Converting Continuous-Space Language Models into *N*-gram Language Models with Efficient Bilingual Pruning for Statistical Machine Translation

RUI WANG, Shanghai Jiao Tong University
MASAO UTIYAMA, National Institute of Information and Communications Technology
ISAO GOTO, NHK and National Institute of Information and Communications Technology
EIICHIRO SUMITA, National Institute of Information and Communications Technology
HAI ZHAO and BAO-LIANG LU, Shanghai Jiao Tong University

The Language Model (LM) is an essential component of Statistical Machine Translation (SMT). In this article, we focus on developing efficient methods for LM construction. Our main contribution is that we propose a Natural *N*-grams based Converting (NNGC) method for transforming a Continuous-Space Language Model (CSLM) to a Back-off *N*-gram Language Model (BNLM). Furthermore, a Bilingual LM Pruning (BLMP) approach is developed for enhancing LMs in SMT decoding and speeding up CSLM converting. The proposed pruning and converting methods can convert a large LM efficiently by working jointly. That is, a LM can be effectively pruned before it is converted from CSLM without sacrificing performance, and further improved if an additional corpus contains out-of-domain information. For different SMT tasks, our experimental results indicate that the proposed NNGC and BLMP methods outperform the existing counterpart approaches significantly in BLEU and computational cost.

Part of this work was done as Rui Wang visited NICT.
Rui Wang, Hai Zhao, and Bao-Liang Lu were partially supported by the National Natural Science Foundation of China (Grant No. 60903119, Grant No. 61170114, and Grant No. 61272248), the National Basic Research Program of China (Grant No. 2013CB329401), the Science and Technology Commission of Shanghai Municipality (Grant No. 13511500200), the European Union Seventh Framework Program (Grant No. 247619), the Cai Yuanpei Program (CSC fund 201304490199, 201304490171), and the art and science interdisciplinary funds of Shanghai Jiao Tong University, No. 14X190040031, and the Key Project of National Society Science Foundation of China, No. 15-ZDA041.
This article is partly motivated by Wang et al. [2013]. Additional contents include a novel bilingual LM pruning method for efficiently converting large LMs, and comparison experiments between our CSLM converting method and approaches developed by Arsoy et al. [2013, 2014] and Vaswani et al. [2013]. We also apply our methods on NIST corpus and 5 billion words NTCIR patent corpus.
Authors' addresses: R. Wang, Center for Brain-like Computing and Machine Intelligence, Department of Computer Science and Engineering, Shanghai Jiao Tong University, 800 Dongchuan Road, Shanghai, China, 200240; email: wangrui.nlp@gmail.com; M. Utiyama (corresponding author), Multilingual Translation Laboratory, National Institute of Information and Communications Technology, 3–5 Hikaridai, Keihanna Science City, Kyoto 619-0289, Japan; email: mutiyama@nict.go.jp; I. Goto, NHK Science & Technology Research Laboratories, 1-10-11 Kinuta, Setagaya-ku, Tokyo 157-8510, Japan; email: goto.i-es@nhk.or.jp; E. Sumita, Multilingual Translation Laboratory, National Institute of Information and Communications Technology, 3-5 Hikaridai, Keihanna Science City, Kyoto 619-0289, Japan; email: eiichiro.sumita@nict.go.jp; H. Zhao (corresponding author), (1) Center for Brain-like Computing and Machine Intelligence, Department of Computer Science and Engineering, Shanghai Jiao Tong University and (2) Key Laboratory of Shanghai Education Commission for Intelligent Interaction and Cognitive Engineering, Shanghai Jiao Tong University, 800 Dongchuan Road, Shanghai, China, 200240; email: zhaohai@cs.sjtu.edu.cn; Bao-Liang Lu, (1) Center for Brain-like Computing and Machine Intelligence, Department of Computer Science and Engineering, Shanghai Jiao Tong University and (2) Key Laboratory of Shanghai Education Commission for Intelligent Interaction and Cognitive Engineering, Shanghai Jiao Tong University, 800 Dongchuan Road, Shanghai, China, 200240; email: bllu@sjtu.edu.cn.

ACM 2375-4699/2016/01-ART11 $15.00
DOI: http://dx.doi.org/10.1145/2843942

CCS Concepts: • **Computing methodologies** → **Natural language processing; Machine translation**

Additional Key Words and Phrases: Machine translation, continuous-space language model, neural network language model, language model pruning

ACM Reference Format:
Rui Wang, Masao Utiyama, Isao Goto, Eiichiro Sumita, Hai Zhao, and Bao-Liang Lu. 2016. Converting continuous-space language models into *N*-gram language models with efficient bilingual pruning for statistical machine translation. ACM Trans. Asian Low-Resour. Lang. Inf. Process. 15, 3, Article 11 (January 2016), 26 pages.
DOI: http://dx.doi.org/10.1145/2843942

1. INTRODUCTION

Language Models (LMs) are important for various natural language processing tasks [Ma and Zhao 2012; Wang et al. 2015a; Jia and Zhao 2014], such as speech recognition and Statistical Machine Translation (SMT) [Zhang and Zhao 2013; Zhang et al. 2014; Xu and Zhao 2012; Zhao et al. 2013]. There are mainly two ways to improve the performance of LMs in SMT. One is to estimate the probabilities of *N*-grams better and the other is to use a larger corpus for LM construction.

Traditionally, Back-off *N*-gram Language Models (BNLMs) [Chen and Goodman 1996, 1999; Stolcke 2002] are widely used for probability estimation. For better probability estimation [Zhao et al. 2009a, 2009b, 2013; Zhao 2009], Continuous-Space Language Models (CSLMs), especially Neural Network based Language Models (NNLMs) [Bengio et al. 2003; Schwenk 2007; Le et al. 2011], have been used in SMT in recent years [Schwenk et al. 2006; Son et al. 2010; Schwenk et al. 2012; Son et al. 2012; Niehues and Waibel 2012]. These researches have shown that CSLMs can improve BLEU scores of SMT in comparison with BNLMs with the same sized training corpus. However, in practice, CSLMs have not been widely used in SMT due to a too high computational cost. Various methods have been proposed to tackle the training cost issues [Son et al. 2010; Schwenk et al. 2012; Mikolov et al. 2011], though there has been little progress on reducing the decoding cost, until recently. High time cost makes it difficult to use CSLMs in decoding directly. A common approach in SMT using CSLMs is a two-pass way: the first pass uses a BNLM in decoding to produce an *n*-best list, and in the second pass, a CSLM is used to rerank those *n*-best translations [Schwenk et al. 2006; Son et al. 2010; Schwenk et al. 2012; Son et al. 2012]. Another approach is to use Restricted Boltzmann Machines (RBMs) [Niehues and Waibel 2012] instead of using multilayer neural networks [Bengio et al. 2003; Schwenk 2007; Le et al. 2011]. Since the probability of an RBM can be calculated efficiently [Niehues and Waibel 2012], the RBM LM can be used in SMT decoding, but RBMs had only been used in a small-scale SMT task because of high training cost as well.

Vaswani et al. [2013] extend the Neural Probabilistic Language Model (NPLM) [Bengio et al. 2003] from two angles: (1) the rectified linear unit plays as the activation function that computes much more efficiently than Sigmoid or hyperbolic tangent units [Nair and Hinton 2010], and (2) Noise-Contrastive Estimation (NCE) [Gutmann and Hyvärinen 2010; Mnih and Hinton 2008] is adopted for model training, where the repeated summations over the vocabulary are not necessary. These two extensions enable building large-scale NPLMs efficiently. The main contribution of their work is to reduce the training cost of CSLM and apply the CSLM to a SMT decoder. However, they do not show any improvement in decoding speed in comparison with *N*-gram LMs. In this article, we propose a novel method to speed up decoding. The main idea of using NNLMs efficiently behind our method is to store the precalculated probabilities from CSLMs into *N*-gram format, which is quite different from that of Vaswani et al. [2013]. We will compare the decoding efficiency in Section 4.2.5. There are also some related researches that implement NNLMs or NN translation models for SMT [Auli et al. 2013; Kalchbrenner and Blunsom 2013; Liu et al. 2013; Zou et al. 2013; Devlin et al. 2014; Liu et al. 2014; Gao et al. 2014; Peng and Gildea 2014; Sundermeyer et al. 2014; Li et al. 2014; Cho et al. 2014; Lauly et al. 2014]. In particular, Devlin et al. [2014] propose a fast joint model of the LM and translation model using self-normalized NN and pre-computing the hidden layer. However, to integrate these techniques into SMT, almost all these existing methods more or less need modifications over the SMT decoder, while our method can be directly integrated into SMT without any modification.

In recent years, LMs in real application systems, such as Amazon Web Services, are increasing in size, because the performance (BLEU in SMT) of the LM is largely determined by the size of the corpus, and available corpora have become very large [Brants et al. 2007]. As BNLMs with cheaper computational cost can be trained from much larger corpora than CSLMs, to improve a BNLM by using a CSLM trained from a smaller corpus seems simpler and more realistic. Actually, a CSLM from a smaller corpus has been shown to enhance SMT [Papineni et al. 2002] reranking [Schwenk 2010; Huang et al. 2013]. In this article, we will demonstrate that a BNLM simulating a CSLM can improve BLEU scores in the first-pass decoding. The existing converting approaches [Deoras et al. 2011; Arsoy et al. 2013, 2014] mainly focus on speech recognition. Deoras et al. [2011] use a Recurrent Neural Network Language Model (RNNLM) [Mikolov et al. 2010] to generate a large amount of texts, which is generated by sampling words from the probability distributions calculated by RNNLM. They train a BNLM from the text using the interpolated Kneser-Ney smoothing. Arsoy et al. [2013] convert NNLMs into BNLMs using *artificial* higher-order N-grams constructed from lower-order N-grams. That is, all the words in the short-list are added to the tail of i-grams in the corpus in order to produce $(i+1)$-grams that may not be in the corpus. These $(i+1)$-grams are called *artificial N-grams*. However, most of the artificial N-grams do not have linguistic sense. For example, if a trigram is "*would like to*," then the extended 4-grams will be "*would like to* *," where "*" stands for any word in the short-list. A lot of meaningless 4-grams, such as "*would like to would*" or "*would like to cat*," will be generated. Although some of them can be pruned using a modified entropy-based method [Arsoy et al. 2014], a large part of meaningless 4-grams will still remain in the converted CSLM. In addition, a huge intermediate LM will be produced, which is commonly thousands of times larger than the original LM before pruning,[1] and the probabilities of its N-grams will have to be calculated using the CSLM. Consequently, the huge intermediate LM should be pruned into a specified size. Because the Arsoy method is very time and space consuming, it can only be applied to small corpora. For our proposed method, N-grams in the corpus are used as the input N-grams of CSLM. We therefore call them *natural N-grams*. Since no any intermediate LM will be constructed during converting, our method can be applied to very large corpora. Wang et al. [2014] and Wang et al. [2015b] propose a series of bilingual LM growing methods, which can be viewed as LM adaptation methods because they focus on special domain corpora.

To apply the proposed converting method to an extremely large corpora with billions of words, the time cost of CSLM converting is still an obstacle. Therefore, to prune LMs before converting LMs, without LM quality loss, is important for CSLM converting. A commonly used method for pruning LMs is *cutoff*, which simply discards N-grams whose frequencies are below a certain threshold. However, Heafield et al. [2013] have shown that the cutoff method may lead to significant performance loss in SMT. Another well-known method is entropy-based pruning [Stolcke 1998]. According to our best

[1]We are aware that this intermediate LM can be pruned parallel during converting. However, it still costs more space (depending on the threshold set for pruning) than the original one.

knowledge, this is the *state-of-the-art* pruning method. A shortcoming of this method is that it only uses monolingual information from the target language.

In a SMT system, every target language phrase is from a bilingual phrase table. So monolingual LM pruning methods may discard some useful N-grams, or keep some useless ones for SMT decoding. To overcome the deficiency of the existing pruning approaches, we propose a novel method that explicitly takes the bilingual phrase table into consideration for LM pruning. This Bilingual LM Pruning (BLMP) method is based on the following observation: the translation hypotheses of a phrase-based SMT system are a concatenation of the phrases from the phrase table. Suppose that two phrases "*would like to learn*" and "*Chinese as second language*" are in the phrase table and N-grams LM, respectively. In decoding, the two phrases may be connected together as "*would like to learn Chinese as second language*." However, the N-grams "*would like to learn Chinese*" or "*learn Chinese as second language*," which are concatenations of two phrases in decoding, may possibly be outside the N-gram LM. In other words, the connecting phrases may be defined to be the minimum set of N-grams that are necessarily used in decoding and ensure that all useful N-grams are retained.

The large LM constructed from an additional monolingual corpus can be pruned into a small LM without sacrificing its quality. The pruned LM can even outperform the large LM if the additional corpus introduces useful out-of-domain information. By using the proposed pruning and converting methods together, we can first prune the BNLM built from the large corpus, and then only convert the pruned LM. This will save a lot of computing time, and make it possible to use a very huge corpus in a reasonable time.

The remainder of this article is organized as follows. Section 2 describes BNLM and CSLM, and reviews CSLM converting methods using artificial N-grams. Section 3 proposes a novel method for converting a CSLM into a BNLM with efficient bilingual LM pruning. Section 4 reports experiments and analyzes the results. We summarize this article in Section 5.

2. LANGUAGE MODELS

This section will introduce the structure and probability estimation of a BNLM and CSLM, and the existing CSLM conversion methods using artificial N-grams.

2.1. Standard Back-Off *N*-gram Language Model

A BNLM predicts the probability of a word w_i given its preceding $(n-1)$ words history, $h_i = w_{i-n+1}^{i-1}$. 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 interpolated Kneser-Ney smoothing [Chen and Goodman 1996, 1999], the probability of w_i given h_i under a BNLM, $P_b(w_i|h_i)$, is defined as

$$P_b(w_i|h_i) = \hat{P}_b(w_i|h_i) + \alpha(h_i)P_b\big(w_i|w_{i-n+2}^{i-1}\big), \qquad (1)$$

where $\hat{P}_b(w_i|h_i)$ is a discounted probability and $\alpha(h_i)$ is the back-off weight.

2.2. CSLM Structure and Probability Calculation

A four-layer feedforward neural network based CSLM works in the following way: input layer projects all words in the context h_i onto projection layer (the first hidden layer); the second hidden layer and output layer achieve nonlinear probability estimation and calculate LM probability $P(w_i|h_i)$ for a given context [Schwenk 2007].

The CSLM calculates the probabilities of all words in the vocabulary of corpus given the context at once. However, due to the high computational complexity of calculating the probabilities of all words, the CSLM can only be used to calculate the probabilities of a small subset of the entire 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 inside and outside the short-list by assigning a neuron for that purpose. The probabilities of other words outside the short-list are still obtained from a BNLM [Schwenk 2007, 2010].

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

$$P(w_i|h_i) = \begin{cases} \frac{P_c(w_i|h_i)}{\sum_{w \in V_0} P_c(w|h_i)} P_s(h_i) & \text{if } w_i \in V_0 \\ P_b(w_i|h_i) & \text{otherwise,} \end{cases} \tag{2}$$

where V_0 is the short-list, $P_c(\cdot)$ is the probability calculated by the CSLM, $\sum_{w \in V_0} P_c(w|h_i)$ is the sum of probabilities over all the words in the short-list in the output layer, $P_b(\cdot)$ is the probability calculated by the BNLM as in Equation (1), and

$$P_s(h_i) = \sum_{v \in V_0} P_b(v|h_i). \tag{3}$$

We can also understand that the CSLM redistributes the probability mass of all words in the short-list. This probability mass is calculated by using the BNLM.

2.3. Using Artificial *N*-grams for CSLM Conversion

Since the CSLM can calculate the probabilities of N-grams outside the training corpus, one way to construct a large Converted CSLM (CONV) is to generate N-grams based on the training corpus. The existing methods use the short-list to construct those *new* N-grams.

The method in Arsoy et al. [2013] and Arsoy et al. [2014] adds all the words in the short-list after the tail word of the i-grams, to construct the $(i+1)$-grams. Then the probabilities of the $(i+1)$-grams are calculated using the $(i+1)$-CSLM.[2] As a result, a very large converted $(i+1)$-gram LM will be generated, and then this large LM needs to be pruned into a smaller size using entropy-based LM pruning [Stolcke 1998].[3] The $(i+2)$-grams are grown using $(i+1)$-grams, recursively. At last the CONV is interpolated with the original BNLM. Figure 1 illustrates the Arsoy method.

3. PROPOSED METHODS

3.1. Basic Idea

Since CSLM outperforms BNLM in probability estimation accuracy and BNLM outperforms CSLM in computational time, our key idea of converting CSLM into BNLM is to use the probabilities of N-grams calculated by CSLM to rewrite the probabilities of N-grams of BNLM. That is, N-grams from BNLM are used as the input of CSLM, and the output probabilities of CSLM together with the corresponding N-grams of BNLM constitute CONV.

The N-grams in the CONV can be viewed as a subset of the CSLM. In other words, this CONV can only represent part of the CSLM. To construct a larger CONV to simulate the CSLM and eventually improve the LM, the N-grams outside the corpus

[2]The $(i+1)$-CSLM indicates the $(i+1)$-order CSLM with i history words as input, which can calculate the probabilities of $(i+1)$-grams.
[3]They use the $P_b(w|h)$ of the background BNLM in Equation (3) to approximate the whole unpruned CONV, and this will save a lot of computational time.

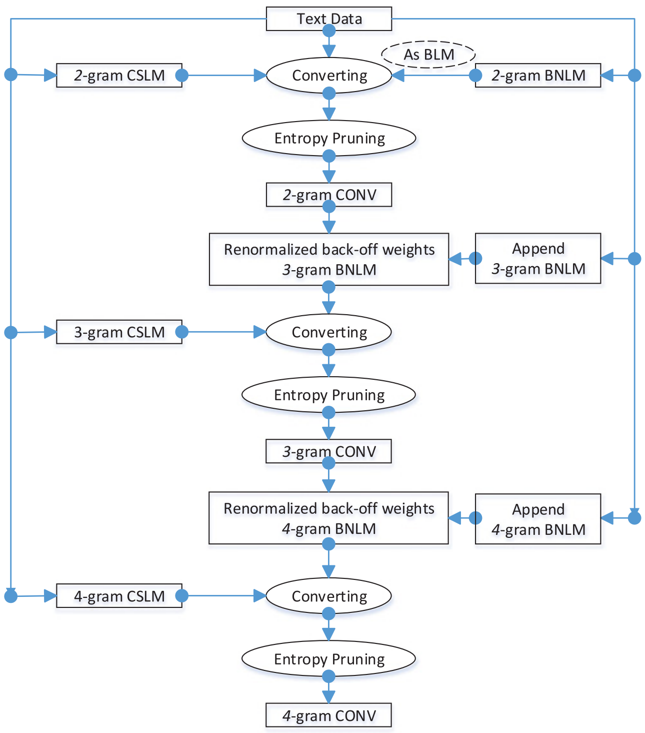

Fig. 1. Converting CSLM into BNLM using artificial N-grams [Arsoy et al. 2013, 2014]. The "BLM" indicates the Background LM. After calculating the probabilities of the bigrams ending with the words in the short-list, Arsoy et al. [2013] apply entropy-based pruning, producing a bigram pruned back-off CONV. The size of this model is determined by a pruning threshold. To create trigram background LM, they append trigrams from the trigram BNLM to the bigram pruned back-off CONV and renormalize the model. The recursive steps keep going on until the *4*-gram CONV is conducted.

must be considered.[4] The N-grams can be divided into artificial ones and natural ones. The converting method using artificial N-grams has been reviewed at Section 2.3.

In short, our proposed converting method uses natural N-grams that actually occur in corpus, while the Arsoy method uses artificial N-grams. For a small corpus such as 42M words/1M sentences, which is commonly used for SMT, the Arsoy method may perform well, because a large CONV can be generated without an extra corpus. However, the Arsoy method is hard to scale up, because the computational cost will increase dramatically as facing a very large corpus. In contrast, our method can work on an extremely large corpus with billions of words by selecting the most useful N-grams in a large LM for CSLM converting. That is, we can use our bilingual pruning method specially designed for SMT at first, and then apply our CSLM converting method.

3.2. Using Natural *N*-grams for CSLM Conversion

A drawback of the Arsoy method is overgeneration of poor-quality artificial N-grams, such as "*would like to would*," "*would like to cat*," and so on. Although part of these N-grams can be pruned by using an entropy-based strategy [Arsoy et al. 2014], a lot of useless N-grams will still remain in the converted CSLM. Instead of using artificial

[4]Both of the larger CSLM and CONV can improve the performance of LM. Because constructing a larger CSLM is very time consuming, constructing a larger CONV is a better choice.

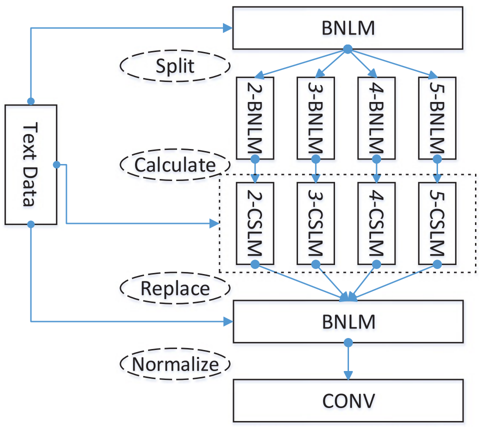

Fig. 2. Illustration of NNGC CSLM converting.

N-grams, we put forward generating natural N-grams from a larger in-domain corpus as the original training corpus.

The pipeline of NNGC is given in Figure 2. Our new approach can be summarized as the following four steps:

(1) **Split** the BNLM into different order N-grams ($n = 2,3,4,5$), and use the unigrams in BNLM as they are.
(2) Take N-grams ($n = 2,3,4,5$) in the BNLM as input of corresponding order ($n = 2,3,4,5$) CSLM[5] and **calculate** the probabilities of the N-grams by using Equations (2) and (3).
(3) **Replace** the probabilities of N-grams ($n = 2,3,4,5$) in the BNLM with the probabilities calculated by the corresponding CSLMs. Note that only the probabilities of N-grams ending with a word in the short-list need to be rewritten. When a BNLM trained from a larger corpus is rewritten, the N-grams in the BNLM often contain unknown words. In that case, the probabilities of unknown words remain unchanged as they are.
(4) **Renormalize** the probabilities and back-off weights $\alpha(h_i)$ in accordance with Equations (1) and (4) using SRILM's *-renorm* option [Stolcke 2002; Stolcke et al. 2011]. This is because the probabilities of N-grams in the BNLM have been changed, so the corresponding back-off weights should also be changed [Chen and Goodman 1996, 1999]. The renormalized $\alpha(h_i)$ can be calculated as

$$\alpha(h_i) = \frac{P_{\text{leftover}}}{\sum P_b\left(w_i|w_{i-n+2}^{i-1}\right)}, \tag{4}$$

where the P_{leftover} indicates the leftover probability mass of the (N-1)-gram and $P_b(w_i|w_{i-n+2}^{i-1})$ is the same as in Equation (1).

This converting method is hereafter referred to as *Natural N-gram based Converting (NNGC)*.

3.3. Bilingual LM Pruning

Though larger and larger LMs can be constructed with rapid development of computing resource, it is still very time consuming for those extremely large LM constructing

[5]Note that the N-gram CSLM means that the length of its history is $n - 1$.

Table I. Extracted LMs from Large LMs

LMs	Size	N-grams	PPL	BLEU
KN42M	3.0G	73.9M	108.8	32.35
EKN746M	3.0G	73.9M	101.3	32.46
EKN5B	3.0G	73.9M	102.6	32.28
KN746M	38.9G	935.2M	77.9	34.08
KN5B	163.1G	3945.3M	**73.4**	**34.32**

Note: The horizontal line between the LMs separates the extracted LMs (EKN) and the original LMs (KN).

tasks. Therefore, to develop an efficient LM pruning method without performance loss is still necessary and very important.

To overcome the drawback of the existing monolingual pruning approaches, we propose a method that explicitly takes the bilingual phrase table into consideration to identify the minimum set of N-grams that are potentially to be used in decoding.

3.3.1. The Essence in Large LMs. LM from a larger monolingual in-domain corpus[6] can improve both PPL and BLEU [Brants et al. 2007]. Our question is, which detailed factor yields the improvement of a large LM?

There are two main hypotheses about the improvement: (1) the probabilities in a larger LM are better estimated than those in a smaller LM; (2) a larger LM has more possibly useful N-grams, because a larger LM has more N-grams than a smaller LM.

To testify the first hypothesis, we first made three back-off 5-gram LMs from three NTCIR English corpora consisting of 42M, 746M, and 5B words, respectively. We call the corresponding LMs *KN42M*, *KN746M*, and *KN5B*. Both the 746M and 5B word corpora included the 42M word corpus. We used the 42M word parallel corpus in the Chinese-to-English SMT experiments. Detailed settings will be further described in Section 4. All of the LMs were made with the interpolated Kneser-Ney (KN) smoothing method without cutoff by using the vocabulary of KN42M.

We extracted the same sets of N-grams as contained in KN42M from KN746M and KN5B. The Extracted LMs were renormalized to produce EKN746M and EKN5B. These LMs (KN42M and EKN746M/5B) have the same N-grams with different probabilities and back-off weights. We conducted SMT experiments using these LMs under the same settings of other models. Their PPL and BLEU on test data are shown in Table I. In this table, "Size" means the sizes in gigabytes (G) and "N-gram" means the number of N-grams in millions (M) of uncompressed N-gram files. Table I shows that the extracted LMs obtain better PPL, but similar BLEU in comparison with KN42M.

The results of Table I demonstrate that higher PPL does not lead to much better BLEU, which suggests that the BLEU improvement with a larger LM may not directly come from the better probability estimation for the same N-grams in the smaller LM, but from more N-grams.

3.3.2. Connecting Phrase-Based Pruning. Because the translation hypotheses of a phrase-based SMT system are various concatenations of phrases from the phrase table, only the N-grams in the hypotheses are used in SMT decoding. Namely, only the phrases in the phrase table and the connecting phrases can be generated and used in the decoding, and these N-grams are considered as *useful*. N-grams other than these phrases are not used in decoding and they are considered as *useless*.

Although some N-grams in LMs constructed from a large monolingual corpus have linguistic senses, such as "mixed polarity compound" or "conventional lithographic etch

[6]Usually, target LMs are trained on a much larger corpus than the parallel corpus for SMT training.

process" in KN5B of Table I, they are neither in the phrase table constructed from a small bilingual corpus, nor connecting phrases. So these N-grams will never occur in SMT decoding.

Thus, we propose a new LM pruning method based on the phrase table. An N-gram that appears in a translation output satisfies either one of the following two conditions: (1) it is already included in a phrase in the phrase table or (2) it is the result of concatenating two or more phrases in the phrase table.

According to the preceding discussion, we design the following procedure:

Step 1. All the N-grams included in the phrase table should be maintained at first.
Step 2. The connecting phrases are defined in the following way.

Let w_a^b be a target-language phrase starting from the a-th word and ending with the b-th word, and $\gamma w_a^b \beta$ be a phrase including w_a^b as a part of it, where γ and β represent any word sequence or none. An i-gram phrase $w_1^k w_{k+1}^i$ $(1 \leq k \leq i-1)$ is a connecting phrase,[7] if

(1) w_1^k is the right (rear) part of one phrase γw_1^k in the phrase table, and
(2) w_{k+1}^i is the left (front) part of one phrase $w_{k+1}^i \beta$ in the phrase table.

For example, let "$a\ b\ c\ d$" be a 4-gram phrase; it is a connecting phrase if at least one of the following conditions holds:

(1) "γ " and "$b\ c\ d\ \beta$" are in the phrase table, or
(2) "$\gamma\ a\ b$" and "$c\ d\ \beta$" are in the phrase table, or
(3) "$\gamma\ a\ b\ c$" and "$d\ \beta$" are in the phrase table.

Therefore, the N-grams in the phrase table from Step 1 and the connecting phrases from Step 2 are kept, and then their probabilities are renormalized.

3.3.3. Tuning the Size of the Pruned LM. Using connecting phrases, a large LM can be pruned into a smaller one. During this pruning process, we may determine the size of the pruned LM. Thus, the *more useful* connecting phrases should be selected by ranking the occurrence probabilities of the connecting phrases in SMT decoding.

Suppose that $P(e|f)$ is the translation probability from f (source phrase) to e (target phrase), which can be calculated using bilingual parallel training data. In decoding, the probability of a target phrase e occurring in SMT can be estimated as

$$P_{\text{target}}(e) = \sum_{f} P_{\text{source}}(f) \times P(e|f), \tag{5}$$

where $P_{\text{source}}(f)$ is the occurrence probability of a source phrase, which can be calculated using source LMs in the training data. This $P_{\text{target}}(e)$ will absorb bilingual information rather than only using monolingual target LMs.[8] If a target phrase e is long, its aligned source phrase f will also be long and has a low occurrence probability $P_{\text{source}}(f)$. According to Equation (5), $P_{\text{target}}(e)$ will also be low.

[7]We are aware that connecting phrases can be applied to not only two phrases, but also three or more. However, the occurrence probabilities (which will be discussed in Equation (6) of the next subsection) of connecting phrases are approximately estimated. To estimate probabilities of longer phrases in different lengths will lead to serious bias, and the experiments also showed that using more than two connecting phrases did not perform well (not shown), so only two connecting phrases are applied in this article.

[8]We can prune the connecting phrases by directly discarding the target phrases e with low occurrence probabilities $P_{\text{target}}(e)$. In detail, we first prune the phrase table according to the target phrase probabilities, then the connecting phrases will be pruned accordingly. In the experiments (not shown), the performance of this method is not as good as the method in this article.

After all of the connecting phrases are generated, their occurrence probabilities are calculated to determine which connecting phrases should be discarded. For an i-gram connecting phrase $w_1^k w_{k+1}^i$, its probability can be roughly estimated as[9]

$$P_{\text{connect}}\left(w_1^k w_{k+1}^i\right) = \sum_{k=1}^{i-1} \left(\sum_{\gamma} P_{\text{target}}\left(\gamma w_1^k\right) \times \sum_{\beta} P_{\text{target}}\left(w_{k+1}^i \beta\right) \right), \tag{6}$$

where w_1^k is part of γw_1^k, w_{k+1}^i is part of $w_{k+1}^i\beta$, and γw_1^k and $w_{k+1}^i\beta$ are phrases in the phrase table. At last, a threshold for $P_{\text{connect}}(w_1^k w_{k+1}^i)$ is set, and only the connecting phrases whose occurrence probabilities are higher than the threshold are retained. In this way, the sizes of the pruned LMs are tuned to smaller proper ones.

The thresholds for each order N-grams are different. The distribution ratios of different order N-grams ($n = 2, 3, 4, 5$) in pruned LMs follow the small LM. It should be noted that the unigrams are not pruned because we use the vocabulary of small corpus for all LMs. In practice, the N-grams in different orders are ranked independently. The top N-grams in different orders of large LMs, such as KN5B, are selected with the same distribution ratio of small LMs, such as around 46: 148: 244: 295 of KN42M in NTCIR task (please refer to Section 4.1 for setting up LMs).

3.4. Comparison of Computational Complexity

3.4.1. Computational Complexity of CSLM Converting Methods. The time complexity of the CSLM is given as [Schwenk 2007]

$$(n-1) \times P \times H + H + (H \times N) + N, \tag{7}$$

where n is the order of N-grams, P is the size of the projection layer, H is the size of the hidden layer, and N is the size of the output layer. The original N is equal to the size of the vocabulary ($|V|$). To reduce the time complexity, the short-list V_0 in Equation (2), which is a subset of the whole vocabulary, is used as the output layer N. The probabilities of other words outside the short-list in the output layer will be calculated using the background BNLM.

Arsoy et al. have applied their method to 4-gram LM for speech recognition [Arsoy et al. 2013, 2014]. The Arsoy method needs to add every word in the short-list after the tail word of i-gram to construct the $(i + 1)$-gram. The short-list usually includes thousands of words, so the generated $(i + 1)$-grams will be thousands larger than the i-grams before they are pruned. Because the higher-order N-grams commonly contain more N-grams, computational cost significantly grows as the order of N-gram increases. The limited size of vocabulary and low-order N-gram make the Arsoy method applicable to speech recognition. However, SMT needs a higher-order BNLM (5-gram LM as a common setting for SMT, compared with 4-gram LM for speech recognition). Thus, the Arsoy method is not feasible for large BNLMs because the converting time for construction is basically prohibitive.

Since we rewrite the N-grams from the original BNLM, we only need to calculate the same number of N-grams as the original BNLM, instead of thousands more times of the original BNLM as that in Arsoy et al. [2013] and Arsoy et al. [2014]. The NNGC method thus requires a much lower computational cost, which makes NNGC have good scalability on large corpora.

[9]In practice, we have also tried to measure the occurrence probability of a target phrase using all the features in a phrase table, such as phrase translation probabilities, word lexical weights, and phrase length penalty. However, the performance will decrease (BLEU decrease around 0.3 in comparison with using Equation (6)). So we apply the current method, although it is straightforward.

Table II. Computation Complexity

LMs	Time Complexity
BNLM	$n \times \mathbb{C}(mapping)$[a]
CSLM	$(n-1) \times P \times H + H \times \mathbb{C}(tanh) + H \times N + N \times \mathbb{C}(softmax)$[b]
NPLM	$(n-1) \times P \times H + H \times \mathbb{C}(RLU)$[c] $+ H \times N + N$
Arsoy	$n \times \mathbb{C}(mapping)$
CONV	$n \times \mathbb{C}(mapping)$

[a]$\mathbb{C}(\cdot)$ stands for time of calculating functions or methods, and *mapping* stands for searching N-grams in BNLM. [b]n is order of N-grams, P is size of projection layer, H is size of hidden layer, and N is size of output layer. Because N is much larger than P or H, $N \times \mathbb{C}(softmax)$ is the most time-consuming part. [c]RLU stands for rectified linear units.

3.4.2. Computational Complexity of CSLM Methods in Decoding. To improve the efficiency of CSLM in SMT decoding, NPLMs [Vaswani et al. 2013] use rectified linear units [Nair and Hinton 2010], where its activations are cheaper to compute in comparison with sigmoid or hyperbolic tangent units. They also use NCE [Gutmann and Hyvärinen 2010; Mnih and Hinton 2008] in the output layer, instead of softmax, which requires a summation over all the units in the output layer in training, to produce "approximately normalized probabilities" and "simply ignore normalization" in decoding [Vaswani et al. 2013]. The Arsoy converting method and the proposed NNGC method both store the N-grams and their precalculated probabilities from CSLM into N-gram LM format. So computation complexity is similar with the N-gram LM. In summary, Table II shows the computation complexity of all the methods mentioned previously.[10]

3.4.3. Computational Complexity of LM Pruning Methods. For the time complexity, the entropy-based pruning method needs to calculate all the related probabilities and back-off weights for the N-grams with the same history h and back-off history h'. Our proposed BLMP only takes the N-grams themselves into account, which allows parallel computation speedup.[11]

For the space complexity, the entropy-based pruning method needs to load all the related N-grams of the LM into the memory, which is hard to implement if a very huge LM (such as larger than 1T) needs to be pruned. In contrast, BLMP only needs to take the N-grams used for pruning, rather than loading any other N-gram.

Therefore, BLMP has advantages in both time and space complexities.

4. EXPERIMENTS

In this section, comparison experiments between our proposed methods and the existing approaches were conducted and the results were analyzed. Our implementation of the proposed NNGC CSLM converting method has been available online.[12]

[10]Devlin et al. [2014] propose a joint model of LM and translation model using self-normalized NN and precomputing the hidden layer. Because their method is a joint model rather than LM only, we did not conduct comparison experiments.

[11]Arsoy et al. proposed a modified pruning method especially for CSLM converting, which can run parallel especially in the CSLM converting method with a short-list [Arsoy et al. 2013, 2014]. The Arsoy method can only work recursively parallel, not fully parallel. That is, they can only prune i-grams parallel, and $(i+1)$-grams parallelly one after another, but cannot prune both N-grams simultaneously. BLMP can prune N-grams in different orders parallel. In addition, their method cannot be used in the common LM pruning, where there is no short-list for CSLM converting. This is because the neuron in the CSLM of calculating the probabilities of N-grams inside or outside the short-list takes part in their pruning method; however, for a common LM pruning, there is no this neuron for the short-list.

[12]It is available at http://bcmi.sjtu.edu.cn/wangrui/program/nngc.zip.

4.1. Experiment Setup

We used the patent bilingual 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 1M, 2,000, and 2,000 sentences, respectively.

The same settings of the NTCIR-9 Chinese-to-English translation baseline system [Goto et al. 2011] were followed besides the part of LMs for the purpose of comparison. 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.[13]

4.1.1. Corpus. Three corpora with different sizes were used:

(1) Corpus42M: The English side of the 1M parallel sentence training data with 42M words in NCTIR-9.
(2) Corpus746M: The English text from the 2005 U.S. patent data distributed in the NTCIR-8 patent translation task [Fujii et al. 2010], which consisted of 746M words.
(3) Corpus5B: English texts extracted from U.S. patent data (from 1993 to 2005) with 5B words.

It should be noted that the Corpus42M was added to Corpus746M and Corpus5B.

4.1.2. BNLM. All the BNLMs were trained using SRILM [Stolcke 2002; Stolcke et al. 2011] and the same vocabulary of Corpus42M was used. As the baseline BNLM, we trained a 5-gram BNLM with interpolated Kneser-Ney smoothing using Corpus42M. We did not discard any N-grams in training this model. That is, we did not use count cutoffs. This BNLM was called *KN42M*.

We also trained a larger 5-gram BNLM with interpolated Kneser-Ney smoothing by using Corpus746M, discarding 3,4,5-grams that occurred only once when we created KN746. This BNLM was called *KN746M(cutoff)*.

The 5-gram LMs were trained with the interpolated KN and Good-Turing (GT) smoothing methods,[14] on the Corpus5B without cutoff. These LMs were called *KN5B* and *GT5B*, respectively. The KN/GT5B would be pruned into smaller ones in different sizes with different pruning methods.

4.1.3. CSLM. A 5-gram CSLM was trained on the same 1M training sentences (Corpus42M) using the CSLM toolkit [Schwenk 2010]. The settings for the CSLM were as follows: projection layer of dimension 256 for each word, hidden layer of dimension 384, and output layer (short-list) of dimension 8,192, which were recommended in the CSLM toolkit and Wang et al. [2013]. This CSLM was called CSLM42M, in which KN42M is used as the background BNLM.

Arsoy et al. [2013] and Arsoy et al. [2014] used around 55M words as the corpus, including 84K words as vocabulary, and 20K words as the short-list. They only constructed a 4-gram LM. In this article, we used around 42M words as the corpus, including 456K words as vocabulary, and 8K words, which covers 92.89% of words in the training corpus,[15] as the short-list for both NNGC and the Arsoy method. We adopted

[13]It is available at http://www.itl.nist.gov/iad/mig/tests/mt/2009/.
[14]We also applied BLMP for Witten-Bell smoothing, and the results were similar to Good-Turing.
[15]The size of the short-list was selected according to the corpus by experiments before we constructed the CSLM. That is, we chose 1K (default setting by the CSLM toolkit), 4K, 8K, 16K, and 24K as the short-list, and then we tested the coverage probability that the words in the training corpus were in the short-list. The results were around 93% (this is because the frequency of words was different, and the rare words seldom appear although they were in the vocabulary) of the words will be in the short-list for 8K as short-list, and

a similar size setting for the corpus and short-list as [Arsoy et al. 2013] and [Arsoy et al. 2014]. Our vocabulary was much larger than theirs, because the whole vocabulary must be used for decoding in SMT, compared with only a small vocabulary used in speech recognition. The ratio of $|V_0|/|V|$ affects how much time cost will reduced for the output layer. In Arsoy's experiments, $|V|$ was small (84K), and nearly 75% time was saved by using 20K as the short-list. For NNGC, $|V|$ was 456K and the short-list was 8K. Therefore, nearly 98% time cost was reduced.

It should be noted that every setting of the CSLM for all of the methods was the same for fair comparison in this article.

4.1.4. SMT Models. Fourteen standard SMT feature scores are used: four translation model scores, one phrase pair number penalty score, one word penalty score, seven distortion scores, and one LM score.

All the models were trained on Corpus42M, except LMs. The weights between the 14 standard SMT features were tuned using MERT independently.

All of the experiments were conducted using the same computer with 2.70GHz CPUs.

4.2. CSLM Conversion

4.2.1. Setup for Proposed NNGC Method. The CSLM can only predict the probability of the same order i-grams as they are. That is, the bigram CSLM can only calculate the probabilities of bigrams. So different order CSLMs must be constructed if we want to convert a *5*-gram LM.

The 2,3,4,5-CSLMs were trained with the same setting for the CSLM described in Section 4.1.3 on Corpus42M using the CSLM toolkit [Schwenk 2010], and KN42M as the background BNLM. These CSLMs were called 2,3,4,5-CSLM42Ms, respectively.

By using the method described in Section 4.2, we rewrote KN42M with CSLM42Ms. This rewritten BNLM was interpolated with the original KN42M. 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 interpolated BNLM in the SMT system to tune the weight parameters on the first half (1,000 sentences) of the development data. Subsequently, we selected the interpolation weight that obtained the highest BLEU score on the second half of the development data. As the interpolation weights were determined, we applied MERT again to the whole 2,000 sentence development data to tune the weight parameters.[16] This converted BNLM was called CONV-KN42M.

4.2.2. Setup for Compared Methods. The same CSLM42Ms were used for the Arsoy method. All the words in short-list were added after the tail word of i-grams,[17] to construct the $(i + 1)$-grams. Then, the probabilities of $(i + 1)$-grams were calculated using the $(i + 1)$-CSLM. So a large converted $(i + 1)$-gram LM will be grown, and then

94% or 95% if we used 16K or 24K. Please note this 1% or 2% improvement was not PPL or BLEU, it was just the probability of whether to use CSLM or BNLM to calculate the probabilities of N-grams. That is, 93% of the N-grams will be calculated using the CSLM and 7% using BNLM. The time computational complexity for the CSLM is nearly linear with the size of vocabulary. If we use a 20K short-list, it will take 2.5 times more time for training and converting CSLM. Therefore, we chose 8K as the short-list for both methods, because more than 8K V_0 did not affect the coverage of words in the short-list so much, but it will take much more time.

[16]We are aware that the interpolation weight might be determined by minimizing the perplexity on the development data; however, this method did not work well according to our experiments. So we optimize the weights by directly maximizing the BLEU score on the development data.

[17]In practice, the probabilities of all the target/tail words in short-list for the history i-grams can be calculated at the same time by neurons, which will save some time.

Table III. Comparison Experiments between Converting Artificial *N*-grams and Natural *N*-grams

LMs	Size	*N*-grams	PPL	BLEU	Ratio in Phrase Table
KN42M	3.0G	73.9M	108.8	32.35	100%
Arsoy42M (pruned)	3.1G	71.8M	104.7	32.38	45.9%
Arsoy42M (interpolated)	5.4G	140.1M	103.5	32.91	52.7%
CONV-KN42M	3.0G	73.9M	106.3	32.88	100%
CONV-KN42M (interpolated)	3.0G	73.9M	106.1	33.07	100%

Note: The "Ratio in Phrase Table" indicates what percentage of the *N*-grams are in the phrase table.

Table IV. Significance Tests for Comparison between Converting Artificial *N*-grams and Natural *N*-grams

	Arsoy42M (interpolated)	CONV-KN42M	Arsoy42M (pruned)	KN42M
CONV-KN42M (interpolated)	$>$	$>$	$\gg$	$\gg$
Arsoy42M (interpolated)		$-$	$\gg$	$\gg$
CONV-KN42M			$\gg$	$\gg$
Arsoy42M (pruned)				$-$

Note: The marks ($\gg$, $>$, and $-$) indicate whether the LM to the left of a mark was significantly better than that above the mark at a certain level. "$\gg$": significantly better at significance level $\alpha = 0.01$; "$>$" : $\alpha = 0.05$; "$-$": not significantly better at $\alpha = 0.05$.

it needs to be pruned into the same size as the original $(i + 1)$-grams using entropy-based LM pruning. The $(i+2)$-grams were grown using $(i + 1)$-grams recursively. At last, the CONV was interpolated with the original BNLM[18] using the same methods in Section 4.2.1.

4.2.3. Comparison between Converting Artificial N-grams and Natural N-grams. The KN42M was the BNLM trained on Corpus42M using the interpolated Kneser-Ney smoothing. The Arsoy42M was the converted CSLM42M using the Arsoy method, which used artificial *N*-grams. The converted CSLM and the original BNLM were interpolated in Arsoy et al. [2013] and Arsoy et al. [2014]. Since the *N*-grams in the converted CSLM and BNLM were quite different, the interpolated LM was larger than both of them. Consequently, the interpolated LM was pruned into the same size as the original BNLM. The CONV-KN42M was the converted CSLM42M using NNGC, which used natural *N*-grams from KN42M, and CONV-KN42M (interpolated) was the CONV-KN42M interpolated with KN42M. All the preceding LMs were trained on Corpus42M.

The comparisons of the Arsoy method and ours are shown in Table III. We performed the paired bootstrap resampling test [Koehn 2004],[19] and sampled 2,000 samples for each significance test. Table IV showed statistical significance test.

[18]Since this interpolated LM was larger than the original one, it was pruned into the same size as the original BNLM according to Arsoy et al. [2014].
[19]We used the code available at http://www.ark.cs.cmu.edu/MT.

Table V. Converting Efficiency

LMs	Converting Time
Arsoy's method	24 hours 24 minutes
NNGC CONV	41 minutes

Table VI. Decoding Efficiency and Performance

LMs	Decoding Time (sec.)	BLEU
KN42M	15.3	32.25
Arsoy42M (pruned)	15.3	32.38
Arsoy42M (interpolated)	15.9	32.91
CONV-KN42M (our)	**15.2**	33.07
CSLM42M	186.5	**33.16**
NPLM42M	50.4	32.78
NPLM42M (renormalized)	456.8	32.85

From the results in Tables III and IV, we can obtain the following observations:

(1) Both the Arsoy method and NNGC improved the PPL compared with the original BNLM. This indicated that using the CSLM for converting can improve the probabilities estimation.
(2) The BLEU score of CONV-KN42M was better than Arsoy42M (pruned) by around 0.5, and CONV-KN42M (interpolated) was slightly better than Arsoy42M (interpolated) by 0.16. For the ratio of N-grams in the phrase table, the Arsoy method can generate more artificial N-grams; however, some N-grams, which were important for SMT (such as the N-grams in the phrase table), were pruned during their converting. For NNGC, all the N-grams in the CONV were inside the phrase table.

4.2.4. Comparison for Converting Efficiency. In this subsection, we showed the converting efficiency of the Arsoy method and NNGC. 1M trigrams were used as the input for the CSLM42M for both methods. In converting, the Arsoy method will generate many more N-grams and then a large part of these N-grams should be pruned.[20] The converting time is shown in Table V.

Table V shows that Arosy's converting method took much more time (nearly 40 times) than ours. The Arsoy method generated all the possible N-grams ending with the words in the short-list (usually thousands of times of the original one) and then pruned them into smaller ones. Although the probabilities of all the tail words in the short-list can be calculated at the same time using neurons in the output layer in the CSLM, the Arsoy method still took much more time than ours. For our NNGC method, the same N-grams with the N-grams in the BNLM were converted.

4.2.5. Comparison for Decoding Efficiency. Vaswani et al. [2013] applied several techniques to speeding up NPLM in SMT; meanwhile, the Arsoy method and NNGC method stored the probabilities of the CSLM into the BNLM format. In this subsection, we compare the decoding efficiency of BNLM, Arsoy, CONV, CSLM, and NPLM. The numbers of hidden layers were set the same as the CSLM and other settings followed the default setting of the NPLM toolkit [Vaswani et al. 2013], with all the same settings as in SMT decoding. The 2,000 sentences of the NTCIR-9 test data were used as the evaluation data. The decoding time and SMT performance are shown in Table VI.

[20]The pruning actually takes a lot of time, but we did not take it into account for fair comparison of converting time. In contrast, NNGC will only produce the same sized N-grams.

Table VII. Comparison of Different Pruning Methods on the 5B Corpus for KN Smoothing

LMs	Size	N-grams	PPL	BLEU
KN5B	163.1G	3945.3M	73.4	34.32
KN5B-Cutoff	36.1G	956.7M	74.8	33.89
KN5B-Entropy-1	80.0G	2029.8M	74.0	34.04
KN5B-Entropy-2	41.3G	1099.5M	77.9	33.67
KN5B-Entropy-3	21.8G	607.9M	82.8	33.41
KN5B-Entropy-4	8.2G	201.9M	93.8	33.16
KN5B-Connect-1	84.0G	2195.9M	79.8	34.58^{++}
KN5B-Connect-2	43.8G	1192.9M	82.0	33.92^{+}
KN5B-Connect-3	19.3G	555.7M	86.9	33.79^{++}
KN5B-Connect-4	8.4G	208.0M	94.2	33.38^{++}
KN5B-Inter-1	119.9G	2995.5M	**71.8**	$\mathbf{34.66}^{++}$
KN5B-Inter-2	71.0G	1871.4M	**72.4**	$\mathbf{34.44}^{++}$
KN5B-Inter-3	35.5G	985.8M	**75.2**	$\mathbf{34.28}^{++}$
KN5B-Inter-4	15.9G	459.0M	**81.0**	$\mathbf{33.93}^{++}$

Table VIII. Comparison of Different Pruning Methods on 5B Corpus for GT Smoothing

LMs	Size	N-grams	PPL	BLEU
GT5B	162.8G	3945.3M	83.5	34.58
GT5B-Cutoff	37.8G	956.7M	82.3	34.07
GT5B-Entropy-1	77.7G	2055.5M	83.3	34.41
GT5B-Entropy-2	43.8G	1206.3M	82.7	34.12
GT5B-Entropy-3	20.3G	576.3M	81.8	34.41
GT5B-Entropy-4	8.5G	229.7M	82.8	34.07
GT5B-Connect-1	85.2G	2195.9M	84.2	**34.42**
GT5B-Connect-2	43.6G	1192.9M	84.5	34.14
GT5B-Connect-3	19.2G	555.7M	87.5	33.58^{--}
GT5B-Connect-4	8.3G	208.0M	92.1	33.97
GT5B-Inter-1	115.1G	2935.1M	**80.5**	34.33
GT5B-Inter-2	70.3G	1894.3M	**78.4**	$\mathbf{34.50}^{++}$
GT5B-Inter-3	33.5G	947.4M	**77.3**	$\mathbf{34.67}^{+}$
GT5B-Inter-4	14.3G	409.9M	**78.3**	$\mathbf{34.42}^{++}$

From the results of Table VI, we reach the following conclusions:

(1) The decoding times using CONV-KN42M and BNLM (KN42M) were nearly all the same. This indicated that the NNGC method can run as fast as the BNLM does.
(2) The decoding time using CSLM was much slower than BNLM, and the decoding time using NPLM (without normalization) was much faster than CSLM and much slower than BNLM. This showed the advantage of NNGC in decoding efficiency.
(3) In comparison with the LMs with similar decoding time, the BLEU of CONV was slightly better than the Arsoy method and much better than KN42M.
(4) In comparison with NNLMs directly used in decoding, which were much slower than BNLM, the BLEU of CONV was slightly better than NPLM and slightly worse than CSLM.

4.3. LM Pruning

In this subsection, we compare the performance of pruned LMs using different methods. The experiments on LM pruning were divided into four groups: the original, cutoff pruning, entropy-based, and BLMP methods. The experiment results are listed in Tables VII and VIII.

For the cutoff LM pruning, the default setting 1-1-2-2-2 in SRILM was applied, which means we did not cut off unigrams and bigrams, but cut off *N*-grams whose frequency was less than two for the higher-order *N*-grams. They are represented by KN/GT5B-Cutoff in Tables VII and VIII. KN/GT5B-Connect-* (where * stands for any ID) were LMs tuned into different sizes using the BLMP method. The sizes of the entropy-based pruned LMs, KN/GT5B-Entropy-*, were tuned to be similar to those of BLMP KN/GT5B-Connect-* pruned LMs by setting thresholds appropriately.

The corresponding (in similar size) entropy-based pruned LMs and BLMP pruned LMs, such as KN5B-Connect-1 and KN5B-Entropy-1, were interpolated with the parameter 0.5 into KN5B-Inter-1 using SRILM. The IDs after the pruned LMs (1/2/3/4) indicate the different sizes (in decreasing order) of LMs for the same pruning method.

We also performed the paired bootstrap resampling test. Two thousand samples were used for each significance test. The marks at the upper right of the BLEU score indicated whether the LMs were significantly better/worse than the entropy-based pruned LMs with the same IDs ("++/−−": significantly better/worse at $\alpha = 0.01$; "+/−" : $\alpha = 0.05$; no mark: not significantly better/worse at $\alpha = 0.05$).

From Tables VII and VIII, we made the following observations:

(1) KN5B-Connect-1 and GT5B-Connect-1 were the pruned LMs with all the connecting phrases. That is, all the connecting phrases and phrases in the phrase table were kept and all the other phrases were pruned. The BLEUs of these two pruned LMs were similar to the original ones (KN/GT5B). This indicated that BLMP kept all the useful information and discarded all the useless information from the original huge LM.
(2) For the BLMP method, as LMs became smaller, the PPL increased and BLEU decreased accordingly for nearly all the cases in both KN and GT smoothing. Since PPL and BLEU were positively correlated, PPL can be used to predict the performance of pruned LMs in SMT. This also indicated that we could use PPL to set up a threshold for tuning the size of pruned LMs.
(3) For the entropy-based pruning method, as LMs became smaller, PPL did not always increase and BLEU did not always decrease accordingly, and the PPL and BLEU of some pruned LMs even decreased/increased. It was interesting that some small pruned LMs performed better than larger LMs in GT smoothing.
(4) For KN smoothing, the entropy-based pruned LMs obtained better PPL on the LMs with similar sizes, but the BLMP pruned LMs always obtained significantly better BLEU.[21] These showed that BLMP performed significantly better in SMT for KN smoothing, although with worse PPL.
(5) For GT smoothing, the entropy-based method also obtained better PPL on the LMs with similar sizes. The BLMP method obtained similar BLEU on large LMs and the entropy-based method obtained better BLEU on small LMs.
(6) Some interpolated LMs obtained comparable BLEU, with comparable 1/4 of the size, compared with the unpruned LMs. The interpolated LMs obtained better PPL and BLEU in nearly all cases, compared to either the entropy-based or BLMP pruned LMs of the same IDs.
(7) By comparing *N*-grams in the entropy-based pruned LMs and BLMP pruned LMs, such as KN5B-Entropy-* and KN5B-Connect-*, the percentage of the overlapping *N*-grams between the entropy-based pruned LMs and BLMP pruned LMs in similar sizes were 60.6%/38.2%/29.2%/10.6%, corresponding to KN5B-Entropy-/KN5B-Connect-1/2/3/4. The overlapping situation was similar for GT smoothing.

[21]Some researchers found that KN smoothing performs badly in the entropy-based LM pruning [Siivola et al. 2007; Chelba et al. 2010; Roark et al. 2013]. The PPL of entropy-based pruned LMs for KN smoothing increased aggressively as their sizes decreased, compared with other smoothing methods.

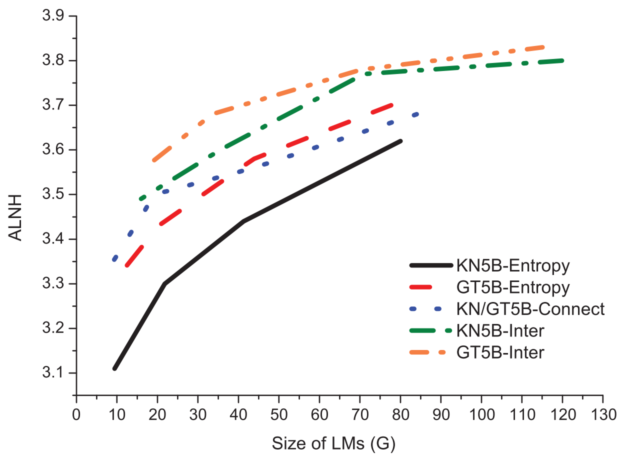

Fig. 3. ALNH of different pruning methods in SMT decoding. Because the N-grams in BLMP pruned LMs for KN and GT (KN/GT5B-Connect) are all the same and only possess different probabilities and back-off weights, their ALNH are also the same.

This indicated that these two pruning methods were quite different, and the pruned LMs had increasingly different N-grams as the pruning keeps going on.

4.3.1. Discussions. The preceding results showed that BLMP performed well in SMT. In this subsection, which kind of N-grams were useful during decoding are investigated. We calculated the log-probabilities of every reference sentence of test data to simulate the performance of 5-gram LMs in SMT decoding, and counted the ratio of different order N-grams used for each pruned LM. Then, the Average Length of the N-grams Hit (ALNH) in SMT decoding for different LMs was calculated as follows:

$$ALNH = \sum_{i=1}^{5} P_{\text{i-gram}} \times i, \tag{8}$$

where $P_{\text{i-gram}}$ means the ratio of the i-grams hit in SMT decoding. The results were illustrated in Figure 3.

As shown in Figure 3, ALNH in SMT decoding of BLMP pruned LMs was longer than that of the entropy-based pruned LMs for KN smoothing, and ALNH of BLMP pruned LMs and the entropy-based pruned LMs were similar for GT smoothing. ALNH of interpolated LMs was longer than both of the two methods for KN and GT smoothing.

As we know, LM refers to the probabilities of $(i-1)$-grams together with the adjusted weights using back-off, if no corresponding i-grams are hit. The preceding statistics indicated that more high-order N-grams were hit for BLMP pruned LMs in SMT compared with the entropy-based pruned LMs. In other words, less back-off was applied with the BLMP pruned LMs in SMT decoding.[22]

There was also a positive correlation between ALNH in Figure 3 and BLEU in Tables VII and VIII. Most of the LMs with larger ALNH also obtained higher BLEU in SMT. This indicated that useful N-grams in SMT were not only the N-grams used in the decoding, but also those useful N-grams that were *long*. Although our current method only focuses on probability, the results suggested that we should also focus on lengths of the N-grams for pruning in the future work.

[22]For the 5-gram LM, if 1/2/3/4-grams were hit in decoding, it indicated that back-off was applied.

Table IX. Comparison of BLEU Scores for LMs with Different Sizes

LMs	Size	*N*-grams	PPL	1st Pass	Rerank	ALNH	Weights
KN42M	3.0G	73.9M	108.8	32.35	32.96	3.03	0.055
CONV-KN42M	3.0G	73.9M	106.1	33.07	33.47	3.03	0.060
KN746M (cutoff)	7.5G	183.7M	95.1	33.27	33.78	3.43	0.073
CONV-KN746M	7.5G	183.7M	**92.9**	**33.59**	**33.91**	3.43	0.081

Table X. Significance Tests for LMs with Different Sizes (for Section 4.4.1)

	KN746M (rerank)	CONV-KN746M (1st)	CONV-KN42M (rerank)	KN746M (cutoff) (1st)	CONV-KN42M (1st)	KN42M (rerank)	KN42 (1st)
CONV-KN746M (rerank)	–	$\gg$	$\gg$	$\gg$	$\gg$	$\gg$	$\gg$
KN746M (rerank)		–	$>$	$\gg$	$\gg$	$\gg$	$\gg$
CONV-KN746M (1st)			–	$>$	$\gg$	$\gg$	$\gg$
CONV-KN42M (rerank)				$\gg$	$\gg$	$\gg$	$\gg$
KN746M (cutoff) (1st)					–	$\gg$	$\gg$
CONV-KN42M (1st)						–	$\gg$
KN42M(rerank)							$\gg$

Note: The "1st" is short for "1st pass." The marks ($\gg$, $>$, and –) indicate whether LM to the left of a mark was 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 $\alpha = 0.05$.

4.4. Converting Large LM

As mentioned in Section 1, a large additional corpus may let the LMs better. In this subsection, experiments on converting large LMs were conducted on both NTCIR and NIST datasets.

4.4.1. Effects of Corpus Size. In Section 4.2, the results showed that the proposed NNGC CONV outperformed the original BNLM and Arsoy method on a small corpus containing 42M words. In this subsection, we show the results of the NNGC method on a large corpus containing around 1B words.

Both the Corpus42M and the Corpus746M were used for this experiment. The KN42M and CONV-KN42M were the same as in Table III. We obtained the CONV-KN746M by rewriting the KN746M (cutoff) with CSLM42M in the same way as CONV-KN42M.

Table IX showed BLEU scores on the test data. The figures in the "1st Pass" column of Table IX show the BLEU scores in the first pass decoding when we changed LMs. The figures in the "Reranking" column show BLEU scores when we applied CSLM42M to rerank the 100-best lists for different LMs.[23] When we applied CSLM42M for reranking, the CSLM42M score was added as the additional 15th feature. The weight parameters were tuned by using Z-MERT [Zaidan 2009].

The statistical significance test is listed in Table X.

[23]In practice, we also conducted 1,000-best lists reranking (not shown), and the results are similar to 100-best lists reranking.

From the results of Tables IX and X, we can observe:

(1) Reranking by applying the CSLM42 increased the BLEU scores for all the first-pass LMs. This observation is consistent with those of previous work [Schwenk 2010; Huang et al. 2013].
(2) The CONV42M was better than the KN42M for both first pass and reranking. This also held in the case of CONV746M and KN746M. This indicated that the NNGC method improved BNLMs, even if the underlying BNLM was trained on a larger corpus than that used for training CSLM. As described in the Introduction, this is very important because BNLMs can be trained from much larger corpora (containing around 1B words here and 5B words in the next subsection) than those that can be used for training CSLMs.
(3) The first pass of CONV42M and CONV746M (33.07 and 33.59) were comparable with those of the reranking results of KN42M and KN746M (32.96 and 33.78), respectively. That is, there were no significant differences between these results. This indicates that the NNGC method preserves the performance of reranking using CSLM.
(4) The last column in Table IX indicates the tuned SMT feature weights for different LMs. There is also a positive correlation between LMs and weights. The LMs with better PPL and BLEU have larger weights.

The preceding results indicate that NNGC can scale up well on large corpora.

4.4.2. Conversion of Pruned LMs. Although NNGC using natural N-grams worked well on large corpora, it is time consuming to convert a BNLM trained from an even larger corpus (such as one containing around 5B words). Since the BLMP method can prune the useless N-grams from large LMs, without sacrificing their SMT performance significantly, instead of converting the huge BNLM, we can first make the pruning before converting it.

The smallest pruned LMs (KN/GT5B-Entropy-4 and KN/GT5B-Connect-4) that were nearly 1/20 the size of the original LM (KN/GT5B), and KN/GT5B-Connect-3 that were nearly 1/10 the size of the original LM (KN/GT5B) in Tables VII and VIII, were selected for conducted converting experiments. We took them as the input BNLM instead of the original one. The same setting for natural N-grams converting experiments were set here with the pruned LMs as the input. We also conducted significance tests to show whether the BLEU of CONV was significantly better than the corresponding BNLM, and the results are shown in Table XI.

From the results of Table XI, we can observe the following:

(1) The converted pruned CONVs outperformed the pruned BNLM significantly. This indicated that NNGC works well together with BLMP to convert a large LM efficiently.
(2) As the size of CONV increased, both PPL and BLEU were further improved. These results strengthened that the performance of converted LM becomes better, as the sizes of input BNLM increased.
(3) The BLEU of CONV-GT* were significantly better than the corresponding GT*. This indicated that NNGC also worked well together with GT smoothing.

In addition, the proposed BLMP could also be applied to the Arsoy method. That is, all the words in the short-list were added to the tails of i-grams in the corpus in order to produce $(i+1)$-grams called *artificial N-grams*. BLMP was applied to these artificial N-grams to select connecting phrases. These connecting phrases (N-grams) were used as input of CSLM and their probabilities were obtained as output. These generated

Table XI. Converting Pruned BNLM Using CSLM

LMs	Size	*N*-grams	PPL	BLEU
KN5B	163.1G	3945.3M	73.4	34.32
GT5B	162.8G	3945.3M	83.5	**34.58**
KN5B-Connect-3	19.3G	555.7M	86.9	33.79
CONV-KN5B-Connect-3	19.3G	555.7M	83.6	34.24^{++}
GT5B-Connect-3	19.2G	555.7M	87.5	33.58
CONV-GT5B-Connect-3	19.2G	555.7M	82.8	$\mathbf{34.30^{++}}$
KN5B-Entropy-4	8.2G	201.9M	93.8	33.16
CONV-KN5B-Entropy-4	8.2G	201.9M	89.5	33.57^{++}
GT5B-Entropy-4	8.5G	229.7M	82.8	34.07
CONV-GT5B-Entropy-4	8.5G	229.7M	79.8	34.19^{++}
KN5B-Connect-4	8.4G	208.0M	94.2	33.38
CONV-KN5B-Connect-4	8.4G	208.0M	90.1	33.83^{++}
GT5B-Connect-4	8.3G	208.0M	92.1	33.97
CONV-GT5B-Connect-4	8.3G	208.0M	88.5	$\mathbf{34.32^{++}}$
CONV-KN746M	7.5G	183.7M	85.9	33.59
CONV-KN42M	3.0G	73.9M	106.1	33.07

Note: The LMs in this table are the same as the LMs with the same names in Tables VII and VIII. ("++/−−": significantly better/worse at $\alpha = 0.01$; "+/−" : $\alpha = 0.05$; no mark: not significantly better/worse at $\alpha = 0.05$). The BLEU in bold indicate it is the highest one among the LMs with similar size.

Table XII. BLMP with Arsoy Method

LMs	Size	*N*-grams	PPL	BLEU
KN5B-Connect-3	19.3G	555.7M	86.9	33.79
Arsoy-Connect-3	19.6G	578.5M	100.3	32.98
CONV-KN5B-Connect-3	19.3G	555.7M	**83.6**	**34.24**
KN5B-Connect-4	8.4G	208.0M	94.2	33.38
Arsoy-Connect-4	8.0G	198.6M	101.2	33.05
CONV-KN5B-Connect-4	8.4G	208.0M	**90.1**	**33.83**
KN42M	3.0G	73.9M	108.8	32.35
Arsoy42M (pruned)	3.1G	71.8M	104.7	32.38
Arsoy42M (interpolated)	5.4G	140.1M	**103.5**	**32.91**
CONV-KN42M	3.0G	73.9M	106.1	33.07

N-grams were interpolated with the original BNLM to form the larger converted LM consequently.

Table XII showed that we used BLMP to prune both Arsoy's converted LMs and NNGC converted LMs.

(1) Arsoy's converting and the proposed BLMP can work well together (Arsoy-Connect-3/4) to enhance LM and SMT performance in comparison with KN42M.
(2) The NNGC with BLMP (CONV-KN5B-Connect-3/4) outperformed Arsoy's converting with BLMP (Arsoy-Connect-3/4) significantly. This result indicated natural *N*-grams were more helpful in comparison with artificial ones.

4.4.3. Experiments on NIST Dataset. We also evaluated the performance of our proposed methods on the NIST Chinese-to-English dataset. OpenMT06 in the NIST open machine translation 2006 Evaluation[24] was used. The parallel corpus followed the setting of Wang et al. [2014], and it mainly consists of news and blog texts. The training set

[24]http://www.itl.nist.gov/iad/mig/tests/mt/2006/.

Table XIII. Experiments on NIST Dataset

LMs	Size	N-grams	PPL	BLEU
NIST-KN12M	1.1G	23.9M	145.8	31.88
NIST-Arsoy12M	1.1G	23.9M	142.2	32.23
CONV-NIST-KN12M	1.1G	23.9M	**141.0**	32.45
NIST-KN206M-connect-1	2.7G	67.4M	176.8	31.96
CONV-NIST-KN206M-connect-1	2.7G	67.4M	174.2	**32.65**
NIST-KN206M-connect-2	4.6G	119.4M	179.3	31.77
CONV-NIST-KN206M-connect-2	4.6G	119.4M	177.8	32.37
NIST-KN206M	12.8G	260.0M	177.9	31.65

consists of 430K sentences and 12M words. The English corpus from the United Nations dataset[25] was used as the large monolingual corpus,[26] which included around 7M out-of-domain sentences and 206M words as the parallel corpus. The datasets of NIST Eval 2002 to 2005 was used as development data for MERT tuning. The NIST Eval 2006 was used as evaluation data. The same settings of Section 4.1 for SMT were followed.

We compared the performances of small/large BNLM (NIST-KN12/206M), the Arsoy method (NIST-Arsoy12M), the proposed NNGC method (CONV-NIST-KN12M), the BLMP method (NIST-KN206M-connect-1/2), and their converted LMs (CONV-NIST-KN206M-connect-1/2).

From the results of Table XIII, we make the following observations:

(1) Because the large extra monolingual corpus and parallel corpus were in different domains, the BLEU of BNLM constructed from the large corpus (NIST-KN206M) is slightly worse than NIST-KN12M. The pruned LM (NIST-KN206M-connect-1) was slightly better than NIST-KN206M and NIST-KN12M. This indicated that the bilingual LM pruning method can also be applied to LM adaptation.
(2) The proposed NNGC method worked well and performed slightly better than the Arsoy method on the NIST dataset. NNGC and BLMP also worked well together (CONV-NIST-KN206M-connect-1/2) on the NIST dataset.

5. CONCLUSION AND FUTURE WORK

In this article, we have proposed a NNGC CSLM converting method for better probability estimation, and a BLMP pruning approach for enhancing LMs for SMT decoding and improving the efficiency of CSLM converting. Our methods have the following two attractive features: (1) the NNGC and BLMP can select the N-grams with linguistic sense, which are potentially to be used in decoding, and (2) the proposed pruning and converting methods have lower computational complexity in comparison with the existing methods.

A series of experiments on various datasets have been conducted to evaluate the performance and efficiency of our proposed methods. The experimental results show that both the converting and pruning methods outperform the existing approaches for SMT. We have also shown that the NNGC converting method and the BLMP pruning method can work well together to convert a huge LM trained from 5B words efficiently. The experiments on NIST data also suggest that BLMP can be applied to LM adaptation.

In this article, the proposed NNGC converting method can be regarded as an optimization method of the probabilities of the items in an N-gram LM that requires a small computational cost, using an NNLM that has a high generalization ability and

[25]LDC2013T06 Info Local CD/DVDL310 "1993–2007 United Nations Parallel Text" (three DVDs).
[26]The English side of the small bilingual corpus was added to the large monolingual corpus.

requires a large computational cost. We would also like to use the distributed semantic representation in this optimization method as our future work.

ACKNOWLEDGMENTS

We appreciate the helpful discussion with Dr. Andrew Finch, Dr. Xiaolin Wang, Zhisong Zhang, and all of the anonymous reviewers and editors for many invaluable comments and suggestions to improve our article.

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 *Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP'13)*. IEEE, 8242–8246.

Ebru Arsoy, Stanley F. Chen, Bhuvana Ramabhadran, and Abhinav Sethy. 2014. Converting neural network language models into back-off language models for efficient decoding in automatic speech recognition. *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 22, 1 (2014), 184–192. DOI:http://dx.doi.org/10.1109/TASLP.2013.2286919

Michael Auli, Michel Galley, Chris Quirk, and Geoffrey Zweig. 2013. Joint language and translation modeling with recurrent neural networks. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Seattle, Washington, 1044–1054.

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

Thorsten Brants, Ashok C. Popat, Peng Xu, Franz J. Och, and Jeffrey Dean. 2007. Large language models in machine translation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*. Association for Computational Linguistics. 858–867.

Ciprian Chelba, Thorsten Brants, Will Neveitt, and Peng Xu. 2010. Study on interaction between entropy pruning and kneser-ney smoothing. In *Proceedings of The Annual Conference of the International Speech Communication Association*. 2242–2245.

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)*. 310–318. DOI:http://dx.doi.org/10.3115/981863.981904

Stanley F. Chen and Joshua Goodman. 1999. An empirical study of smoothing techniques for language modeling. *Computer Speech & Language* 13, 4 (1999), 359–393. DOI:http://dx.doi.org/10.1006/csla.1999.0128

Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 1724–1734.

Anoop Deoras, Tomas Mikolov, Stefan Kombrink, Martin Karafiát, and Sanjeev Khudanpur. 2011. Variational approximation of long-span language models for LVCSR. In *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP'11)*. IEEE, 5532–5535. DOI:http://dx.doi.org/10.1109/ICASSP.2011.5947612

Jacob Devlin, Rabih Zbib, Zhongqiang Huang, Thomas Lamar, Richard Schwartz, and John Makhoul. 2014. Fast and robust neural network joint models for statistical machine translation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 1370–1380.

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

Jianfeng Gao, Xiaodong He, Wen-tau Yih, and Li Deng. 2014. Learning continuous phrase representations for translation modeling. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 699–709.

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*. 559–578.

Michael Gutmann and Aapo Hyvärinen. 2010. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of the International Conference on Artificial Intelligence and Statistics*. 297–304.

Kenneth Heafield, Ivan Pouzyrevsky, Jonathan H. Clark, and Philipp Koehn. 2013. Scalable modified Kneser-Ney language model estimation. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. 690–696.

Zhongqiang Huang, Jacob Devlin, and Spyros Matsoukas. 2013. BBN's systems for the Chinese-English sub-task of the NTCIR-10 PatentMT evaluation. In *Proceedings of NII Testbeds and Community for Information Access Research (NTCIR'10)*.

Zhongye Jia and Hai Zhao. 2014. A joint graph model for Pinyin-to-Chinese conversion with typo correction. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*. 1512–1523.

Nal Kalchbrenner and Phil Blunsom. 2013. Recurrent continuous translation models. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*. 1700–1709.

Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, Dekang Lin and Dekai Wu (Eds.). 388–395.

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)*. 48–54. DOI:http://dx.doi.org/10.3115/1073445.1073462

Stanislas Lauly, Hugo Larochelle, Mitesh Khapra, Balaraman Ravindran, Vikas C. Raykar, and Amrita Saha. 2014. An autoencoder approach to learning bilingual word representations. In *Advances in Neural Information Processing Systems*. 1853–1861.

Hai-Son Le, I. Oparin, A. Allauzen, J. Gauvain, and F. Yvon. 2011. Structured output layer neural network language model. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP'11)*. 5524–5527. DOI:http://dx.doi.org/10.1109/ICASSP.2011.5947610

Peng Li, Yang Liu, Maosong Sun, Tatsuya Izuha, and Dakun Zhang. 2014. A neural reordering model for phrase-based translation. In *Proceedings of, the 25th International Conference on Computational Linguistics: Technical Papers (COLING 2014)*, 1897–1907. http://www.aclweb.org/anthology/C14-1179.

lemao Liu, Taro Watanabe, Eiichiro Sumita, and Tiejun Zhao. 2013. Additive neural networks for statistical machine translation. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 791–801.

Shujie Liu, Nan Yang, Mu Li, and Ming Zhou. 2014. A recursive recurrent neural network for statistical machine translation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 1491–1500.

Xuezhe Ma and Hai Zhao. 2012. Fourth-order dependency parsing. In *Proceedings of the 24th International Conference on Computational Linguistics*. 785–796.

Tomas Mikolov, Anoop Deoras, Daniel Povey, Lukas Burget, and Jan Cernock. 2011. Strategies for training large scale neural network language models. In *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP'11)*. 196–201.

Tomas Mikolov, Martin Karafiát, Lukas Burget, Jan Cernockỳ, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *Proceedings of the Annual Conference of the International Speech Communication Association*. 1045–1048.

Andriy Mnih and Geoffrey E. Hinton. 2008. A scalable hierarchical distributed language model. In *Advances in Neural Information Processing Systems 21*, D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou (Eds.). 1081–1088.

Vinod Nair and Geoffrey E. Hinton. 2010. Rectified linear units improve restricted Boltzmann machines. In *Proceedings of the 27th International Conference on Machine Learning (ICML'10)*. 807–814.

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'12)*. 311–318.

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*. 160–167. DOI:http://dx.doi.org/10.3115/1075096.1075117

Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics* 29, 1 (March 2003), 19–51. DOI:http://dx.doi.org/10.1162/089120103321337421

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)*. 311–318. DOI:http://dx.doi.org/10.3115/1073083.1073135

Xiaochang Peng and Daniel Gildea. 2014. Type-based MCMC for sampling tree fragments from forests. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP'14)*. 1735–1745.

Brian Roark, Cyril Allauzen, and Michael Riley. 2013. Smoothed marginal distribution constraints for language modeling. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (ACL)*. 43–52.

Holger Schwenk. 2007. Continuous space language models. *Computer Speech and Language* 21, 3 (2007), 492–518. DOI:http://dx.doi.org/10.1016/j.csl.2006.09.003

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

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)*. 723–730.

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)*. 11–19.

Vesa Siivola, Teemu Hirsimki, and Sami Virpioja. 2007. On growing and pruning Kneser-Ney smoothed N-gram models. *IEEE Transactions on Audio, Speech, and Language Processing* 15, 5 (2007), 1617–1624. DOI:http://dx.doi.org/10.1109/TASL.2007.896666

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)*. 778–788.

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)*. 39–48.

Andreas Stolcke. 1998. Entropy-based pruning of backoff language models. In *Proceedings of the DARPA Broadcast News Transcription and Understanding Workshop*. 270–274.

Andreas Stolcke. 2002. SRILM—An extensible language modeling toolkit. In *Proceedings of the International Conference on Spoken Language Processing*, 257–286.

Andreas Stolcke, Jing Zheng, Wen Wang, and Victor Abrash. 2011. SRILM at sixteen: Update and outlook. In *Proceedings of the IEEE Automatic Speech Recognition and Understanding Workshop*.

Martin Sundermeyer, Tamer Alkhouli, Joern Wuebker, and Hermann Ney. 2014. Translation modeling with bidirectional recurrent neural networks. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 14–25.

Ashish Vaswani, Yinggong Zhao, Victoria Fossum, and David Chiang. 2013. Decoding with large-scale neural language models improves translation. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*. 1387–1392.

Rui Wang, Masao Utiyama, Isao Goto, Eiichro Sumita, Hai Zhao, and Bao-Liang Lu. 2013. Converting continuous-space language models into N-gram language models for statistical machine translation. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*. 845–850.

Rui Wang, Hai Zhao, and Bao Liang Lu. 2015a. English to Chinese translation: How Chinese character matters?. In *Proceedings of the 29th Pacific Asia Conference on Language, Information and Computation*, 274–284.

Rui Wang, Hai Zhao, Bao Liang Lu, Masao Utiyama, and Eiichiro Sumita. 2014. Neural network based bilingual language model growing for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*. 189–195.

Rui Wang, Hai Zhao, Bao-Liang Lu, M. Utiyama, and E. Sumita. 2015b. Bilingual continuous-space language model growing for statistical machine translation. *Audio, Speech, and Language Processing, IEEE/ACM Transactions on* 23, 7 (July 2015), 1209–1220. DOI:http://dx.doi.org/10.1109/TASLP.2015.2425220

Xiaolin Wang, Masao Utiyama, Andrew Finch, and Eiichiro Sumita. 2014. Empirical study of unsupervised Chinese word segmentation methods for SMT on large-scale corpora. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 752–758. http://www.aclweb.org/anthology/P14-2122.

Qiongkai Xu and Hai Zhao. 2012. Using deep linguistic features for finding deceptive opinion spam. In *Proceedings of the 24th International Conference on Computational Linguistics*. 1341–1350.

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 (2009), 79–88.

Jingyi Zhang, Masao Utiyama, Eiichiro Sumita, and Hai Zhao. 2014. Learning hierarchical translation spans. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*. 183–188.

Jingyi Zhang and Hai Zhao. 2013. Improving function word alignment with frequency and syntactic information. In *Proceedings of the 23rd International Joint Conference on Artificial Intelligence*. 2211–2217.

Hai Zhao. 2009. Character-level dependencies in Chinese: Usefulness and learning. In *Proceedings of the 12th Conference of the European Chapter of the ACL (EACL'09)*. 879–887.

Hai Zhao, Wenliang Chen, and Chunyu Kit. 2009a. Semantic dependency parsing of NomBank and PropBank: An efficient integrated approach via a large-scale feature selection. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*. 30–39.

Hai Zhao, Yan Song, Chunyu Kit, and Guodong Zhou. 2009b. Cross language dependency parsing using a bilingual lexicon. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*. Suntec, Singapore, 55–63.

Hai Zhao, Masao Utiyama, Eiichiro Sumita, and Bao-Liang Lu. 2013. An empirical study on word segmentation for Chinese machine translation. In *Computational Linguistics and Intelligent Text Processing*. Lecture Notes in Computer Science, Vol. 7817. 248–263.

Will Y. Zou, Richard Socher, Daniel Cer, and Christopher D. Manning. 2013. Bilingual word embeddings for phrase-based machine translation. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, 1393–1398.

Received May 2014; revised July 2015; accepted November 2015