031022/1 (Natural Speech Technology) and DARPA under the Broad Operational Language Translation (BOLT) and RATS programs. The paper does not necessarily reflect the position or the policy of US Government and no official endorsement should be inferred. Xie Chen is supported by Toshiba Research Europe Ltd, Cambridge Research Lab.

achieve non-linearity. This is then passed to the output layer to produce normalized RNNLM probabilities using a softmax activation, as well as recursively fed back into the input layer as the "future" remaining history to compute the LM probability for the following word $P_{\text{RNN}}(w_{i+1}|w_i, \boldsymbol{v}_{i-1})$.

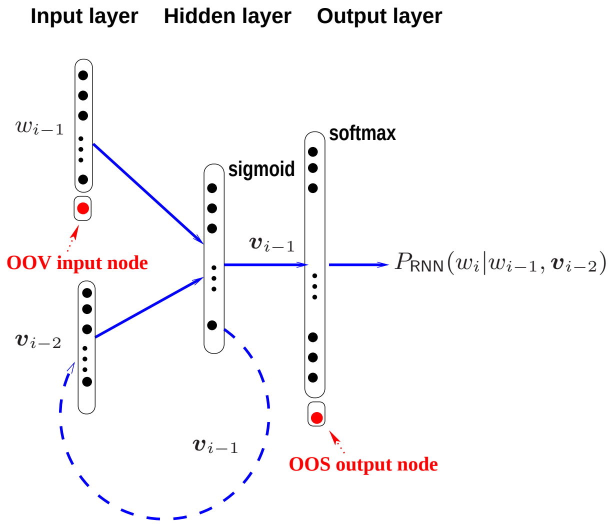

Fig. 1. *A full output layer RNNLM with OOS nodes.*

To reduce computational cost, a shortlist based output layer vocabulary limited to the most frequent words can be used. This was previously used for feedforward NNLMs [27, 6]. A similar approach may also be used at the input layer when a large vocabulary is used. In order to reduce the bias to in-shortlist words during NNLM training, two alternative network architectures that model a full vocabulary at the output layer can be considered. The first uses a class based factorized output layer structure [23]. Each word in the output layer vocabulary is attributed to a unique class.As the number of classes are normally significantly smaller than the output layer vocabulary size, training time speed-ups can be achieved for both feedforward NNLMs [23] and RNNLMs [20]. The second explicitly models the probability mass of out-of-shortlist (OOS) words using an additional output node [25, 11]. This ensures that all training data are used in training, and the probabilities of in-shortlist words are smoothed by the OOS probability mass to obtain a more robust parameter estimation. The performance sensitivity to word classing in class based RNNLMs is also removed. This form of full ouput RNNLMs with OOS nodes is used in the rest of this paper.

RNNLMs can be trained using an extended form of the standard back propagation algorithm, back propagation through time [26], where the error is propagated through recurrent connections back in time for a specific number of time steps. This allows the recurrent network to record information for several time steps in the hidden layer. Full output RNNLM can be efficiently trained in a bunch mode on GPUs [4]. A modified version of the RNNLM toolkit [21] supporting the above full output architecture with an output layer OOS node and the associated bunch mode GPU training is used.

In state-of-the-art speech recognition systems, NNLMs are often linearly interpolated with n-gram LMs to obtain both a good coverage of contexts and strong generalisation ability [27, 6, 25, 19, 29, 11]. The interpolated LM probability is given by

$$P(w_i|h_1^{i-1}) \;=\; \lambda P_{\text{NG}}(w_i|h_1^{i-1}) + (1-\lambda) P_{\text{RNN}}(w_i|h_1^{i-1}) \quad (1)$$

λ is the weight assigned to the back-off n-gram LM $P_{\text{NG}}(\cdot)$, and kept fixed as 0.5 in all experiments of this paper. In the above interpolation, the probability mass of OOS words assigned by the RNNLM component needs to be re-distributed among all OOS words [25, 11].

3. PARAPHRASTIC LANGUAGE MODELS

Paraphrastic Language Models (PLMs) [14, 17] directly target expressive richness related variability in natural languages. A statistically trained phrase level generative model is used to produce multiple paraphrase sentences for each training data sentence. Paraphrastic LM probabilities are then estimated by maximizing the marginal probability of these paraphrase variants. For an L word long sentence $\mathcal{W} \; =< \; w_1, w_2, ..., w_i, ..., w_L \; >$ in the training data, the marginal probability over all paraphrase sequences is maximized,

$$\mathcal{F}(\mathcal{W}) = \ln \left(\sum_{\boldsymbol{\psi}, \boldsymbol{\psi}', \mathcal{W}'} P(\mathcal{W}|\boldsymbol{\psi}) P(\boldsymbol{\psi}|\boldsymbol{\psi}') P(\boldsymbol{\psi}'|\mathcal{W}') P_{\text{PLM}}(\mathcal{W}') \right) \quad (2)$$

where

- $P_{\text{PLM}}(\mathcal{W}')$ is paraphrastic LM probability to be estimated.
- $P(\boldsymbol{\psi}'|\mathcal{W}')$ is a word to phrase segmentation model assigning the probability of a phrase level segmentation, $\boldsymbol{\psi}'$, given a paraphrase word sequence $\mathcal{W}'$;
- $P(\boldsymbol{\psi}|\boldsymbol{\psi}') \;=\; \prod_{v,v'} P(v|v')$ uses a phrase to phrase paraphrase model to compute probability of a phrase sequence $\boldsymbol{\psi}$ being paraphrastic to another $\boldsymbol{\psi}'$;
- $P(\mathcal{W}|\boldsymbol{\psi})$ is a phrase to word segmentation model that converts a phrase sequence $\boldsymbol{\psi}$ to a word sequence $\mathcal{W}$, and by definition is a deterministic, one-to-one mapping, thus considered non-informative.

3.1 Automatic n-gram paraphrase induction: In order to generate multiple paraphrase variants $\{\mathcal{W}'\}$, the phrase level paraphrase model $\{P(v|v')\}$ in equation (2) needs to be estimated. To obtain sufficient phrase coverage, a large number of paraphrase phrase pairs are required. As it is impractical to obtain expert semantic labelling at the phrase level, a *distributional similarity* [8] based automatic n-gram paraphrase induction algorithm proposed in [14, 17] is used. The co-occurrence counts of two phrases of variable lengths, for example, from one word to four words maximum, sharing the same left and right three word contexts, are used to estimate the phrase level paraphrase model[1]. Ambiguity can occur during word to phrase segmentation. If there is no clear reason to favor one phrase segmentation over another, $P(\boldsymbol{\psi}'|\mathcal{W}')$ can be treated as non-informative.

3.2 Generation of paraphrase variants: In order to train paraphrastic LMs, multiple paraphrase variants are required. Weighted finite state transducers (WFST) [24] can be can used to efficiently generate paraphrases [14, 17]. For each training data sentence, the paraphrase word lattice $\mathcal{T}_{\mathcal{W}'}$ is generated using a sequence of WFST composition operations as

$$\mathcal{T}_{\mathcal{W}'} \;=\; \det\left(\pi_{\mathcal{W}'}\left(\mathcal{T}_{\mathcal{W}:\mathcal{W}} \circ \mathcal{T}_{\mathcal{W}:\boldsymbol{\psi}} \circ \mathcal{T}_{\boldsymbol{\psi}:\boldsymbol{\psi}'} \circ \mathcal{T}_{\boldsymbol{\psi}':\mathcal{W}'}\right)\right) \quad (3)$$

where $\mathcal{T}_{\mathcal{W}:\mathcal{W}}$ is the transducer containing the original word sequence, $\mathcal{T}_{\mathcal{W}:\boldsymbol{\psi}}$ is the word to phrase segmentation transducer, $\mathcal{T}_{\boldsymbol{\psi}:\boldsymbol{\psi}'}$

[1] In common with other paraphrase induction methods [1, 18], this scheme can also produce phrase pairs that are non-paraphrastic, for example, antonyms. However, this is of less concern for language modelling, for which improving context coverage is the prime aim.

the phrase to phrase paraphrase transducer and $\mathcal{T}_{\boldsymbol{\psi}':\mathcal{W}'}$ the phrase to word transducer derived by taking the word to phrase transducer's inverse. $\circ$, $\det(\cdot)$ and $\pi(\cdot)$ denote the WFST composition, determinization and projection operations. It is possible in general that some phrases may have no suitable paraphrases available. In order to ensure the resulting paraphrase lattice is fully connected, self-reflexive arcs that map the input phrases to the same output are also included in the paraphrase transducer with zero cost.

In order to deweight the statistics accumulated from very unlikely paraphrase sequences and improve efficiency, a standard bigram LM trained on the surface word sequence can be applied to the paraphrase lattices generated using equation (3). Using this WFST based decoding approach and a paraphrase model trained on 545 million words of conversational data, for a sentence "*And I generally prefer*", the following paraphrase variants inside the lattice are among those generated: "*And I just like*", "*I mean I want* ", "*I guess I prefer* ", "*You know I need*", "*And I appreciate* ", "*I would have* ", "*'Cause I like* ", "*Well I need* " and "*So I like*".

3.3 Paraphrastic discounting: After paraphrase generation, the sufficient statistics, $C(h_i, w_i)$, used to estimate the paraphrastic LM probability $P_{\mathsf{PLM}}(w_i|h_i)$ predicting word w_i following history h_i, are now accumulated over each paraphrase word sequence in the paraphrase lattices, or equivalently n-best lists, and weighted by its posterior probability. This form of intuitive and interpretable counts smoothing automatically re-distributes statistics to alternative expressions of the same observed word sequence. It was previously exploited to improve the context coverage and generalization for several forms of LMs that do not explicitly capture the expressive richness related variability in natural languages, including back-off n-gram LMs [14], and feedforward NNLMs [15].

4. PARAPHRASTIC RNN LANGUAGE MODELS

As discussed in section 1, RNNLMs acquire their strong generalization by internally clustering variable length full history contexts using the similarity measure between their vector space representations. In this process, linguistic factors, such as paraphrase alternatives, are implicitly learned but not explicitly used in training to drive the underlying smoothing mechanism. An assumption is made that history contexts that differ significantly in their surface form or vector representations, despite being strongly related in meaning, are considered unlikely to share a similar RNNLM distribution. This can lead to poor context coverage and generalization. In order to address this issue, the general form of paraphrastic language modelling framework presented in section 3 can also be used to improve RNNLMs' performance.

4.1 Paraphrastic RNNLM training: The phrase level paraphrase model estimation and paraphrase lattice generation stages described in sections 3.1 to 3.2 are performed first during paraphrastic RNNLM training. As discussed in section 2, RNNLMs model untruncated, full history contexts. Hence, explicit N-best representation of multiple paraphrase variants is required in paraphrastic RNNLM training. When the number of alternative expressions increases, directly training RNNLMs on N-best paraphrase sentences becomes highly expensive. To handle this problem, the GPU based bunch mode training algorithm (bunch size 128) proposed in [4] was used to efficiently train paraphrastic RNNLMs.

The paraphrastic LM counts smoothing presented in section 3.3 uses the posterior probability of each paraphrase alternative in the paraphrase N-best list to deweight the contribution from unlikely paraphrase variants. The storage and use of these posterior probabilities complicates the bunch mode training algorithm [4] and also introduces additional computational overhead. To address this issue, a simplified approach is used in this paper. The generated N-best paraphrase data are arranged into a total of N blocks. Each block provides a paraphrase re-expression of the entire training corpus so that the i^{th} block contains the $(N - i + 1)^{th}$ paraphrases for each of the training sentences occurring in the same order. Such ordering of the blocks retains the ranking ordering of the N-best paraphrase alternatives and implicitly assigns a higher weighting to the more likely paraphrases with higher posterior probabilities during training. The overall model training process is summarized below.

```
phrase level paraphrase model estimation on LM data using the
  $n$-gram paraphrase induction algorithm described in [14];
**for** every sentence in training data **do**
    generate a paraphrase lattice using WFSTs as in section 3;
    extract N-best paraphrase alternatives from the lattices;
**end for**
arrange N-best paraphrase data into N blocks such that the $i^{th}$
  block contains the $(N - i + 1)^{th}$ paraphrase variants for each
  of the training sentence ocurring in the same order.
back propagation through time training in spliced sentence
  bunch mode until convergence.
```

In common with paraphrastic back-off or feedforward NNLMs, the resulting paraphrastic RNNLM $P_{\mathsf{PRNN}}(\cdot|h_i)$ is interpolated the conventional RNNLM $P_{\mathsf{RNN}}(\cdot|h_i)$ trained on the standard surface text data only. Let $\bar{P}_{\mathsf{RNN}}(\tilde{w}_i|h_i)$ denote the interpolated RNNLM probability for an in-vocabulary word $\tilde{w}_i$ following some history h_i,

$$\bar{P}_{\mathsf{RNN}}(\tilde{w}_i|h_i) \;=\; \lambda P_{\mathsf{RNN}}(\tilde{w}_i|h_i) + (1-\lambda) P_{\mathsf{PRNN}}(\tilde{w}_i|h_i) \qquad (4)$$

where λ is the weight assigned to the baseline RNNLM distribution $P_{\mathsf{RNN}}(\cdot)$ to be optimized on the perplexity of some held-out data.

4.2 Phrase and multi-level paraphrastic RNNLMs: In order to increase the context span, phrase level paraphrastic RNNLMs can also be trained by optimizing a simplified form of the criterion in equation (2), by dropping the word to phrase segmentation model $P(\boldsymbol{\psi}'|\mathcal{W}')$. Phrase level segmented paraphrase lattices or N-best alternatives can then be generated and used to train phrase level paraphrastic RNNLMs. In order to incorporate richer linguistic constraints, RNNLMs that model different units, for example, words and phrases, can be used. In common with previous research on paraphrastic feedforward NNLMs [17], these RNNLMs are first equal weight interpolated with the respective word, or phrase level, back-off n-gram LMs, before finally log-linearly combined. The resulting multi-level paraphrastic RNNLMs can be used to further improve discrimination [12, 13]. When applying this model, word level lattices need to be first converted to phrase level lattices before the log-linear combination is performed. The log-linear interpolation weights were empirically set as 2:1 for word and phrase level RNNLMs, and kept fixed for all experiments of this paper.

5. EXPERIMENTS AND RESULTS

In this section performance of paraphrastic RNNLMs are evaluated on the CU-HTK LVCSR system for conversational telephone speech (CTS) used in the 2004 DARPA EARS evaluation. The acoustic models were trained on approximately 2000 hours of Fisher conversational speech released by the LDC. A 59k recognition word list was used in decoding. The system uses a multi-pass recognition framework. A detailed description of the baseline system can be found in [7]. The 3 hour **dev04** data, which includes 72 Fisher conversations was used. For all results presented in this paper, matched pairs sentence-segment word error (MAPSSWE) based statistical significance test was performed at a significance level $\alpha = 0.05$.

The baseline 4-gram back-off LM was trained on a total of 545 million words from 2 text sources: the LDC Fisher acoustic transcriptions, **Fisher**, of 20 million words (weight 0.75), and the University Washington conversational web data [3], **UWWeb**, of 525 million words (weight 0.25). The baseline RNNLM was trained on the **Fisher** data. The modified architecture described in section 2 with 512 hidden layer nodes, a 38k word input layer vocabulary and 20k word output layer shortlist was used. A total of 3.0M phrase pairs of one word to four words maximum were automatically extracted from the **Fisher** and **UWWeb** data. More detailed description of the paraphrase extraction is in [17]. These were then used to generate N-best paraphrase sentences for the **Fisher** data to train various paraphrastic RNNLMs. These RNNLMs are then used in lattice rescoring using a 6-gram truncated history based approximation [16] for word error rate (WER) performance evaluation.

5.1. Perplexity of Paraphrastic RNNLMs

	RNNLM		
LM	**Paraph.**	**N-best**	**dev04**
w4g	-	-	51.80
rnn	×	-	51.02
	√	1	48.92
		2	47.89
		3	47.14
		4	46.99
		5	46.89
		10	**45.97**
		25	46.17
w4g+rnn	×	-	45.64
	√	1	45.16
		2	44.82
		3	44.47
		4	44.45
		5	44.45
		10	**43.97**
		25	44.10

Table 1. Perplexity of paraphrastic RNNLMs on **dev04**. "w4g" denotes a 4-gram back-off LM. "rnn" is an RNNLM trained on **Fisher** data and optionally its paraphrases as in Eq. (4). "w4g+rnn" is an interpolated LM combining "w4g" with "rnn" with equal weights.

The perplexity performance of various word level RNNLMs are shown in table 1. Consistent perplexity reductions were obtained by increasing the number of paraphrase alternatives used in paraphrastic RNNLM training from 1 to 10. The 10-best paraphrase variants based paraphrastic RNNLM (8th line in table 1), outperformed the non-paraphrastic baseline RNNLM (2nd line in table 1) by 5.1 points in perplexity (10% relative). However, no further perplexity improvements were found when further increasing the number of paraphrase alternatives used to 25. The same trend was also found after these paraphrastic RNNLMs were interpolated with the baseline 4-gram back-off LM (1st line in table 1). These are shown the bottom section of table 1. Again the 10-best paraphrase variants based paraphrastic RNNLM gave the lowest perplexity.

5.2. WER Performance of Paraphrastic RNNLMs

	RNNLM		
LM	**Paraph.**	**N-best**	**dev04**
w4g	-	-	16.67
w4g+rnn	×	-	15.30
	√	1	15.09
		2	15.00
		3	**14.90**
		4	15.00
		5	15.11
		10	14.97
		25	14.97
(w4g+rnn) $\circ$ (p4g+rnn$_{\rm p}$)	√	3	**14.73**

Table 2. Word error rate performance of paraphrastic RNNLMs on **dev04**. "(w4g+rnn) $\circ$ (p4g+rnn$_{\rm p}$)" denotes a multi-level RNNLM log-linearly combining word and phrase sequence level RNNLMs. Other naming conventions same as table 1.

The WER performance of these RNNLMs are shown in table 2 for **dev04**. The word level 4-gram baseline LM "w4g" gave a WER of 16.7%. After linearly interpolated with the baseline RNNLM trained on the original **Fisher** data, the WER was reduced to 15.30%, as shown in the 2nd line in table 2. In common with the trend found on perplexity in table 1, paraphrastic RNNLMs trained with various number of paraphrase variants consistently outperformed the baseline RNNLM by 0.19%-0.40% absolute in WER. Using 3 paraphrase alternatives in paraphrastic RNNLM training gave the lowest WER of 14.90%. Again, no further improvements were obtained using a larger number of paraphrase variants. This may be due to the deterioration of the paraphrase quality further down the N-best lists. It was also found that when restricting the paraphrase generation of section 3.2 to only allow word to word paraphrasing, the paraphrastic RNNLMs trained on the resulting paraphrase variants also outperformed the baseline RNNLM, though the improvements were smaller than using a full phrase level paraphrase model.

The incorporation of phrase level constraints in the form of a multi-level paraphrastic RNNLM that log-linearly combines the word level paraphrastic RNNLM "w4g+rnn" using 3 paraphrase alternatives and a comparable phrase level paraphrastic RNNLM "p4g+rnn$_{\rm p}$" gave a further small reduction in WER, as is shown in the last line of table 2. Using this paraphrastic multi-level RNNLM, total WER reductions of 1.94% absolute and 0.57% absolute were obtained over the baseline 4-gram word level LM "w4g" and the non-paraphrastic RNNLM "w4g+rnn" respectively, both being statistically significant. The overall improvements over the baseline 4-gram LM was increased from 8.4% to 11.6% relative.

6. CONCLUSION AND RELATION TO PRIOR WORK

Paraphrastic RNNLMs were investigated in this paper. Experimental results suggest the proposed method is also effective in improving RNNLM performance, consistent with the improvements reported in the earlier research on back-off n-gram LMs [14] and feedforward NNLMs [15]. Significant error rate reductions of 1.94% absolute (12% relative) were obtained on a state-of-the-art large vocabulary speech recognition task. In contrast, previous research on RNNLMs used no explicit paraphrastic modelling [19]. Future research will focus on modeling richer linguistic variabilities for RNNLMs.

7. REFERENCES

[1] I. Androutsopoulos and P. Malakasiotis (2010). "A survey of paraphrasing and textual entailment methods", *Journal of Artificial Intelligence Research*, 38:135-187, 2010.

[2] Y. Bengio and R. Ducharme (2003), "A neural probabilistic language model," *Journal of Machine Learning Research*, vol. 3, pp. 1137–1155, 2003.

[3] I. Bulyko, M. Ostendorf, and A. Stolcke (2003), "Getting more mileage from web text sources for conversational speech language modeling using class-dependent mixtures," in *Proc. HLT*, Edmonton, Canada, 2003.

[4] X. Chen, Y. Wang, X. Liu, M. J. F. Gales and P. C. Woodland (2014). "Efficient GPU-based training of recurrent neural network language models using spliced sentence bunch", in *Proc. ISCA Interspeech2014*, Singapore, pp. 641–645.

[5] A. Deoras, T. Mikolov, S. Kombrink and K. Church (2013), "Approximate inference: A sampling based modeling technique to capture complex dependencies in a language model," *Speech Communication*, vol. 55, no. 1, pp. 162–177, January 2013.

[6] A. Emami and L. Mangu (2007), "Empirical study of neural network language models for Arabic speech recognition," in *Proc. ASRU*, Kyoto, Japan, 2007, pp. 147–152.

[7] G. Evermann, H. Y. Chan, M. J. F. Gales, B. Jia, D. Mrva, P. C. Woodland, and K. Yu (2005), "Training LVCSR systems on thousands of hours of data," in *Proc. ICASSP*, Philadelphia, PA, 2005, vol. 1, pp. 209–212.

[8] Z. Harris (1954). "Distributional structure", *Word*, 10(2):3 pp.146-162.

[9] T. Hori, Y. Kubo, and A. Nakamura (2014). "Real-time one-pass decoding with recurrent neural network language model for speech recognition", in *Proc. IEEE ICASSP2014*, Florence, Italy.

[10] Z. Huang, G. Zweig, and B. Dumoulin (2014). " Cache based recurrent neural network language model inference for first pass speech recognition", in *Proc. IEEE ICASSP2014*, Florence, Italy, pp. 6404–6408.

[11] H.-S. Le, I. Oparin, A. Allauzen, J. Gauvain, and F. Yvon (2013), "Structured output layer neural network language models for speech recognition," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 21, no. 1, pp. 197–206, 2013.

[12] X. Liu, M. J. F. Gales, and P. C. Woodland (2013), "Use of contexts in language model interpolation and adaptation," *Computer Speech & Language*, vol. 27, no. 1, pp. 301–321, January 2013.

[13] X. Liu, J. L. Hieronymus, M. J. F. Gales and P. C. Woodland (2013). "Syllable language models for Mandarin speech recognition: exploiting character sequence models", *Journal of the Acoustical Society of America*, Volume 133, Issue 1, pp. 519-528, January 2013.

[14] X. Liu, M. J. F. Gales and P. C. Woodland (2012). "Paraphrastic language models", in *Proc. ISCA Interspeech2012*, Portland, Oregon.

[15] X. Liu, M. J. F. Gales and P. C. Woodland (2014). "Paraphrastic neural network language models", in *Proc. IEEE ICASSP2014*, Florence, Italy.

[16] X. Liu, Y. Wang, X. Chen, M. J. F. Gales and P. C. Woodland (2014). "Efficient lattice rescoring using recurrent neural network language models", in *Proc. IEEE ICASSP2014*, Florence, Italy.

[17] X. Liu, M. J. F. Gales, and P. C. Woodland (2014). "Paraphrastic language models", *Computer Speech and Language*, Volume 28, Issue 6, pp. 1298-1316., November 2014.

[18] N. Madnani and B. Dorr (2010). "Generating phrasal and sentential paraphrases: a survey of data-driven methods", *Computational Linguistics*, Vol. 36, No. 3, 2010.

[19] T. Mikolov, M. Karafiat, L. Burget, J. Cernocky, and S. Khudanpur (2010), "Recurrent neural network based language model," in *Proc. ISCA Interspeech*, Makuhari, Japan, 2010, pp. 1045–1048.

[20] T. Mikolov, S. Kombrink, L. Burget, J. H. Cernocky, and S. Khudanpur (2011), "Extensions of recurrent neural network language model," in *Proc. ICASSP*, Prague, Czech Republic, 2011, pp. 5528–5531.

[21] T. Mikolov, S. Kombrink, L. Burget, J. H. Cernocky and S. Khudanpur (2011), "RNNLM - Recurrent neural network language modeling toolkit", in demo session of *IEEE ASRU2011*, Hawaii.

[22] T. Mikolov, W. Yih, and G. Zweig (2013). "Linguistic regularities in continuous space word representations", in *Proc. NAACL-HLT-2013*, Association for Computational Linguistics, 27 May, 2013.

[23] F. Morin and Y. Bengio (2005), "Hierarchical probabilistic neural network language model," in *Proc. International workshop on artificial intelligence and statistics*, Barbados, 2005, pp. 246–252.

[24] M. Mohri (1997), "Finite-state transducers in language and speech processing," *Computational linguistics*, vol. 23, no. 2, pp. 269–311, 1997.

[25] J. Park, X. Liu, M. J. F. Gales, and P. C. Woodland (2010), "Improved neural network based language modelling and adaptation," in *Proc. ISCA Interspeech*, Makuhari, Japan, 2010, pp. 1041–1044.

[26] D. E. Rumelhart, G. E. Hintont, and R. J. Williams (1986), "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[27] H. Schwenk (2007) , "Continuous space language models," *Computer Speech & Language*, vol. 21, no. 3, pp. 492–518, 2007.

[28] Y. Si, Q. Zhang, T. Li, J. Pan, and Y. Yan (2013), "Prefix tree based n-best list re-scoring for recurrent neural network language model used in speech recognition system," in *Proc. ISCA Interspeech*, Lyon, France, 2013, pp. 3419–3423.

[29] M. Sundermeyer, R. Schlüter, and H. Ney (2012), "LSTM neural networks for language modeling," in *Proc. ISCA Interspeech*, Portland, OR, 2012.

[30] M. Sundermeyer, I. Oparin, J. L. Gauvain, B. Freiberg, R. Schluter, and H. Ney (2013), "Comparison of feedforward and recurrent neural network language models," in *Proc. ICASSP*, Vancouver, Canada, 2013, pp. 8430–8434.