7-22-2013

RNN Language Model with Word Clustering and Class-based Output Layer

Yongzhe Shi
Tsinghua University

Wei-Qiang Zhang
Tsinghua University

Jia Liu
Tsinghua University

Michael T. Johnson
Marquette University, michael.johnson@marquette.edu

Published version. *EURASIP Journal on Audio, Speech, and Music Processing*, Vol. 2013, No. 22 (July 22, 2013). DOI.

EURASIP Journal on
Audio, Speech, and Music Processing
a SpringerOpen Journal

RESEARCH Open Access

RNN language model with word clustering and class-based output layer

Yongzhe Shi[1*], Wei-Qiang Zhang[1], Jia Liu[1] and Michael T Johnson[2]

Abstract

The recurrent neural network language model (RNNLM) has shown significant promise for statistical language modeling. In this work, a new class-based output layer method is introduced to further improve the RNNLM. In this method, word class information is incorporated into the output layer by utilizing the Brown clustering algorithm to estimate a class-based language model. Experimental results show that the new output layer with word clustering not only improves the convergence obviously but also reduces the perplexity and word error rate in large vocabulary continuous speech recognition.

Keywords: Brown word clustering; RNN language model; Speech recognition

1 Introduction

Statistical language models estimate the probability of a word occurring in a given context, which plays an important role in many natural language processing applications such as speech recognition, machine translation, and information retrieval. Standard *n*-gram back-off language models (LMs) are widely used for their simplicity and efficiency. However, in this approach, words are modeled as discrete symbols with richer linguistic information, such as syntax and semantic, ignored completely. Additionally, large numbers of parameters need to be estimated, and due to the sparsity characteristics of natural language, the probability of low- and zero-frequency events is estimated crudely and inaccurately using various smoothing algorithms.

The distributional hypothesis in linguistics states that words occurring in the same context tend to have similar meanings. It is a reasonable assumption that similar words occur in the same context with similar probability, for example, 'America,' 'China,' and 'Japan' which usually come after the same preposition or as the subject of a sentence. Based on this assumption, neural network language models (NNLMs) [1,2] project the discrete word indices into a continuous space where similar words occur close together. The predicted probability of the next word is returned by a smooth function of the context representation, which alleviates the sparsity issue to some extent and leads to better generalization for unseen *n*-grams. In 2010, Elman's recurrent neural network was first used for language modeling by Mikolov [3] and then an extension of this model was proposed in 2011 [4,5]. The recurrent neural network language model (RNNLM) has a longer memory and has recently performed better than other modeling methods [3,4,6]. Accordingly, we select the RNNLM as our baseline approach in this paper.

One key issue is the heavy computational cost for the RNNLM. As the output layer contains one unit for each word in the vocabulary, it is infeasible to train the model for large vocabulary with hundreds of thousands of words. Therefore, reducing the complexity of neural network language models has been an important topic. Perhaps one method is to estimate the several thousand most frequent words (the shortlist) via NNLMs, while other words are estimated via *n*-gram back-off models. Unfortunately, it has been shown that this technique causes severe degradation of performance for a small shortlist [1]. Other tree-structured output layer methods have also been proposed to speed up the NNLMs [7,8]. In these

*Correspondence: shiyz09@gmail.com
[1]Tsinghua National Laboratory for Information Science and Technology, Department of Electronic Engineering, Tsinghua University, 100084 Beijing, China
Full list of author information is available at the end of the article

methods, the tree structure of the output layer needs to be constructed carefully using linguistic knowledge such as WordNet [9] or word continuous representation. In general, speed and performance need to be balanced so that training and testing process is accelerated as much as possible, without deteriorating the performance of the model.

In this paper, we introduce a new method for constructing a class-based output layer using the Brown clustering algorithm. The closest previous work to this is a simple frequency-based word factorizing algorithm used to construct the output layer [4]. Words are roughly clustered according to their frequencies in this method, with training speed increasing but performance degraded. We extend this work to improve the performance of RNNLM and speed up the training. Words are clustered off-line and then the word classes are embedded into the output layer to estimate the class-based language model, where the RNN is used to estimate the conditional probability of classes and words.

This paper is organized as follows: In Section 2, we introduce our baseline RNNLM and the proposed Brown clustering method for constructing the output layer. Perplexity evaluation on a public corpus is performed in Section 3. Our proposed model is further evaluated on the Wall Street Journal (WSJ) and Switchboard speech-to-text tasks in Sections 4 and 5. Finally, Section 6 concludes this paper and gives the future work.

2 Model description

2.1 RNN language model

An Elman recurrent neural network (RNN) [10] is shown in Figure 1. The hidden state is a function of the entire input history. RNNs are well known for their long memory and are widely used for dynamic system modeling and sequence prediction. Let V denotes the vocabulary with '1-of-$|V|$' coding used in the input layer, so that the ith word of the vocabulary is encoded as a binary $|V|$-dim vector, where the ith element is set as 1 and all others are 0. Let $h_t = \text{sigmoid}(W_{ih}x_t + W_{hh}h_{t-1})$, where x_t and h_t denote the input and the hidden activation at the current time step, respectively. The hidden state h_t is activated by the current input x_t and the previous hidden activation h_{t-1}. Define $\text{P}(w_t|w_1^{t-1}) = o_t = \text{softmax}(W_{ho}h_t)$, where w_t and w_1^{t-1} denote the next word and the context. The output layer o_t corresponds to the predicted probability of all words in the vocabulary. To speed up the training of RNNLM, a frequency-based extension of RNNLM is introduced in [4], which is a class-based model as shown in Figure 2. Let $\text{P}(w_t|w_1^{t-1}) = \text{P}(\text{C}(w_t)|w_1^{t-1})\text{P}(w_t|\text{C}(w_t), w_1^{t-1})$, where $\text{C}(w_t)$ denotes the class of the word w_t. The predicted probabilities of classes and specific words are estimated to decrease the computational complexity.

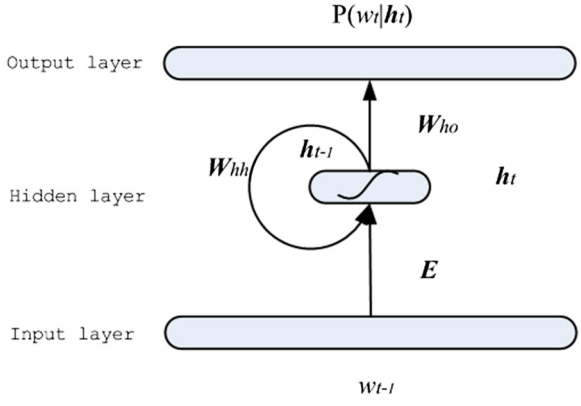

Figure 1 Recurrent neural network language model.

To speed this up, a simple frequency-based factorizing method is used to construct the equivalence class of words in the class-based model. Compared with RNNLM without a class-based layer, the perplexity (PPL) of the model shown in Table 1 is 10% higher for the Penn Treebank Corpus (one million words). Details can be found in Section 3.

2.2 Word clustering for output layer

2.2.1 Frequency-based clustering

Frequency-based word clustering is referred to as the frequency binning factorization method [4], where words are assigned to classes proportionally. Figure 3 gives the unigram cumulative probability distribution for the Penn Treebank Corpus, which describes the well-known phenomenon of Zipf's law in natural language. This method divides the cumulative probability into K partitions to form K frequency binnings which correspond to K clusters, a very rough word partition which only considers frequency. Zipf's law states that given some corpus of natural language utterances, the frequency of any word is inversely proportional to its rank in the frequency table. This means that the very few high-frequency words occupy most of the text corpus. It can be seen in Figure 3 that the most frequent 150 words and 2,800 words occupy more than 50% and 80% of the text corpus, respectively. In order to have 100 equal clusters, the top 150 words make up the first 50 clusters and the top 2,800 words make up the first 80 clusters, which means the last 7,200 words form 20 clusters. Therefore, clusters containing high-frequency words are very small, possibly even containing one word. In contrast, most words are in the remaining clusters, each containing hundreds or even thousands of words. Thus, the clustering results of this method depend severely on frequency distribution of the training corpus, leading to unsatisfactory clustering results.

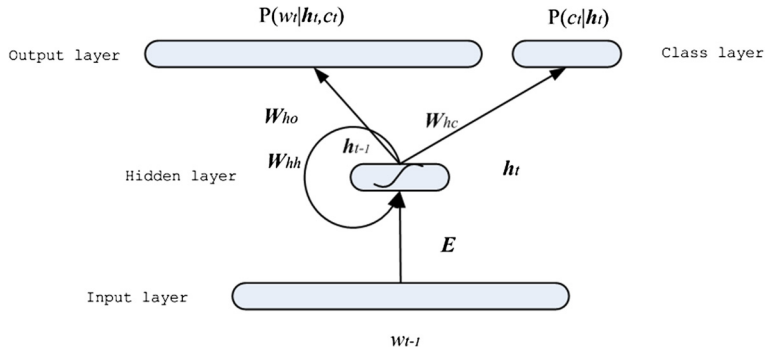

Figure 2 Class-based output layer for RNNLM.

2.2.2 Brown clustering

Brown clustering is a data-driven hierarchical word clustering algorithm [11,12], which is widely used in natural language processing. The input to this algorithm is text, which is a sequence of words $w_1, w_2, ..., w_n$, and the output of the clustering algorithm is a binary tree, the leaves of which are words. In this paper, we interpret all leaves with the same parent node as a cluster in the tree. The Brown clustering algorithm was first proposed to estimate a class-based language model [11]. Let V, C, and T denote the vocabulary, the word clusters and the text corpus, respectively. The optimization object is the cross entropy of the text corpus:

$$\begin{aligned}\text{loss}(C) &= -\frac{1}{|T|}\log \mathrm{P}(w_1 ... w_{|T|}) \\ &= -\frac{1}{|T|}\log \prod_{i=1}^{|T|} \mathrm{P}(\mathrm{C}(w_i)|\mathrm{C}(w_{i-1})\mathrm{P}(w_i|\mathrm{C}(w_i)))\end{aligned} \tag{1}$$

where $|T|$ denotes the length of text and $\mathrm{C}(\cdot)$ maps the words to the specific clusters. $\mathrm{P}(\mathrm{C}(w_i)|\mathrm{C}(w_{i-1}))$ and $\mathrm{P}(w_i|\mathrm{C}(w_i))$ can be estimated by frequency. Therefore, the object loss function can be rewritten as follows:

$$\begin{aligned}\text{loss}(C) &= -\frac{1}{|T|}\sum_{i=1}^{|T|} \log \frac{n_{c_i,c_{i-1}}}{n_{c_{i-1}}} * \frac{n_{w_i}}{n_{c_i}} \\ &= -\sum_{c,c' \in C} \mathrm{P}(c,c') \log \frac{\mathrm{P}(c,c')}{\mathrm{P}(c)\mathrm{P}(c')} \\ &\quad -\sum_{w \in V} \mathrm{P}(w)\log \mathrm{P}(w),\end{aligned} \tag{2}$$

Table 1 Perplexities on test set of Penn Treebank Corpus

Model	Perplexity
RNNLM (class 100)	135.49
RNNLM (no class layer)	123.00

where n_{w_i} denotes the occurrence count of pattern w_i that occurs in the corpus.

Initially, the algorithm starts with each word in its own cluster. As long as there are more than one cluster left, the algorithm merges the two clusters that minimizes the loss of the clustering result as shown in Equation 2. The naive algorithm has time complexity $O(|V|^3)$ [11] and is impractical for hundreds of thousands of words. Fortunately, a variant algorithm with time complexity $O(|V|K^2 + |T|)$ was proposed in [13], where K denotes the number of clusters. The algorithm is described as follows with details available in [13].

Input: text corpus **T**, the number of clusters K
Output: K word clusters

- Take the K most frequent words, put each into its own cluster $c_1, c_2, ..., c_K$
- For $i = K + 1 : |V|$
  1. Create a new cluster c_{K+1} for the ith most frequent word, giving $K + 1$ clusters.

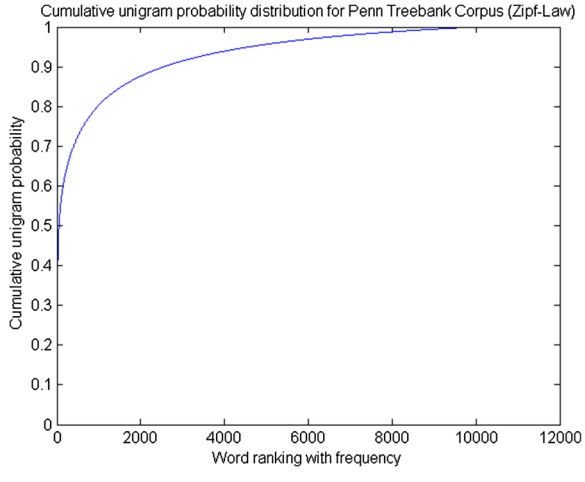

Figure 3 Cumulative unigram probability distribution for Penn Treebank Corpus with about one million words (Zipf's law).

2. Choose two clusters from $c_1, c_2, ..., c_{K+1}$ to merge, selecting these that minimize Equation 2.

- Implement $|V| - K$ merges to create a full hierarchical word clusters.

In this paper, we partition the words into clusters using this algorithm and demonstrate its effectiveness for RNN language modeling in terms of both perplexity and word error rate.

3 Penn Treebank Corpus evaluation

The proposed language model is evaluated on the Wall Street Journal portion of the Penn Treebank which is preprocessed by lowercasing words, removing punctuation and replacing numbers with the '*N*' symbol. Sections 00-20 (930K words) are used as training sets, sections 21-22 as validation sets (74K words), and sections 23-24 as test sets (82K words). The vocabulary size is 10K, including a special token for unknown words.

We compare the proposed model with the baseline RNNLM model [5,6]. We denote our proposed model as RNNLM-Brown and the baseline as RNNLM-Freq for convenience, where the 'Brown' and 'Freq' mean the Brown clustering and the frequency-based clustering, respectively. Both models have the same basic configuration (200 hidden units) for comparisons in the following experiments. The truncated backpropagation through time algorithm (BPTT) is used for training the RNNLMs using ten time steps. When the perplexity decreases very slowly or increases, the learning rate is halved. The basic 5-gram back-off language model (LM-KN5) is trained with the modified Kneser-Ney smoothing algorithm. Figure 4 shows the convergence process of the validation set's perplexity for RNNLM-Freq and RNNLM-Brown, where RNNLM-Brown with 13 epochs obtains the same perplexity as RNNLM-Freq with 24 epochs. We can see that the proposed RNNLM-Brown converges twice faster and obtains lower perplexity on the validation set. Accordingly, appropriate word clustering for the output layer can speed up the convergence, which is especially important for a large training corpus.

Table 2 Comparisons of perplexities on test set of Penn Treebank Corpus with different sizes of class layer

Class	RNNLM-Freq / +KN5 (words per second)	RNNLM-Brown / +KN5 (words per second)
30	135.57 / 113.13 (744)	131.46 / 110.83 (567)
50	136.39 / 113.53 (938)	129.79 / 109.96 (862)
100	135.49 / 113.07 (1,047)	128.36 / 109.33 (970)
200	136.03 / 112.89 (1,013)	128.52 / 109.13 (1,000)
400	135.75 / 113.04 (847)	128.03 / 109.09 (906)
800	134.98 / 112.51 (645)	128.09 / 109.23 (710)
1,600	133.44 / 111.93 (367)	128.67 / 109.47 (480)
10,000 (full)	123.00 / 106.00 (65)	123.00 / 106.00 (65)

The full model use the whole 10K vocabulary as the class layer, which is the same for both models. Perplexity of LM-KN5 on test set is 141.46.

In the following experiments, perplexity and training speed are evaluated with different sizes of class layers, as shown in Table 2. The first two columns refer to the baseline (RNNLM-Freq) and the interpolated model with LM-KN5 (RNNLM-Freq + KN5), respectively, which is consistent with the results reported in [4]. The last two columns correspond to the proposed language model (RNNLM-Brown) and the interpolated version with LM-KN5 (RNNLM-Brown + KN5). The full model uses the entire vocabulary for the class layer, in which each word

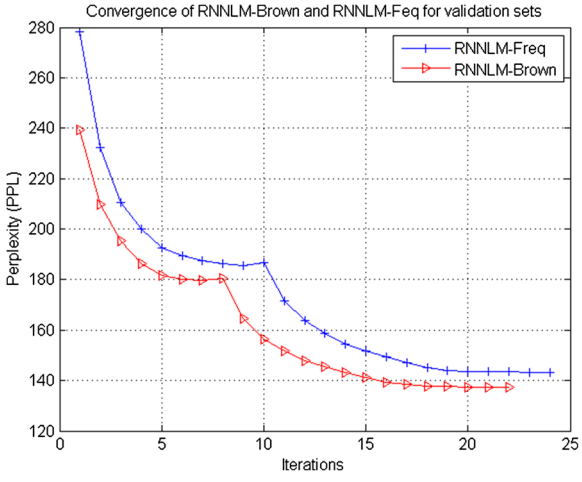

Figure 4 PPL convergence of RNNLM-Freq/Brown on validation sets.

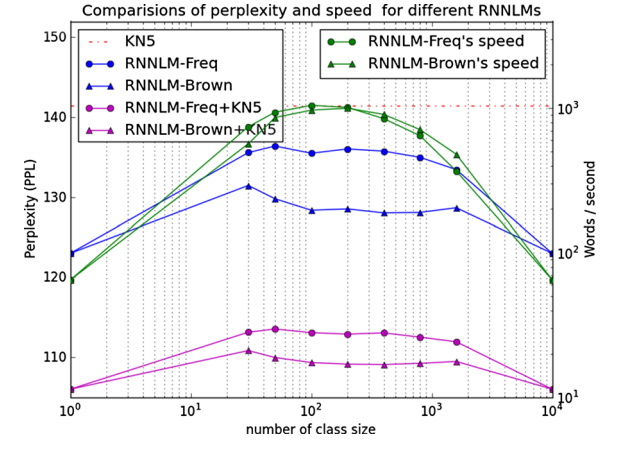

Figure 5 Detailed comparisons of perplexity and speed for different RNNLMs on Penn Treeback Corpus.

Table 3 Comparisons of perplexities on test set of five million training corpus with different sizes of class layer

Class	RNNLM-Freq / +KN5	RNNLM-Brown / +KN5
50	218.13 / 178.10	206.19 / 172.08
100	220.47 / 178.50	208.37 / 172.06
200	219.60 / 178.21	206.54 / 171.21
400	219.73 / 178.09	205.02 / 170.56

Perplexity of LM-KN5 with the same training text (five million words) on test set is 231.02.

corresponds to a separate cluster. The interpolated coefficients are determined according to the validation set. To make the observation easier, we plot the test perplexity and the speed of training in Figure 5, where the blue and red lines correspond to the left y-axis for perplexity and the green lines correspond to the right y-axis for the training speed. The training speed is evaluated by the number of words processed per second on a machine with an Intel® Core™2 Quad CPU Q9400 at 2.66 GHz, 8-GB RAM. In practice, the training speed first increases and then decreases with the increasing number of clusters. There is a trade-off between the perplexity and the training speed, especially for a larger corpus. From Figure 5, 100 or 200 clusters are the best choices balancing this, and the training speed is 15 times faster than that of the full model. Empirically, the best number of clusters is around $\sqrt{|V|}$, where $|V|$ denotes the size of vocabulary. In the experiments, the perplexity is reduced by about 5% for both single and interpolated models without reducing the speed of training, compared with the baseline. Moreover, the performance of the proposed RNNLM-Brown is much closer to that of the full model without the class layer.

4 WSJ speech recognition experiment

To evaluate the performance of the proposed language model for speech recognition, we use the WSJ task which is a standard task for language model evaluation. The acoustic model is a 6,000-tied-state continuous model with 32 Gaussians, trained on the WSJ1 Corpus; the details of which can be found in [14]. The LM training text contains 26 million words from the entire NYT-1994 section of the English Gigaword Corpus. The top 64,000 most frequent words are selected as the vocabulary

Table 4 WER for development set rescored with different RNNLMs and LM-KN5

	Class			
Model	50	100	200	400
LM-KN5 + RNNLM-Freq (%)	11.7	11.6	11.6	11.7
LM-KN5 + RNNLM-Brown (%)	11.5	11.5	11.5	11.4

The WER of 1-best hypothesis is 13.4% and the WER for LM-KN5 is 12.9%.

Table 5 WER for evaluation set rescored with different RNNLMs and LM-KN5

	Class			
Model	50	100	200	400
LM-KN5 + RNNLM-Freq (%)	13.0	13.1	13.0	13.1
LM-KN5 + RNNLM-Brown (%)	12.7	12.7	12.7	12.4

The WER of 1-best hypothesis is 14.1% and the WER for LM-KN5 is 13.8%.

and other words are mapped to a special token. Trigram (LM-KN3) and 5-gram (LM-KN5) models are trained on the text using the MITLM toolkit [15] with the modified Kneser-Ney smoothing algorithm for decoding and rescoring.

In the experiment, we use the NIST 1993 CSR Hub and Spoke Benchmark Test Corpora [16] as our test bed. We select the hub 1 and spoke 1, 2 and 4 sections for evaluations, which contain 1,251 utterances (about 25K words, 2.5 h of voice data) All the utterances are divided equally into two parts as the development set and the evaluation set.

Due to the complexity of training the neural network language model, this requires several days or an even longer time period to converge for several million training words. Thus, we randomly select about five million words from the NYT-1994 section of English Gigaword Corpus as the training data, 500K words as the validation data for early stopping and 500K words as the test data for perplexity evaluation. The top 30K frequent words are selected as the vocabulary. The truncated BPTT is used for training the different RNNLMs with ten time steps. Two hundred hidden units are used. The learning rate is initially set to 0.1 and halved when the perplexity of the validation data is increased. The detailed results are given

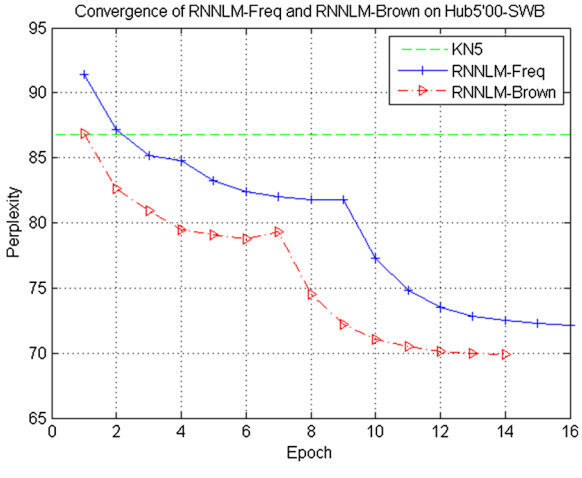

Figure 6 PPL convergence of RNNLM-Freq/Brown on Hub5'00-SWB set.

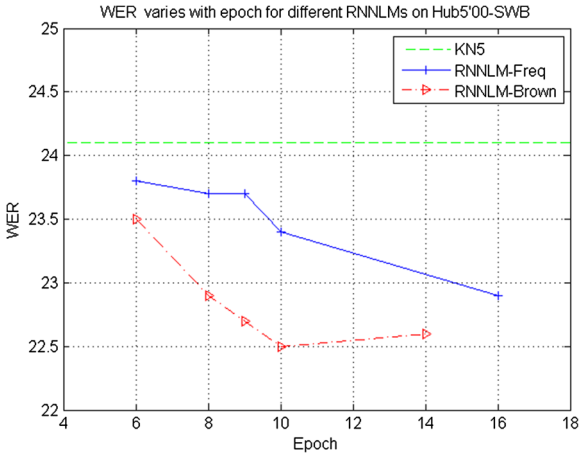

Figure 7 WER convergence of 100-best rescoring with RNNLM-Freq/Brown on Hub5'00-SWB set.

in Table 3, where consistent improvements are observed with different class sizes for the larger training corpus. The perplexity is reduced by approximately 5%, compared with the baseline.

In the following experiments, the 200-best hypotheses are generated using back-off trigram trained on the entire 26 million words and then rescored with different language models for comparisons. The interpolated coefficient of RNNLM and LM-KN5 is determined on the development set. Detailed results can be found in Tables 4 and 5. We can see that the word error rate (WER) for evaluation set is consistently reduced by 0.3% to 0.7% absolutely, compared with that of RNNLM-Freq, and 1.4% to 1.7% compared with the 1-best hypothesis (i.e. more than 10% relative reduction of WER).

5 Switchboard speech recognition experiment

In this section, the effectiveness of our proposed model on the task of speech-to-text transcription is evaluated on the 309h Switchboard-I training set [17], a larger corpus than WSJ Corpus. The system uses 13-dimensional PLP features with rolling-window mean-variance normalization and up to third-order derivatives, reduced to 39 dimensions by HLDA. The speaker-independent three-state cross-word triphones share 9,308 CART-tied states. The GMM-HMM baseline system has 40-Gaussian mixtures per state, trained with maximum likelihood and refined discriminatively with the boosted maximum-mutual-information criterion.

The data for system development is the 1831-segment SWB part of the NIST 2000 Hub5 evaluation set (Hub5'00-SWB). The FSH half of the 6.3 h Spring 2003 NIST Rich Transcription set (RT03S-FSH) acts as the evaluation set. Based on Kneser-Ney smoothing, a back-off trigram language model (LM-KN3) was trained on the 2000h Fisher transcripts containing 20 million tokens for decoding, where the vocabulary is limited to 53K words and unknown words are mapped into a special token <unk>. Note that no other unkown text is used to train LMs for interpolations so that the following experimental results are easily repeatable. The pronouncing dictionary comes from the CMU pronouncing dictionary [18].

Two models (RNNLM-Freq/Brown) with 300 hidden units are trained on the entire training text for comparisons. The perplexity convergence of RNNLM-Freq and RNNLM-Brown on Hub5'00-SWB set is shown in Figure 6. It can be seen that waiting long enough, the RNNLM with brown clustering converges better than that with frequency partitions. Moreover, the perplexity of RNNLM-Brown with 9 epochs competes with that of RNNLM-Freq with 16 epochs; it means that RNNLM-Brown converges twice faster than RNNLM-Freq.

Subsequently, these models are further compared to rescore N-best hypotheses. For convenience, 100-best hypotheses are generated from Hub5'00-SWB and RT03S-FSH and rescored by different LMs. The interpolation weights are tuned on Hub5'00-SWB, and the performances of these LMs are evaluated on RT03S-FSH. These intermediate models during the training are used for rescoring, and the performance of these models in word error rate is plotted in Figure 7 for comparisons, where the RNNLM-Brown converges much faster and better than

Table 6 100-Best rescoring with different LMs on Hub5'00-SWB and RT03S-FSH

Model	Perplexity		WER (%, absolute change)	
	Hub5'00-SWB	RT03S-FSH	Hub5'00-SWB	RT03S-FSH
LM-KN3	89.40	66.76	24.5	27.5
LM-KN5	86.78	63.80	24.1 (−0.4)	27.1 (−0.4)
RNNLM-Freq	72.47	55.76	22.9 (−1.6)	25.9 (−1.6)
RNNLM-Freq + LM-KN5	67.66	52.15	22.4 (−2.1)	25.5 (−2.0)
RNNLM-Brown	69.91	54.48	22.6 (−1.9)	25.7 (−1.8)
RNNLM-Brown + LM-KN5	*66.00*	*51.24*	*22.2 (−2.3)*	*25.3 (−2.2)*

Values in italics indicate the lowest perplexity and WER on Hub5'00-SWB and RT03S-FSH.

RNNLM-Freq. We can see that RNNLM-Brown with 9 epochs obtains the same WER as RNNLM-Freq with 16 epochs.

The performances of different LMs in perplexity and word error rate are shown in Table 6, where Hub5'00-SWB and RT03S-FSH are used for validation and evaluation, respectively. We can see that the WER of our proposed model RNNLM-Brown interpolated with LM-KN5 obtains the lowest perplexity 51.24 and word error rate 25.3% on the evaluation set. In a word, our proposed RNNLM-Brown converges faster and better in the experiment.

6 Conclusions

In this paper, the Brown word clustering algorithm is proposed to construct a class layer for the RNNLM. Experimental results show that our proposed RNNLM-Brown improves the perplexity and decreases the word error rate obviously. The performance of our proposed RNNLM-Brown is much closer to that of the full model without a class layer. Moreover, our proposed RNNLM-Brown converges twice faster than RNNLM-Freq, which is critical for a large-scale dataset. Additionally, we notice that the outputs of the brown clustering algorithm include a binary tree structure of clusters, which is not used in this paper. In future work, we will further investigate this tree-structured output layer according to the hierarchical word cluster results. Additionally, we will also investigate whether soft clustering of words [19] can be incorporated into the RNNLM to further improve its performance.

Abbreviations

ASR: Automatic speech recognition; BPTT: Backpropagation through time algorithm; LM: Standard back-off *n*-gram language model; NNLM: Neural network language model; RNNLM: Recurrent neural network language model; Freq: Freq-based word clustering; Brown: Brown word clustering; KN: Kneser-Ney smoothing algorithm; LM-KN5: 5-gram based on Kneser-Ney smoothing algorithm; PPL: Perplexity; WER: Word error rate; WSJ: Wall Street Journal.

Competing interests

The authors declare that they have no competing interests.

Acknowledgements

This work was supported by the National Natural Science Foundation of China under grant nos. 61273268, 61005019 and 90920302, and in part by Beijing Natural Science Foundation Program under grant no. KZ201110005005.

Author details

[1]Tsinghua National Laboratory for Information Science and Technology, Department of Electronic Engineering, Tsinghua University, 100084 Beijing, China. [2]Department of Electrical Engineering, Marquette University, WI 53201, Milwaukee, USA.

Received: 8 October 2012 Accepted: 2 July 2013
Published: 22 July 2013

References

1. F Balado, NJ Hurley, EP Mccarthy, GCM Silvestre, Continuous space language models. Comput. Speech. Lang. **21**(3), 492–518 (2007)
2. LH Son, R Allauzen, G Wisniewski, F Yvon, in *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Training continuous space language models: some practical issues. Massachusetts, 9–11 October 2010, pp. 778–788
3. M Tomas, K Martin, B Lukas, HG Jan, K Sanjeev, in *Proceedings of the Annual Conference of International Speech Communication Association (INTERSPEECH)*. Recurrent neural network based language model. Chiba, 26–30 September 2010, pp. 1045–1048
4. M Tomas, K Stefan, B Lukas, HG Jan, K Sanjeev, in *Proceedings of IEEE International Conference on Acoustic, Speech and Signal Processing (ICASSP)*. Extensions of recurrent neural network language model. Prague, 22–27 May 2011
5. M Tomas, D Anoop, K Stefan, B Lukas, HG Jan, in *Proceedings of Automatic Speech Recognition and Understanding Workshop (ASRU)*. RNNLM - recurrent neural network language modeling toolkit. Waikoloa, 11–15 December 2011
6. M Tomas, D Anoop, K Stefan, B Lukas, HG Jan, in *Proceedings of the Annual Conference of International Speech Communication Association (INTERSPEECH 2011)*. Empirical evaluation and combination of advanced language modeling techniques. Florence, 27–31 August 2011
7. F Morin, Y Bengio, in *Proceedings of AISTATS*. Hierarchical probabilistic neural network language model. Christchurch, 6–8 January 2005, pp. 246–252
8. HS Le, I Oparin, A Allauzen, JL Gauvain, F Yvon, in *Proceedings of IEEE International Conference on Acoustic, Speech and Signal Processing (ICASSP)*. Structured Output Layer neural network language model. Prague, 22–27 May 2011, pp. 5524–5527
9. C Fellbaum, *WordNet: An Electronic Lexical Database*. (MIT Press, Cambridge, 1998)
10. JL Elman, Finding structure in time. Cogn. Sci. **14**(2), 179–211 (1990)
11. PF Brown, PV deSouza, RL Mercer, VJD Pietra, JC Lai, Class-based n-gram models for natural language. Comput. Linguist. **18**(4), 467–479 (1992)
12. S Martin, J Liermann, H Ney, Algorithms for bigram and trigram word clustering. Speech Commun. **24**, 1253–1256 (1998)
13. P Liang, Semi-supervised learning for natural language processing. Master's thesis, MIT, 2005. http://dspace.mit.edu/handle/1721.1/33296
14. K Vertanen, *Baseline WSJ acoustic models for HTK and Sphinx: training recipes and recognition experiments*, (2006). http://keithv.com/software/
15. BJP Hsu, MIT language modeling toolkit (2009). http://code.google.com/p/mitlm/
16. 1993 ARPA CSR Hub and Spoke Benchmark Tests Corpora (1993). http://www.ldc.upenn.edu/Catalog/readme_files/csr2.readme.html
17. J Godfrey, E Holliman, *Switchboard-1 Release*, vol. 2. (Linguistic Data Consortium, Philadelphia, 1997)
18. The CMU Pronouncing Dictionary Release 0.7a (2007). http://www.speech.cs.cmu.edu/cgi-bin/cmudict
19. Y Su, in *Proceedings of IEEE International Conference on Acoustic, Speech and Signal Processing (ICASSP)*. Bayesian class-based language models. Prague, 22–27 May 2011

doi:10.1186/1687-4722-2013-22
Cite this article as: Shi *et al.*: **RNN language model with word clustering and class-based output layer.** *EURASIP Journal on Audio, Speech, and Music Processing* 2013 **2013**:22.