Converting Continuous-Space Language Models into N-gram Language Models for Statistical Machine Translation

Rui Wang^{1,2,3}, Masao Utiyama^{2}, Isao Goto^{2}, Eiichro Sumita^{2}, Hai Zhao^{1,3} and Bao-Liang Lu^{1,3}
1 Center for Brain-Like Computing and Machine Intelligence,
Department of Computer Science and Engineering,
Shanghai Jiao Tong Unviersity, Shanghai, 200240, China
2 Multilingual Translation Laboratory, MASTAR Project,
National Institute of Information and Communications Technology
3-5 Hikaridai, Keihanna Science City, Kyoto, 619-0289, Japan
3 MOE-Microsoft Key Lab. for Intelligent Computing and Intelligent Systems
Shanghai Jiao Tong Unviersity, Shanghai 200240 China
wangrui.nlp@gmail.com, mutiyama/igoto/eiichiro.sumita@nict.go.jp, zhaohai@cs.sjtu.edu.cn, bllu@sjtu.edu.cn

Abstract

Neural network language models, or continuous-space language models (CSLMs), have been shown to improve the performance of statistical machine translation (SMT) when they are used for reranking n-best translations. However, CSLMs have not been used in the first pass decoding of SMT, because using CSLMs in decoding takes a lot of time. In contrast, we propose a method for converting CSLMs into back-off n-gram language models (BNLMs) so that we can use converted CSLMs in decoding. We show that they outperform the original BNLMs and are comparable with the traditional use of CSLMs in reranking.

1 Introduction

Language models are important in natural language processing tasks such as speech recognition and statistical machine translation. Traditionally, back-off n-gram language models (BNLMs) (Chen and Goodman, 1996; Chen and Goodman, 1998; Stolcke, 2002) are being widely used for these tasks.

Recently, neural network language models, or continuous-space language models (CSLMs) (Bengio et al., 2003; Schwenk, 2007; Le et al., 2011) are being used in statistical machine translation (SMT) (Schwenk et al., 2006; Son et al., 2010; Schwenk et al., 2012; Son et al., 2012; Niehues and Waibel, 2012). These works have shown that CSLMs can improve the BLEU (Papineni et al., 2002) scores of SMT when compared with BNLMs, on the condition that the training data for language modeling are the same size. However, in practice, CSLMs have not been widely used in SMT.

One reason is that the computational costs of training and using CSLMs are very high. Various methods have been proposed to tackle the training cost issues (Son et al., 2010; Schwenk et al., 2012; Mikolov et al., 2011). However, there has been little work on reducing using costs. Since the using costs of CSLMs are very high, it is difficult to use CSLMs in decoding directly.

A common approach in SMT using CSLMs is the two pass approach, or n-best reranking. In this approach, the first pass uses a BNLM in decoding to produce an n-best list. Then, a CSLM is used to rerank those n-best translations in the second pass. (Schwenk et al., 2006; Son et al., 2010; Schwenk et al., 2012; Son et al., 2012)

Another approach is using restricted Boltzmann machines (RBMs) (Niehues and Waibel, 2012) instead of using multi-layer neural networks (Bengio et al., 2003; Schwenk, 2007; Le et al., 2011). Since probability in a RBM can be calculated very efficiently (Niehues and Waibel, 2012), they can use the RBM language model in SMT decoding. However, the RBM was just used in an adaptation of SMT, not in a large SMT task, because the training costs of RBMs are very high.

The last approach is using a BNLM to simulate a CSLM (Deoras et al., 2011; Arsoy et al., 2013). (Deoras et al., 2011) used a recurrent neural network language model (RNNLM) to generate a large amount of text, which was generated by sampling words from the probability distributions calculated by the RNNLM. Then, they trained the BNLM from the text using the interpolated Kneser-Ney smoothing method. (Arsoy et al., 2013) converted neural network language models of increasing order to pruned back-off language models, using lower-order models to constrain the n-grams allowed in higher-order models.

Both of these methods were used in decoding for speech recognition. These methods were applied to not-so-large scale experiments (55 million (M) words for training their BNLMs) (Arsoy et al., 2013). In contrast, our method is applied to SMT and can be used to improve a BNLM created from 746 M words by using a CSLM trained from 42 M words.

Because BNLMs can be trained from much larger corpora than those that can be used for training CSLMs, improving a BNLM by using a CSLM trained from a smaller corpus is very important. Actually, a CSLM trained from a smaller corpus can improve the BLEU scores of SMT if it is used in the n-best reranking (Schwenk, 2010; Huang et al., 2013). In contrast, we will demonstrate that a BNLM simulating a CSLM can improve the BLEU scores of SMT in the first pass decoding.

Our approach is as follows: (1) First, we train a CSLM (Schwenk, 2007) from a corpus. (2) Second, we also train a BNLM from the same corpus or larger corpus. (3) Finally, we rewrite the probability of each n-gram of the BNLM with that probability calculated from the CSLM. We also re-normalize the probabilities of the BNLM, then use the re-written BNLM in SMT decoding.

In Section 2, we describe the BNLM and CSLM (Schwenk, 2010) used for re-writing BNLMs. In Section 3, we describe the method of converting a CSLM into a BNLM. In Sections 4 and 5, we evaluate our method and conclude.

2 Language Models

In this section, we will introduce the standard BNLM and CSLM structure and probability calculation.

2.1 Standard back-off ngram language model

A BNLM predicts the probability of a word w_i given its preceding $n-1$ words $h_i = w_{i-n+1}^{i-1}$. But it will suffer from data sparseness if the context, h_i, does not appear in the training data. So an estimation by "backing-off" to models with smaller histories is necessary. In the case of the modified Kneser-Ney smoothing (Chen and Goodman, 1998), the probability of w_i given h_i under a BNLM, $P_b(w_i|h_i)$, is:

$$P_b(w_i|h_i) = \hat{P}_b(w_i|h_i) + \gamma(h_i)P_b(w_i|w_{i-n+2}^{i-1}) \tag{1}$$

where $\hat{P}_b(w_i|h_i)$ is a discounted probability and $\gamma(h_i)$ is the back-off weight. A BNLM is used with a CSLM as shown below.

2.2 CSLM structure and probability calculation

The main structure of a CSLM using a multi-layer neural network contains four layers: the input layer projects all words in the context h_i onto the projection layer (the first hidden layer); the second hidden layer and the output layer achieve the non-liner probability estimation and calculate the language model probability $P(w_i|h_i)$ for the given context. (Schwenk, 2007).

The CSLM calculates the probabilities of all words in the vocabulary of the corpus given the context at once. However, because the computational complexity of calculating the probabilities of all words is quite high, the CSLM is only used to calculate the probabilities of a subset of the whole vocabulary. This subset is called a *short-list*, which consists of the most frequent words in the vocabulary. The CSLM also calculates the sum of the probabilities of all words not in the short-list by assigning a neuron for that purpose. The probabilities of other words not in the short-list are obtained from a BNLM (Schwenk, 2007; Schwenk, 2010).

Let w_i, h_i be the current word and history. The CSLM with a BNLM calculates the probability of w_i given h_i, $P(w_i|h_i)$, as follows:

$$P(w_i|h_i) = \begin{cases} \frac{P_c(w_i|h_i)}{1-P_c(o|h_i)}P_s(h_i) & \text{if } w_i \in \text{short-list} \\ P_b(w_i|h_i) & \text{otherwise} \end{cases} \tag{2}$$

where $P_c(\cdot)$ is the probability calculated by the CSLM, $P_c(o|h_i)$ is the probability of the neuron for the words not in the short-list, $P_b(\cdot)$ is the probability calculated by the BNLM as in Eq. 1, and

$$P_s(h_i) = \sum_{v \in \text{short-list}} P_b(v|h_i). \tag{3}$$

It can be considered that the CSLM redistributes the probability mass of all words in the short-list. This probability mass is calculated by using the BNLM.

3 Conversion of CSLM into BNLM

As described in the introduction, we first train a CSLM from a corpus. We also train a BNLM from the same corpus or a larger corpus. Then, we rewrite the probability of each ngram in the BNLM with the probability calculated from the CSLM.

First, we use the probabilities of 1-grams in the BNLM as they are. Next, we rewrite the probabilities of n-grams (n=2,3,4,5) in the BNLM with the probabilities calculated by using the n-gram CSLM, respectively. Note that the n-gram CSLM means that the length of its history is $n-1$. Note also that we only need to rewrite the probabilities of n-grams ending with a word in the short-list. Finally, we re-normalize the probabilities of the BNLM using the SRILM's '-renorm' option.

When we rewrite a BNLM trained from a larger corpus, the ngrams in the BNLM often contain unknown words for the CSLM. In that case, we use the probabilities in the BNLM as they are.

4 Experiments

4.1 Common settings

We used the patent data for the Chinese to English patent translation subtask from the NTCIR-9 patent translation task (Goto et al., 2011). The parallel training, development, and test data consisted of 1 M, 2,000, and 2,000 sentences, respectively.

We followed the settings of the NTCIR-9 Chinese to English translation baseline system (Goto et al., 2011) except that we used various language models to compare them. We used the MOSES phrase-based SMT system (Koehn et al., 2003), together with Giza++ (Och and Ney, 2003) for alignment and MERT (Och, 2003) for tuning on the development data. The translation performance was measured by the case-insensitive BLEU scores on the tokenized test data. We used `mteval-v13a.pl` for calculating BLEU scores.[1]

[1]It is available at `http://www.itl.nist.gov/iad/mig/tests/mt/2009/`

We used the 14 standard SMT features: five translation model scores, one word penalty score, seven distortion scores and one language model score. Each of the different language models was used to calculate the language model score.

As the baseline BNLM, we trained a 5-gram BNLM with modified Kneser-Ney smoothing using the English side of the 1 M sentences training data, which consisted of 42 M words. We did not discard any n-grams in training this model. That is, we did not use count cutoffs. We call this BNLM as *BNLM42*.

A 5-gram CSLM was trained on the same 1 M training sentences using the CSLM toolkit (Schwenk, 2010). The settings for the CSLM were: projection layer of dimension 256 for each word, hidden layer of dimension 384 and output layer (short-list) of dimension 8192, which were recommended in the CSLM toolkit. We call this CSLM *CSLM42*. CSLM42 used BNLM42 as the background BNLM.

We also trained a larger 5-gram BNLM with modified Kneser-Ney smoothing by adding sentences from the 2005 US patent data distributed in the NTCIR-8 patent translation task (Fujii et al., 2010) to the 42 M words. The data consisted of 746 M words. We call this BNLM *BNLM746*. We discarded 3,4,5-grams that occurred only once when we created BNLM746.

Next, we re-wrote BNLM42 with CSLM42 by using the method described in Section 3. This re-written BNLM was interpolated with BNLM42. The interpolation weight was determined by the grid search. That is, we changed the interpolation weight to 0.1, 0.3, 0.5, 0.7, 0.9 to create an interpolated BNLM. Then we used that BNLM in the SMT system to tune the weight parameters on the first half of the development data. Next, we selected the interpolation weight that obtained the highest BLEU score on the second half of the development data. After we selected the interpolation weight, we applied MERT again to the 2,000 sentence development data to tune the weight parameters.[2] We call this BNLM *CONV42*. We also obtained *CONV746* by re-writing BNLM746 with CSLM42

[2]We aware that the interpolation weight might be determined by minimizing the perplexity on the development data. However, we opted to directly maximize the BLEU score.

in the same way.

The vocabulary of these language models was the same, which was extracted from the 1 M training sentences.

4.2 Experimental results

Table 1 shows the percent BLEU scores on the test data. The figures in the "1st pass" column show the BLEU scores in the first pass decoding when we changed the language model. The figures in the "reranking" column show the BLEU scores when we applied CSLM42 to rerank the 100-best lists for the different language models. When we applied CSLM42 for reranking, we added the CSLM42 score as the additional 15th feature. The weight parameters were tuned by using Z-MERT (Zaidan, 2009).

LMs	1st pass	rerank
BNLM42	31.60	32.44
CONV42	32.58	32.98
BNLM746	32.83	33.36
CONV746	33.22	33.54

Table 1: Comparison of BLEU scores

We also performed the paired bootstrap resampling test (Koehn, 2004).[3] We sampled 2000 samples for each significance test.

Table 2 shows the results of a statistical significance test, in which the "1st" is short for the "1st pass". The marks indicate whether the LM to the left of a mark is significantly better than that above the mark at a certain level. ("$\gg$": significantly better at $\alpha = 0.01$, "$>$": $\alpha = 0.05$, "$-$": not significantly better at $\alpha = 0.05$)

First, as shown in the tables, the reranking by applying CSLM42 increased the BLEU scores for all language models. This observation is in accordance with those of previous work (Schwenk, 2010; Huang et al., 2013).

Second, the reranking results of BNLM42 (32.44) were not better than those of the first pass of BNLM746 (32.83). This indicates that if the underlying BNLM is made from a small corpus, the reranking using CSLM can not compensate for it.

[3]We used the code available at `http://www.ark.cs.cmu.edu/MT/`.

	BNLM746 (rerank)	CONV746 (1st)	CONV42 (rerank)	BNLM746 (1st)	CONV42 (1st)	BNLM42 (rerank)	BNLM42 (1st)
CONV746 (rerank)	$-$	$\gg$	$\gg$	$\gg$	$\gg$	$\gg$	$\gg$
BNLM746 (rerank)		$-$	$\gg$	$>$	$\gg$	$\gg$	$\gg$
CONV746 (1st)			$\gg$	$-$	$\gg$	$\gg$	$\gg$
CONV42 (rerank)				$-$	$\gg$	$\gg$	$\gg$
BNLM746 (1st)					$-$	$\gg$	$\gg$
CONV42 (1st)						$-$	$\gg$
BNLM42 (rerank)							$\gg$

Table 2: Significance tests for systems with different LMs

Third, CONV42 was better than BNLM42 for both first-pass and reranking. This also holds in the case of CONV746 and BNLM746. This indicated that our conversion method improved the BNLMs, even if the underlying BNLM was trained on a larger corpus than that used for training the CSLM. As described in the introduction, this is very important because BNLMs can be trained from much larger corpora than those that can be used for training CSLMs. This observation has not been found in the previous work.

In addition, the first-pass of CONV42 and CONV746 (32.58 and 33.22) were comparable with those of the reranking results of BNLM42 and BNLM746 (32.44 and 33.36), respectively. That is, there were no significant differences between these results. This indicates that our conversion method preserves the performance of the reranking using CSLM.

5 Conclusion

We have proposed a method for converting CSLMs into BNLMs. The method can be used to improve a BNLM by using a CSLM trained from a smaller corpus than that used for training the BNLM. We have also shown that BNLMs created by our method performs as good as the reranking using CSLMs.

Our future work is to compare our conversion method with that of (Arsoy et al., 2013).[4]

[4]We aware that (Arsoy et al., 2013) compared their method with the one that is identical with our method. However, the experiments were conducted on a speech recognition task and the scale of the experiment was not so large. Since we noticed their work just before the submission of our paper, we did not have time to compare their method with our method in SMT.

Acknowledgments

We appreciate the helpful discussion with Andrew Finch and Paul Dixon, and three anonymous reviewers for many invaluable comments and suggestions to improve our paper. This work is supported by the National Natural Science Foundation of China (Grant No. 60903119, No. 61170114 and No. 61272248), the National Basic Research Program of China (Grant No. 2013CB329401) and the Science and Technology Commission of Shanghai Municipality (Grant No. 13511500200).

References

Ebru Arsoy, Stanley F. Chen, Bhuvana Ramabhadran, and Abhinav Sethy. 2013. Converting neural network language models into back-off language models for efficient decoding in automatic speech recognition. In *Proc. of IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP 2013)*, Vancouver, Canada, May. IEEE.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. 2003. A neural probabilistic language model. *Journal of Machine Learning Research (JMLR)*, 3:1137–1155, March.

Stanley F. Chen and Joshua Goodman. 1996. An empirical study of smoothing techniques for language modeling. In *Proceedings of the 34th annual meeting on Association for Computational Linguistics*, ACL '96, pages 310–318, Santa Cruz, California, June. Association for Computational Linguistics.

Stanley F. Chen and Joshua Goodman. 1998. An empirical study of smoothing techniques for language modeling. Technical report, Computer Science Group, Harvard Univ.

A. Deoras, T. Mikolov, S. Kombrink, M. Karafiat, and Sanjeev Khudanpur. 2011. Variational approximation of long-span language models for lvcsr. In *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on*, pages 5532–5535, Prague, Czech Republic, May. IEEE.

Atsushi Fujii, Masao Utiyama, Mikio Yamamoto, and Takehito Utsuro. 2010. Overview of the patent translation task at the ntcir-8 workshop. In *In Proceedings of the 8th NTCIR Workshop Meeting on Evaluation of Information Access Technologies: Information Retrieval, Question Answering and Cross-lingual Information Access*, pages 293–302, Tokyo, Japan, June.

Isao Goto, Bin Lu, Ka Po Chow, Eiichiro Sumita, and Benjamin K. Tsou. 2011. Overview of the patent machine translation task at the NTCIR-9 workshop. In *Proceedings of NTCIR-9 Workshop Meeting*, pages 559–578, Tokyo, Japan, December.

Zhongqiang Huang, Jacob Devlin, and Spyros Matsoukas. 2013. Bbn's systems for the chinese-english sub-task of the ntcir-10 patentmt evaluation. In *NTCIR-10*, Tokyo, Japan, June.

Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, NAACL '03, pages 48–54, Edmonton, Canada. Association for Computational Linguistics.

Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In Dekang Lin and Dekai Wu, editors, *Proceedings of EMNLP 2004*, pages 388–395, Barcelona, Spain, July. Association for Computational Linguistics.

Hai-Son Le, I. Oparin, A. Allauzen, J. Gauvain, and F. Yvon. 2011. Structured output layer neural network language model. In *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on*, pages 5524–5527, Prague, Czech Republic, May. IEEE.

Tomas Mikolov, Anoop Deoras, Daniel Povey, Lukas Burget, and Jan Cernock. 2011. Strategies for training large scale neural network language models. In *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on*, pages 196–201, Prague, Czech Republic, May. IEEE.

Jan Niehues and Alex Waibel. 2012. Continuous space language models using restricted boltzmann machines. In *Proceedings of the International Workshop for Spoken Language Translation*, IWSLT 2012, pages 311–318, Hong Kong.

Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51, March.

Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 160–167, Sapporo, Japan, July. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 311–

318, Philadelphia, Pennsylvania, June. Association for Computational Linguistics.

Holger Schwenk, Daniel Dchelotte, and Jean-Luc Gauvain. 2006. Continuous space language models for statistical machine translation. In *Proceedings of the COLING/ACL on Main conference poster sessions*, COLING-ACL '06, pages 723–730, Sydney, Australia, July. Association for Computational Linguistics.

Holger Schwenk, Anthony Rousseau, and Mohammed Attik. 2012. Large, pruned or continuous space language models on a gpu for statistical machine translation. In *Proceedings of the NAACL-HLT 2012 Workshop: Will We Ever Really Replace the N-gram Model? On the Future of Language Modeling for HLT*, WLM '12, pages 11–19, Montreal, Canada, June. Association for Computational Linguistics.

Holger Schwenk. 2007. Continuous space language models. *Computer Speech and Language*, 21(3):492–518.

Holger Schwenk. 2010. Continuous-space language models for statistical machine translation. *The Prague Bulletin of Mathematical Linguistics*, pages 137–146.

Le Hai Son, Alexandre Allauzen, Guillaume Wisniewski, and François Yvon. 2010. Training continuous space language models: some practical issues. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, EMNLP '10, pages 778–788, Cambridge, Massachusetts, October. Association for Computational Linguistics.

Le Hai Son, Alexandre Allauzen, and François Yvon. 2012. Continuous space translation models with neural networks. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, NAACL HLT '12, pages 39–48, Montreal, Canada, June. Association for Computational Linguistics.

Andreas Stolcke. 2002. Srilm-an extensible language modeling toolkit. In *Proceedings International Conference on Spoken Language Processing*, pages 257–286, November.

Omar F. Zaidan. 2009. Z-MERT: A fully configurable open source tool for minimum error rate training of machine translation systems. *The Prague Bulletin of Mathematical Linguistics*, 91:79–88.