Towards Machine Translation in Semantic Vector Space

JIAJUN ZHANG, Chinese Academy of Sciences
SHUJIE LIU, MU LI, and MING ZHOU, Microsoft Research Asia
CHENGQING ZONG, Chinese Academy of Sciences

Measuring the quality of the translation rules and their composition is an essential issue in the conventional statistical machine translation (SMT) framework. To express the translation quality, the previous lexical and phrasal probabilities are calculated only according to the co-occurrence statistics in the bilingual corpus and may be not reliable due to the data sparseness problem. To address this issue, we propose measuring the quality of the translation rules and their composition in the semantic vector embedding space (VES). We present a recursive neural network (RNN)-based translation framework, which includes two submodels. One is the bilingually-constrained recursive auto-encoder, which is proposed to convert the lexical translation rules into compact real-valued vectors in the semantic VES. The other is a type-dependent recursive neural network, which is proposed to perform the decoding process by minimizing the semantic gap (meaning distance) between the source language string and its translation candidates at each state in a bottom-up structure. The RNN-based translation model is trained using a max-margin objective function that maximizes the margin between the reference translation and the n-best translations in forced decoding. In the experiments, we first show that the proposed vector representations for the translation rules are very reliable for application in translation modeling. We further show that the proposed type-dependent, RNN-based model can significantly improve the translation quality in the large-scale, end-to-end Chinese-to-English translation evaluation.

Categories and Subject Descriptors: I.2.7 [**Natural Language Processing**]: Machine Translation

General Terms: Algorithms, Languages, Experimentation

Additional Key Words and Phrases: statistical machine translation, recursive neural network, semantic meaning distance, vector embedding space, max-margin training

ACM Reference Format:
Zhang, J., Liu, S., Li, M., Zhou, M., and Zong, C. 2015. Towards machine translation in semantic vector space. *ACM Trans. Asian Low-Resour. Lang. Inf. Process.* 14, 2, Article 9 (March 2015), 26 pages.
DOI: http://dx.doi.org/10.1145/2699927

1. INTRODUCTION

Currently, typical statistical machine translation (SMT) models, such as phrase-based models [Koehn et al. 2007], formal syntax-based models [Chiang 2007; Xiong et al. 2006], and linguistically syntax-based models [Galley et al. 2006; Huang et al. 2006; Liu et al. 2006; Zhang et al. 2008, 2011, 2013], perform the decoding process and generate the translation result by compositing a set of the translation rules. As a result,

This research work has been supported by the Natural Science Foundation of China under Grant No. 61303181, the International Science & Technology Cooperation Program of China under Grant No. 2014DFA11350, and the High New Technology Research and Development Program of Xinjiang Uyghur Autonomous Region under Grant No. 201312103.
Authors' addresses: J. Zhang (corresponding author) and C. Zong, National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, Intelligence Building, No. 95, Zhongguancun East Road, Haidian District, Beijing, 100190, China; emails: {jjzhang, cqzong}@nlpr.ia.ac.cn; S. Liu, M. Li, and M. Zhou, Microsoft Research Asia, No. 5 Danling Road, Haidian District, Beijing, 100190, China; emails: {shujliu, muli, mingzhou}@microsoft.com.

DOI: http://dx.doi.org/10.1145/2699927

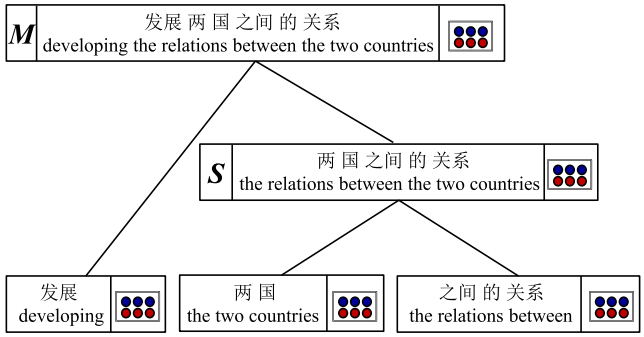

Fig. 1. An example of the RNN-based translation model with (string and semantic vector) representations for source and target side in each node. The leaf nodes are translation rules (phrase pairs), and the nonterminals ***S*** and ***M*** determines which network or composition scheme will be applied to combine the children to yield translations of the longer strings. ***S*** means the target-side phrases of the two children will be swapped after combination, and ***M*** denotes monotone combination. At each nonterminal node, the semantic gap between the source- and target-side vector representations are utilized to guide the process of choosing the best translation candidates.

the quality of the translation rules and the rule composition become the key factors to obtaining good translation results. The problem becomes how to express and measure the quality of the translation rules and how to design rule composition schemes.

Conventionally, the quality of the translation rules is expressed by the translation probabilities (e.g., the translation probabilities at the phrase level and at the word level), which are all computed based on the co-occurrence statistics of the rule's source and target sides in the bilingual corpus. However, the translation probability relying on the co-occurrence statistics is much biased to the existing bilingual corpus, and the data sparseness problem would make the probability estimation unreliable. For example, if a translation rule appears only once in the bilingual corpus, this rule will associate with a very high translation probability, but it is more likely to be a noisy translation rule. Therefore, this kind of translation quality estimation is not sufficient to show whether the source and target sides in a translation rule are of the same meaning.

Previously, the rule composition scheme has been designed to maximize the translation probability of the dynamically-merged translation hypothesis. As we have mentioned, the translation probability does not well reflect the semantic meaning between the source input and the target translation, so the SMT models using the conventional rule composition schemes cannot ensure that the generated translations convey the same semantic meaning of the source-side inputs.

In this article, we propose a recursive neural network (RNN)-based translation framework to perform translation in the continuous semantic space. To address the data sparseness problem of translation rule probability estimation and aim at measuring the semantic meaning between the rule's source and target sides, the RNN-based model learns to represent each lexical translation rule with continuous semantic vectors. The quality of the translation rule will be measured by the distance between the rule's source and target sides in the continuous semantic vector space. Aiming at retaining the semantic meaning during the translation process, the RNN-based model learns how to perform decoding or rule composition using the merging type (*swap* or *monotone*)-dependent recursive neural networks that attempt to find the best translation candidate having minimal semantic gap with the source-side input string.

Figure 1 shows an example for our RNN-based translation model. Each node in Figure 1 is associated with a pair of strings and corresponding semantic vectors.

Since the ideal translation model is the one whose target translation retains the semantic meaning of the source-side input, our designed model objective is to search the best hidden derivation tree with the minimal sum of the semantic gaps in each node.

The nodes in Figure 1 are divided into two groups: leaf nodes and nonterminal nodes. The leaf nodes denote the basic phrasal translation rules, and the nonterminal nodes denote the rule composition schemes. Following Zhang et al. [2014b], our RNN-based model designs two submodels to handle the leaf nodes and nonterminal nodes, respectively. Since the basic phrasal translation rules are directly induced from the bitexts, both source and target sides of the leaf nodes are grammatical, and the semantic gap between them should be a fixed value. We apply our proposed bilingually-constrained recursive auto-encoders (BRAE) Zhang et al. [2014a, 2014b] to semantically embed each source and target grammatical phrases with compact real-valued vectors and find their fixed semantic gaps. BRAE is learned by minimizing the semantic distance between the high-quality translation equivalents (source phrases and their correct translations) and maximizing the semantic distance of non-translation pairs simultaneously. With the learned BRAE model, each translation rule is represented by two compact semantic vectors (one for the source-side string and the other for the target-side string), and each leaf node is denoted with a tuple (bilingual strings and two vectors). The semantic gap between the bilingual strings will be the vector distance in the continuous semantic space. Previously, Zhang et al. [2014a] only focused on learning phrase embeddings and measuring the quality of the phrase pair entries in the phrase table in the conventional phrase-based translation [Koehn et al. 2007]. And the translation model was not the focus of our previous work. In this work, the proposed method [Zhang et al. 2014a] is applied to fulfil the leaf node representation task of the newly proposed RNN-based translation model. For the BRAE model, we further conduct a thorough comparison of our method, initially described in Zhang et al. [2014a, 2014b]. Additional experiments include comparison between our BRAE model and the unsupervised/semi-supervised models and the averaged embedding method.

Given the leaf nodes in tuples, another submodel is proposed to deal with the rule composition schemes. This submodel learns how to composite any two children in the derivation tree recursively. Following the bracketing transduction grammars [Wu 1997], we adopt two types of composition operators: monotone and swap, as shown in Figure 1. Accordingly, two type-dependent networks are designed for monotone and swap composition, respectively. The networks need to learn two kinds of functions: vector composition functions and vector space transformation functions.

The vector composition functions take the vectors of the left and right children as input and output a vector to represent the semantic meaning of the parent. For the source side, only one vector composition function is needed. Since the target-side strings are combined in two different ways according to monotone or swap compositions, two vector composition functions are involved for the target side: one for the monotone operator and the other for the swap operator. There are two vector space transformation functions which transform the source-side embedding space to the target-side embedding space and vice versa. The two kinds of functions in the networks are optimized using an objective of max-margin loss which prefers the gold derivation trees generated by successful forced decoding to the k-best derivation trees yielded by the conventional SMT models. For the RNN-based translation framework, which is initially described in Zhang et al. [2014b], we further conduct thorough comparison experiments, including comparing two kinds of neural network optimization algorithms: stochastic gradient descent (SGD) algorithm and subgradient (AdaGrad) algorithm [Duchi et al. 2011; Socher et al. 2013]. Furthermore, besides the important factor of the optimization algorithm, which much influences the translation quality, we conduct

a comprehensive investigation on other factors of the recursive neural networks which may influence the quality of the translation.

With the learned RNN-based translation model, we run large-scale experiments on Chinese-to-English translation. Extending the experiments in Zhang et al. [2014b], we first show in the experiments that the proposed vector representations for the translation rules are quite reliable for serving as the inputs of the RNN-based translation model. We further show that the proposed type-dependent RNN-based model can significantly outperform the state-of-the-art baseline system.

The remainder of this article is organized as follows: we present the related work in the next section and then introduce our RNN-based translation model in detail. The experimental results and analysis will be elaborated afterwards. Finally, we conclude and give the future work.

2. RELATED WORK

In recent years, many researchers attempt to model the translation process with continuous vector representations for words, phrases, and even sentences. Almost all of them address only some aspects of the statistical machine translation, such as bilingual word/phrase similarity or probability estimation [Gao et al. 2014; Schwenk 2012; Zou et al. 2013], language modeling [Mikolov 2012; Schwenk 2010; Vaswani et al. 2013], more context usage for target language word prediction and the sparsity problem in translation [Auli et al. 2013; Devlin et al. 2014; Kalchbrenner and Blunsom 2013; Liu et al. 2013], and the phrase reordering problem [Li et al. 2013].

For bilingual word/phrase similarity or probability estimation, Zou et al. [2013] proposes a bilingually-constrained word embedding method that can measure bilingual word similarity using distance in the embedding space. They calculate the lexical weights of the phrase pairs with bilingual word similarities and show some gains are achieved in BLEU score. Schwenk [2012] presents a neural-network-based method to estimate the forward phrase translation probability: all the words of the source-side phrase are projected onto a continuous vector space, and the neural network predicts the joint probability of all the words of the target-side phrase. The method demonstrates that the probability of unseen phrase pairs can be estimated and the translation quality can also be improved when using the probability to rerank the translation hypotheses or integrating the probability in the decoding stage. Gao et al. [2014] use the bag-of-words strategy to map bilingual phrases onto a common continuous space and update the phrase embeddings with respect to the BLEU score so that the final phrase embeddings are BLEU sensitive.

Rather than using the conventional N-gram-based language models in statistical machine translation, the feed-forward or recurrent neural networks are applied to design the continuous language models [Mikolov 2012; Schwenk 2010; Vaswani et al. 2013]. The continuous language model can make full use of the long history context of the target side and can alleviate the data spareness problem in language model estimation. Schwenk [2010] and Vaswani et al. [2013] investigate the continuous language models based on feed-forward neural networks. Schwenk [2010] utilizes the continuous language model for translation hypotheses reranking, while Vaswani et al. [2013] integrate the continuous language model into the decoder beside hypotheses reranking. In contrast, Mikolov [2012] uses the continuous language model based on the recurrent neural network to rerank the translation hypotheses. All of these models report improvements in translation quality.

Besides the target-side history words, more source-side context can lead to better prediction of target word translation. Some works [Auli et al. 2013; Devlin et al. 2014; Kalchbrenner and Blunsom 2013; Liu et al. 2013] map both the source-side context and the target-side history into a real-valued vector and utilize the continuous vector

to better predict target word generation. The most notable is the work of Devlin et al. [2014], in which a feed-forward neural network with two hidden layers is designed. The input is formed by concatenating the vectors of an 11-window source-side context and a 3-window target-side history context. The output is the conditional probability of the current target-side word. The remarkable improvement is reported in both Arabic-to-English and Chinese-to-English translation tasks.

Phrase reordering is an important problem in statistical machine translation. Instead of using lexical words as concrete features, Li et al. [2013] has proposed a recursive auto-encoder method to convert each phrase into a continuous real-valued vector which can encode the reordering tendency of the phrases (e.g., swap or monotone).

Different from the previous works, we aim at learning the semantic vector representation for each phrasal translation rule (a pair of source and target phrase), and the translation model is optimized by directly minimizing the semantic gap between a source string and its translation candidate. For semantic phrase representation in vector embedding space, Zou et al. [2013] has shown that semantic word embedding can be obtained with bilingual constraints, and Socher et al. [2010] has proven that recursive auto-encoders provide a reasonable way to embed phrases, so we combine the two ideas together and propose the bilingually-constrained recursive auto-encoders (BRAE) to learn the semantic phrase embeddings.

As the objective of machine translation is to find the target hypothesis sharing the same semantic meaning of the source-side input, we can approach this purpose by optimizing the objective function in the vector embedding space. Socher et al. [2013] has shown that the recursive neural network (RNN) can well predict the parse tree structure, and we know that machine translation can also be considered a tree structure prediction problem, so we extend the application of RNN to machine translation for bottom-up derivation tree structure prediction with a different objective function. Compared to the recurrent neural network, which is good at handling sequence prediction, the recursive neural networks can do well in tree structure prediction (which rules should be used and how these rules should be combined). Thus, we adopt recursive neural networks in our method.

3. RNN-BASED TRANSLATION FRAMEWORK

This section introduces in detail our proposed RNN-based translation framework. First, we introduce the baseline translation system: a formal syntax-based translation model which is based on the bracketing transduction grammars (BTG). Then, we present how to apply the bilingually-constrained recursive auto-encoders to semantically embed each phrasal translation rules with compact real-valued vectors. Finally, we detail the RNN-based translation model, including objective function design, model scoring, and optimization algorithm.

3.1. BTG-Based Translation Model

The BTG-based translation [Wu 1997; Xiong et al. 2006] can be viewed as a monolingual parsing process in which only lexical rules $A \rightarrow (x, y)$ and two binary merging rules $A \rightarrow [A^l, A^r]$ and $A \rightarrow \langle A^l, A^r \rangle$ are allowed.

The lexical translation rule $A \rightarrow (x, y)$ converts a source-language phrase into a target-language phrase, and forms a *block*.[1] It plays the same role as the phrasal translation pairs in the conventional phrase-based translation models [Koehn et al. 2007]. In practice, the BTG-based model and the conventional phrase-based model share the same phrase table. The biggest difference between these two models lies in

[1]A block consists of a source-language string and its target translation candidate.

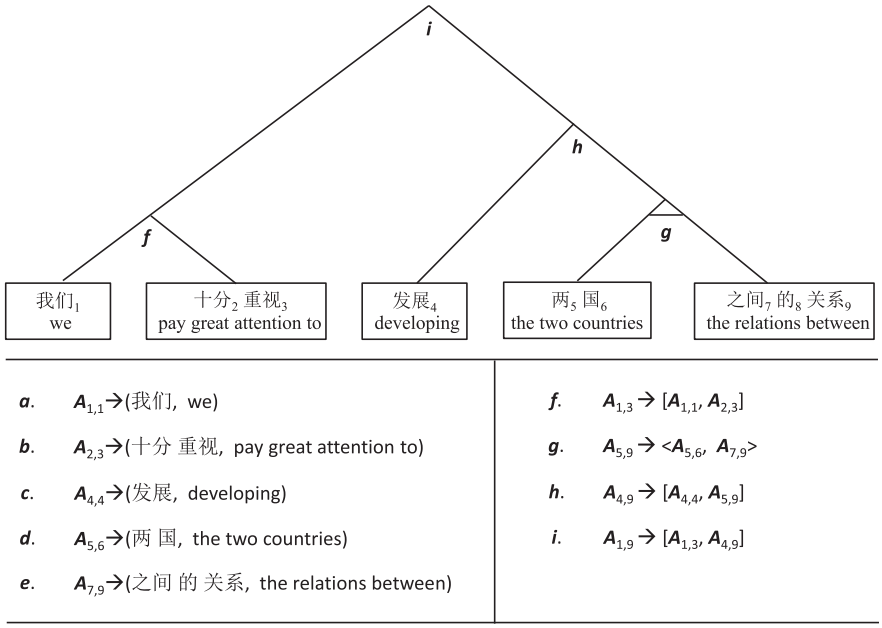

Fig. 2. A BTG derivation for a nine-word Chinese sentence and its translation candidate. The upper part shows the derivation tree, the bottom-left part presents the lexical rules, and the bottom-right part gives the monotone/swap merging rules.

that the conventional phrase-based model generates the target translation from left to right using the beam search algorithm, while the BTG-based model yields the final translation in a bottom-up manner using bracketing transduction grammars with the CYK algorithm. Loosely speaking, the BTG-based translation process is more like tree structure prediction.

The monotone merging rule $A \rightarrow [A^l, A^r]$ combines the two consecutive[2] blocks into a bigger block by concatenating the two partial target translation candidates in order, while the swap rule $A \rightarrow \langle A^l, A^r \rangle$ yields the bigger block by swapping the two partial target translation candidates. Which type of the merging rules should be adopted is considered as a binary classification problem, and maximum entropy model is usually applied [Xiong et al. 2006].

During decoding, the model searches for a BTG derivation consisting of a sequence of production rules, which demonstrate how a source sentence is parsed, and the corresponding target translation is generated simultaneously. Figure 2 illustrates a BTG derivation that translates a Chinese sentence into an English sentence. The Chinese sentence contains nine words. First, the lexical translation rules $\{a, b, c, d, e\}$ translate the source phrases into the target phrases and form blocks $\{A_{1,1}, A_{2,3}, A_{4,4}, A_{5,6}, A_{7,9}\}$. The monotone merging rule f combines the two neighboring blocks $A_{1,1}$ and $A_{2,3}$ into a bigger block $A_{1,3}$. The swap merging rule g combines the two consecutive blocks $A_{5,6}$ and $A_{7,9}$ and results in a bigger block $A_{5,9}$. Then, two monotone merging rules h and i are applied until the whole source sentence is covered, and the final target translation is *we pay great attention to developing the relations between the two countries*.

Typically, a log-linear model [Och and Ney 2002] is applied to find the optimal derivation which consists of a set of translation rules (lexical rules and merging rules).

[2]Two blocks are consecutive as long as the source-language phrases of the blocks are consecutive.

The optimal derivation yields the best translation, and the conditional probability is calculated in the log-linear formulation

$$Pr(e|f) = p_\lambda(e,f) = \frac{exp(\sum_i \lambda_i h_i(f,e))}{\sum_{e'} exp(\sum_i \lambda_i h_i(f,e'))}, \quad (1)$$

in which h_is are feature functions, such as the bidirectional phrasal translation probabilities, the bidirectional lexical weights, the language model, and the reordering model. λs are feature weights.

3.2. Semantic Vector Representations for Translation Rules

To perform decoding in the semantic vector space with recursive neural networks, the first task is to convert each lexical translation rule into a semantic vector representation. As we know that the lexical translation rule (e.g., $\{a,b,c,d,e\}$ in Figure 2) is a tuple consisting of a source-language phrase and a target-language phrase, we just need to represent the source and target phrases with semantic vector representations. We propose bilingually-constrained recursive auto-encoders (BRAE) to learn the semantic vector representation for each phrase. Our BRAE views any phrase as a meaningful composition of its internal words, and the key idea is to learn the word vector representation and the way of composition. We first present the word vector representations and then introduce the proposed BRAE model for learning the semantic composition.

3.2.1. Word Vector Representations. Recently, word vector representations are typically learned with deep neural networks (DNN), which convert a word into a dense, low-dimensional, real-valued vector [Bengio et al. 2003, 2006; Collobert and Wetson 2008; Collobert et al. 2011; Mikolov et al. 2013; Zou et al. 2013]. In the vector, each dimension represents a latent aspect of a word, capturing its semantic and syntactic properties [Bengio et al. 2006]. After learning with DNN, each word in the vocabulary V corresponds to a vector $x \in \mathbb{R}^n$, and all the vectors are stacked into a word embedding matrix $L \in \mathbb{R}^{n \times |V|}$.

Given a phrase which is an ordered list of m words, each word has an index i into the columns of the embedding matrix L. The index i is used to retrieve the word's vector representation using a simple multiplication with a binary vector e, which is zero in all positions except for the ith index:

$$x_i = Le_i \in \mathbb{R}^n. \quad (2)$$

Note that n is usually set empirically, such as $n = 25, 50$. Throughout this article, $n = 3$ is used in the examples for better illustration, as shown in Figure 1.

In the following section, we first present the naïve approach for the phrase vector representations using the standard recursive auto-encoders (RAE).

3.2.2. Unsupervised Phrase Vector Representations. Given a phrase $w_1 w_2 \cdots w_m$, it is first projected into a list of vectors $(x_1, x_2, \cdots, x_m)$ using Equation (2). The RAE learns the vector representation of the phrase by recursively combining two children vectors in a bottom-up manner [Socher et al. 2011]. Figure 3 illustrates an instance of an RAE applied to a binary tree, in which a standard auto-encoder (in box) is reused at each node. The standard auto-encoder aims at learning an abstract representation of its input. For two children $c_1 = x_1$ and $c_2 = x_2$, the standard auto-encoder computes the parent vector $y_1 = p$ as follows:

$$p = f(W^{(1)}[c_1; c_2] + b^{(1)}). \quad (3)$$

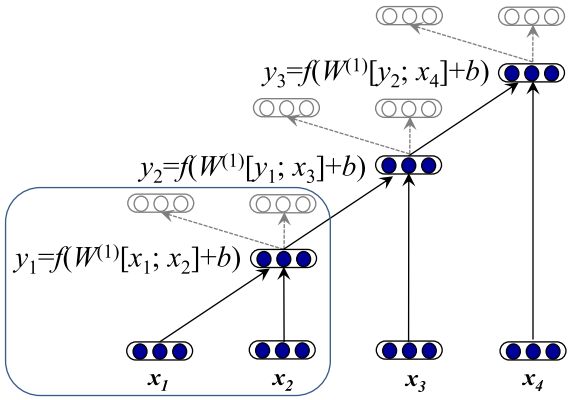

Fig. 3. A recursive auto-encoder for a four-word phrase. The empty nodes are the reconstructions of the input. The box shows a standard auto-encoder.

Where we multiply the parameter matrix $W^{(1)} \in \mathbb{R}^{n\times 2n}$ by the concatenation of two children $[c_1;c_2] \in \mathbb{R}^{2n\times 1}$. After adding a bias term $b^{(1)}$, we apply an element-wise activation function such as $f = tanh(\cdot)$, which is used in our experiments. In order to apply this auto-encoder to each pair of children, the representation of the parent p should have the same dimensionality as the children c_i's.

To assess how well the parent's vector represents its children, the standard auto-encoder reconstructs the children in a reconstruction layer:

$$[c_1';c_2'] = f^{(2)}(W^{(2)}p + b^{(2)}), \tag{4}$$

where c_1' and c_2' are reconstructed children, $W^{(2)}$ and $b^{(2)}$ are the parameter matrix and bias term for reconstruction, respectively, and $f^{(2)} = tanh(\cdot)$.

To obtain the optimal abstract representation of the inputs, the standard auto-encoder tries to minimize the reconstruction errors between the inputs and the reconstructed ones during training:

$$E_{rec}([c_1;c_2]) = \frac{1}{2}||[c_1;c_2] - [c_1';c_2']||^2. \tag{5}$$

Given $y_1 = p$, we can use Equation (3) again to compute y_2 by setting $[c_1;c_2] = [y_1;x_3]$. The same auto-encoder is reused until the vector of the whole phrase is generated.

For unsupervised phrase embedding, the only objective is to minimize the sum of the reconstruction errors at each node in the optimal binary tree:

$$RAE_\theta(x) = \underset{y\in A(x)}{\operatorname{argmin}} \sum_{node\in y} E_{rec}([c_1;c_2]_{node}), \tag{6}$$

where x is the list of vectors of a phrase, and $A(x)$ denotes all the possible binary trees that can be built from inputs x. A greedy algorithm [Socher et al. 2011] is used to generate the optimal binary tree y, and *node* denotes each internal node in the optimal binary tree y. The parameters $\theta = (W, b)$ are optimized over all the phrases in the training data.

3.2.3. Semi-supervised Phrase Representation. Through analysis, we find that the unsupervised method can only induce general representations of the multiword phrases. Recently, several researchers have extended the original unsupervised RAEs to a

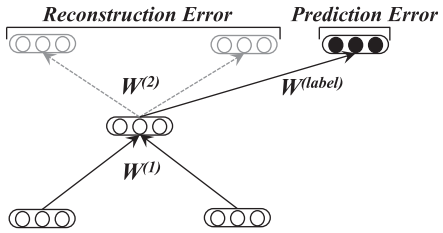

Fig. 4. An illustration of a semi-supervised RAE unit. **Black** nodes show the label distribution.

semi-supervised version so that the induced phrase representations can predict a target label, such as polarity in sentiment analysis [Socher et al. 2011], syntactic category in parsing [Socher et al. 2013], and phrase reordering pattern in SMT [Li et al. 2013].

In semi-supervised RAE for phrase representation, the objective function over a (phrase, label) pair (x, l) includes the reconstruction error and the prediction error, as illustrated in Figure 4.

$$E(x, l; \theta) = \alpha E_{rec}(x, l; \theta) + (1 - \alpha) E_{pred}(x, l; \theta), \tag{7}$$

where the hyper-parameter α is used to balance the reconstruction and prediction error. For label prediction, the cross-entropy error is usually used to calculate E_{pred}:

$$E_{pred}(x, l; \theta) = -\sum_{k=1}^{K} l_k log(f_k(x)), \tag{8}$$

in which K is the number of labels and $f(x) = softmax(W^{(label)}x + b)$.

By optimizing this objective, the phrases in the vector embedding space will be grouped according to the labels. For example, in syntactic parsing using semi-supervised RAEs, the phrases sharing the same syntactic tag are near each other in the embedding space and will be grouped into the same cluster.

3.2.4. The BRAE model. Inspired by the semi-supervised method, we know that semantic vector embedding for phrases would be possible if some gold semantic phrase vector representations were available for supervision. However, no gold semantic phrase vector representations exist in the real world. Fortunately, we know that the translation equivalents should share the same semantic meaning and thus should share the same semantic vector representation ideally. We can further make inference from this fact that if a model can learn the same vector representation for the source and target phrases in the translation equivalents, the learned vector representation must encode the semantics of the phrases, and the corresponding model is our desire.

We know from the preceding analysis that the source phrase and the target phrase in a translation equivalent can supervise each other to induce their semantic meanings. Accordingly, we propose bilingually-constrained recursive auto-encoders (Figure 5 shows the network structure) whose basic goal is to minimize the semantic distance between the phrases and their correct translations.

semantic error $E_{sem}(s, t; \theta)$

It denotes the semantic distance between the learned vector representations p_s and p_t.

Since word embeddings for two languages are learned separately and located in different vector space, we do not force phrase embeddings in two languages to be in the same semantic vector space. We suppose there is a transformation between the two

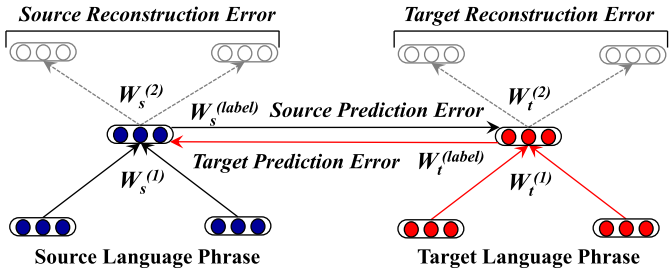

Fig. 5. An illustration of the bilingual-constrained recursive auto-encoders. The two phrases are translation equivalents induced with forced decoding with the baseline SMT.

semantic embedding spaces: one transformation function for each direction. Thus, the semantic distance is bidirectional: the distance between p_t and the transformation of p_s from the source-language embedding space to the target-language embedding space, and that between p_s and the transformation of p_t. As a result, the overall semantic error becomes

$$E_{sem}(s,t;\theta) = E_{sem}(s|t,\theta) + E_{sem}(t|s,\theta), \tag{9}$$

where $E_{sem}(s|t,\theta) = E_{sem}(p_t, f(W_s^{label} p_s + b_s^{label}))$ means the transformation of p_s is performed as follows: we first multiply a parameter matrix W_s^{label} by p_s, and after adding a bias term b_s^{label}, we apply an element-wise activation function $f = tanh(\cdot)$. Finally, we calculate their Euclidean distance:

$$E_{sem}(s|t,\theta) = \frac{1}{2}||p_t - f(W_s^{label} p_s + b_s^{label})||^2. \tag{10}$$

In the preceding equation, we calculate the semantic distance in the target language embedding space. Thus, t is the condition and does not need to do transformation. After transformation of the source phrase representation, $f(\cdot)$ is just another representation of the source phrase s in the target language embedding space.

Ideally, we want the learned BRAE model to ensure that the semantic error for the positive example (a source phrase s and its correct translation t) is much smaller than that for the negative example (the source phrase s and a false translation t'). However, the current model cannot guarantee this, since the semantic error $E_{sem}(s|t,\theta)$ only accounts for positive ones. We thus enhance the semantic error with both positive and negative examples, and the corresponding max-semantic-margin error becomes

$$E^*_{sem}(s|t,\theta) = max\{0, E_{sem}(s|t,\theta) - E_{sem}(s|t',\theta) + 1\}. \tag{11}$$

It tries to minimize the semantic distance between translation equivalents and maximize the semantic distance between non-translation pairs simultaneously. Using this error function, we need to construct a negative example for each positive example. Suppose we are given a positive example (s,t); the correct translation t can be converted into a false translation t' by replacing the words in t with randomly-chosen target language words. Then, a negative example (s,t') is available.

$E^*_{sem}(t|s,\theta)$ can be calculated in exactly the same way.

Similar to the semi-supervised RAEs, besides the semantic distance error for a translation equivalent of a phrase pair (s,t), the objective function includes the reconstruction errors as well:

$$E_{rec}(s,t;\theta) = E_{rec}(s;\theta) + E_{rec}(t;\theta), \tag{12}$$

in which $E_{rec}(s;\theta)$ and $E_{rec}(t;\theta)$ denote the reconstruction error for the source phrase s and target phrase t. They can be calculated with Equation (6).

Then, the joint error for the phrase pair (s,t) will be

$$E(s,t;\theta) = \alpha E_{rec}(s,t;\theta) + (1-\alpha)E_{sem}(s,t;\theta). \quad (13)$$

The hyper-parameter α balances the reconstruction error and the semantic error. The final BRAE objective over the phrase pairs training set (S,T) becomes

$$J_{BRAE} = \frac{1}{N}\sum_{(s,t)\in(S,T)} E(s,t;\theta) + \frac{\lambda}{2}||\theta||^2. \quad (14)$$

To learn the BRAE model, we need to optimize all the parameters associated with the model. The parameters θ can be divided into three sets.

— θ_L Word embedding matrix L for two languages.
— θ_{rec} Recursive auto-encoder parameter matrices $W^{(1)}$, $W^{(2)}$, and bias terms $b^{(1)}$, $b^{(2)}$ for two languages.
— θ_{sem} Transformation matrix W^{label} and bias term b^{label} for two directions in semantic distance computation.

From another point of view, the parameters θ can be divided into the source-side parameters θ_s and the target-side parameters θ_t. As seen in Figure 5, if the target phrase representation p_t is available, the optimization of the source-side parameters becomes a supervised learning problem. We can apply the stochastic gradient descent (SGD) algorithm to optimize each parameter. For parameter initialization, word vector representations θ_L are initialized with a DNN toolkit Word2Vec [Mikolov et al. 2013] using large-scale monolingual data and will be fine-tuned in our BRAE model to capture much more semantics. (Note that the following co-training algorithm performs the fine-tune job). Other parameters (θ_{rec} and θ_{sem}) are randomly initialized according to a normal distribution.

The optimization of the target-side parameters can be performed in the same way if the source phrase representation p_s is available. It seems a paradox that updating θ_s needs p_t, while updating θ_t needs p_s. To solve this problem, we propose a co-training style algorithm which includes three steps.

(1) Pre-training. Applying unsupervised phrase embedding with standard RAE to pre-train the source- and target-side phrase representations p_s and p_t, respectively.
(2) Fine-tuning. With the BRAE model, using target-side phrase representation p_t to update the source-side parameters θ_s and obtain the fine-tuned source-side phrase representation p'_s, meanwhile using p_s to update θ_t and get the fine-tuned p'_t, and then calculate the joint error over the training corpus.
(3) Termination Check. If the joint error reaches a local minima or the iterations reach the predefined number (25 is used in experiments), we terminate the training procedure; otherwise, we set $p_s = p'_s$, $p_t = p'_t$, and go to step 2.

After parameter training, the BRAE model can learn a semantic vector representation for each source and target phrase, respectively. Thus, each lexical translation rule $A \to (x,y)$ can be represented with two semantic compact vectors.

3.3. RNN-Based Translation Model

With the lexical translation rules represented as the semantic compact vectors, the RNN-based model aims to find, for a test source language sentence, the best derivation tree whose target translation has the minimal semantic distance with the source language input. For ease of exposition, we first describe how to score an existing derivation tree in which each node consists of the concrete string and the continuous real-valued vector representations.

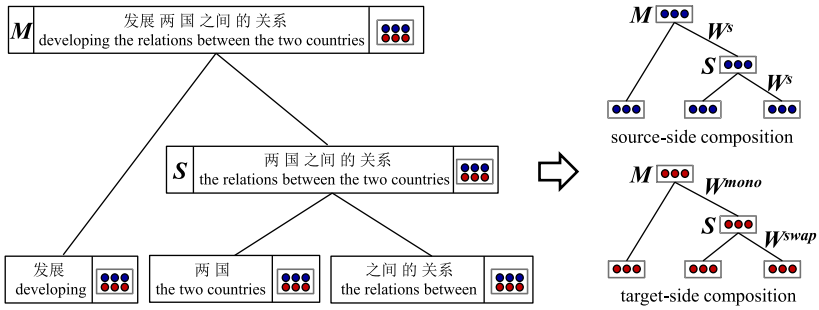

Fig. 6. The type-dependent recursive neural network in which vector composition for target side depends on the merging type during decoding.

3.3.1. Scoring Derivation Trees with RNN. Assuming we are given a derivation tree, as shown in Figure 1, we define the representations of the leaf nodes (lexical translation rules) as (s,t,p_s,p_t), in which s is the source phrase, t is the target phrase, p_s and p_t are the semantic vector representations for s, and t, respectively. p_s and p_t are learned using the BRAE model which has been already introduced in the previous section. We then define the representations of the non-terminal nodes (applying merging rules) as $(type,s',t',p'_s,p'_t)$, where *type* denotes how the two children are combined (*monotone* or *swap*) to generate this current node. s' is a source string, t' is a translation candidate (may be ungrammatical), which is different from t (normal natural language phrase). p'_s and p'_t are both learned with the type-dependent recursive neural networks.

For each non-terminal node, the semantic vector representation for the source string p'_s is generated in a same way no matter what type of merging rule we apply:

$$p'_s = f(W^s[p^l_s;p^r_s]+b^s), \tag{15}$$

in which, p^l_s and p^r_s are semantic vector representations of the source strings for the left and right child, respectively. W^s is the weight matrix of the neural network, b^s is the bias term, and $f = tanh(\cdot)$.

For the semantic vector representation for the target string p'_t, the weight matrix and bias term depend on the type of merging rule:

$$p'_t = f(W^{type}[p^l_t;p^r_t]+b^{type}), \tag{16}$$

where p^l_t and p^r_t are semantic vector representations of the target partial translations for the left and right child, respectively. $W^{type} = W^{mono}$ if we adopt the monotone merging rule, and $W^{type} = W^{swap}$ if we employ the swap merging rule. The bias term b^{type} is similar. Figure 6 shows the difference between the composition functions for the source and target sides. The source side utilizes the same composition function to obtain the parent vector, no matter whether the composition type is monotone (**M**) or swap (**S**). In contrast, the composition function for the target side depends on the composition type applied during decoding.

With the semantic vector representations for the source string and its target translation candidates in each node, we can measure the semantic distance gap between the source string and the translation candidate. Since the source- and target-side vector representations are in different semantic space, we design a transformation function from source to target and from target to source, as is done in the BRAE model. The semantic distance gap becomes

$$S^{gap}_{node}(p_s,p_t) = S^{gap}_{node}(p_s|p_t) + S^{gap}_{node}(p_t|p_s), \tag{17}$$

where $S^{gap}_{node}(p_s|p_t) = S^{gap}_{node}(p_t, f(W^{label}_s p_s + b^{label}_s))$, and the transformation function $f(\cdot)$ is similar to that in the BRAE model. Then, $S^{gap}_{node}(\cdot)$ is computed with the Euclidean distance. It is obvious that the smaller the semantic gap, the better the translation candidate.

Finally, the RNN score for the derivation tree is the sum of the semantic distance gap over all the tree nodes (including the leaf nodes and the non-terminal nodes):

$$S_{RNN} = \sum_{node} S^{gap}_{node}(p_s, p_t). \tag{18}$$

We hope that using this RNN model, the optimal derivation tree indeed leads to the best translation. To guarantee this, we need to design a good objective function for the RNN parameter training. Accordingly, we propose the max-margin training objective.

3.3.2. Max-Margin Training Objective. Given the bilingual sentences in the training data, we can perform forced decoding for the source sentence with the baseline BTG-based translation system to find the gold derivation trees *goldTs*, which lead to exactly the corresponding target reference sentences. At the same time, we can decode the source sentences of the training data (without considering the target reference sentences) using the baseline BTG-based translation model and obtain the k-best (k can be 100, 200, 500, ...) derivation trees *kbestTs*. Ideally, we want the RNN score of the gold derivation tree $S_{RNN}(goldT)$ to be much smaller than that of the k-best derivation tree $S_{RNN}(kbestT)$.

We first define a structured margin loss $\Delta(goldT, kbestT)$ for predicting a derivation tree $kbestT$ given a correct derivation tree $goldT$. The loss increases if the predicted derivation tree is more incorrect and far from the correct derivation tree. The discrepancy between the gold derivation tree and the k-best derivation tree is measured by counting the number of nodes $N(kbest)$ having different source phrase spans in the k-best tree with that in the gold tree.

$$\Delta(goldT, kbestT) = \sum_{d \in N(kbestT)} \kappa \mathbf{1}\{d \notin N(goldT)\}. \tag{19}$$

Different from natural language parsing (binary parsing), in which all the trees for a sentence contain the same number of nodes, different derivation trees for the same source sentence in translation have different numbers of nodes, since decoding in translation is based on phrases rather than words. As a result, we need to normalize the previous margin loss as follows:

$$\Delta^*(goldT, kbestT) = \Delta(goldT, kbestT) \times \frac{N(goldT)}{N(kbestT)}. \tag{20}$$

Following the method applying RNN in natural language parsing [Socher et al. 2013], we set $\kappa = 0.1$ in all of the experiments. Given the structured margin loss, we require that the RNN score of the gold derivation tree should be smaller up to this margin than that of the k-best derivation tree:

$$S_{RNN}(goldT) \leq S_{RNN}(kbestT) - \Delta^*(goldT, kbestT). \tag{21}$$

If m bilingual sentence pairs in the training data are successful in forced decoding, then the overall objective function becomes

$$\begin{aligned} J_\theta &= \frac{1}{m}\sum_i r_i(\theta) + \frac{\lambda}{2}||\theta||^2, \text{where} \\ r_i(\theta) &= \max_{\substack{gt \in goldTs \\ kt \in kbestTs}} (S_{RNN}(gt) + \Delta^*(gt, kt) - S_{RNN}(kt)). \end{aligned} \tag{22}$$

Intuitively, to minimize the preceding objective, the RNN score of the gold derivation tree gt is decreased, and the RNN score of the k-best derivation tree kt is increased.

3.3.3. Optimization Algorithm. Parameters θ in Equation (22) are different from that in the BRAE model, and θ here consists of three kinds of parameters.

(1) Vector composition parameters for the source language string: W^s and b^s.
(2) Vector composition parameters for the target language string: W^{mono}, b^{mono}, W^{swap}, and b^{swap}.
(3) Parameters of transformation functions between the source and the target vector embedding spaces: W_s^{label}, b_s^{label}, W_t^{label}, and b_t^{label}.

As the objective function in Equation (22) is not differentiable due to hinge loss, we generalize the gradient descent with the subgradient algorithm [Ratliff et al. 2007], which calculates a gradient-like direction. The subgradient of Equation (22) with respect to parameters θ becomes

$$\frac{\partial J}{\partial \theta} = \sum_i \frac{\partial S_{RNN}(gt)}{\partial \theta} - \frac{\partial S_{RNN}(kt)}{\partial \theta}. \tag{23}$$

To minimize the objective, we employ and compare two algorithms. First, we apply the SGD (stochastic gradient descent) algorithm just like we have done in the BRAE model:

$$\theta_t = \theta_{t-1} - \eta \frac{\partial J}{\partial \theta}, \tag{24}$$

where η is the learning rate. With a good empirical setting of the learning rate η, the optimization algorithm converges quite quickly (not more than five iterations in our experiments).

As an alternative, we also adopt the diagonal variant of AdaGrad [Duchi et al. 2011; Socher et al. 2013] for the objective minimization. To update the parameters, we first define $g_\tau \in \mathbb{R}^{M\times 1}$ to be subgradient at time step τ and $G_t = \sum_{\tau=1}^{t} g_\tau g_\tau^T$. The parameter update at time τ becomes

$$\theta_t = \theta_{t-1} - \eta (diag(G_t))^{-\frac{1}{2}} g_t. \tag{25}$$

We will compare the two optimization algorithms in the experiments.

4. EXPERIMENTS

With the learned translation model based on the recursive neural networks, we can directly find the best translation which has minimal semantic gap with the source language sentence in the vector embedding space. To retain as well the merits of the baseline BTG-based translation model, we keep in our RNN-based translation model the string-related features of the baseline, such as bidirectional phrasal probabilities,

bidirectional lexical weights, maximum entropy-based phrase reordering probability, language model, phrase-number, and translation length penalty.

Since two submodels have been proposed in the RNN-based framework, we will first test the power of the BRAE model to see whether it can effectively represent the source and target phrases with the semantic compact vectors, and then we test the effectiveness of the type-dependent RNN-based translation model on the end-to-end translation quality. Before detailing the experimental results, we introduce the hyper-parameter settings and the experimental data preparation.

4.1. Hyper-Parameter Settings

The hyper-parameters in the BRAE model and the type-dependent RNN-based model include the dimensionality of the word embedding n in Equation (2), the balance weight α in Equation (13), λs in Equation (14) and Equation (22), and the learning rate η in Equation (24) and Equation (25).

For the dimensionality n, we have tried two settings $n = 25, 50$[3] in our experiments. As the learning rate η controls the training efficiency and final performance, we try and compare four values $\{0.001, 0.005, 0.01, 0.05\}$. We draw the balance weight α from 0.05 to 0.5 with a step 0.05, and λs from $\{10^{-6}, 10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}\}$. The overall score of the BRAE and the type-dependent RNN-based model is employed to guide the search procedure. Finally, we choose $\alpha = 0.15$, $\lambda = 10^{-2}$.

4.2. SMT Data Preparation

The SMT evaluation is conducted on Chinese-to-English translation. The bilingual training data from LDC[4] contains approximately 2 million sentence pairs with 27.7M Chinese words and 31.9M English words. A 5-gram language model is trained on the Xinhua portion of the English Gigaword corpus and the English part of the bilingual training data. The NIST MT03 is used as the development data. NIST MT04, MT05, MT06 (news part), and MT08 (news part) are used as the test data. The specific statistics about the SMT evaluation data are given in Table I.

The SMT model parameters are tuned using the minimum error rate training (MERT) algorithm. In order to get the best performance for each system (including the baseline system), we run MERT four times with different initial parameters and choose the the parameters with the highest BLEU. Case-insensitive BLEU is employed as the evaluation metric. The statistical significance test is performed by the pairwise resampling approach [Koehn 2004].

In addition, we pre-train the word vector representations with the toolkit Word2Vec [Mikolov et al. 2013] on the large-scale monolingual data, including the aforementioned data for SMT. The monolingual data contains 1.06B words for Chinese and 1.12B words for English. To obtain high-quality bilingual phrase pairs as translation equivalents to train our BRAE model, we perform forced decoding on the bilingual training sentences and collect the phrase pairs used. After removing the duplicates, the remaining 1.12M high-quality bilingual phrase pairs (length ranging from 1 to 7) are obtained. For max-margin training in the type-dependent RNN-based translation model, we have chosen a subset of high-quality sentence pairs (about 100K sentence pairs) which have at least one gold derivation tree in the forced decoding.

[3]In theory, the larger the dimensionality the more powerful the semantic vector representation will be. Due to the time complexity, we just try these two settings in our experiments.

[4]LDC category numbers: LDC2000T50, LDC2002L27, LDC2003E07, LDC2003E14, LDC2004T07, LDC2005T06, LDC2005T10, and LDC2005T34.

Table I. Data Statistics for the SMT Evaluation

Data	Sen#		Word#	
	Chinese	English	Chinese	English
bilingual data	2,086,731	2,086,731	27,682,508	31,872,639
monolingual data		10,912,683		279,096,910
NIST03	919	919×4	24,155	114,056
NIST04	1,788	1,788×4	49,597	231,526
NIST05	1,082	1,082×4	29,893	141,695
NIST06	1,000	1,000×4	24,362	119,342
NIST08	691	691×4	17,707	92,524

4.3. Evaluation on Phrase Vector Representations with the BRAE Model

4.3.1. Specific Examples. To have a good intuition about the power of the BRAE model at learning semantic vector representations for phrases, we first show some examples in Table II. Intuitively, if the real-valued vector representations capture well the semantics of the phrases, the phrases sharing same or similar meanings should be close to each other. Therefore, we test the phrase embeddings in the following scenario: given the bilingual phrase training set, we search from the set the most semantically similar English phrases for any new input English phrase in the vector embedding space.

We compare three models for phrase embeddings: (1) unsupervised RAE, that learns the vector composition parameters using only the reconstruction error as objective; (2) average embedding (AvgEmbed), which gets the vector representation of the phrase by averaging the embeddings of all the words in the phrase; (3) our proposed bilingually-constrained RAE (BRAE).

Note that the input phrases are not in the phrase training set, and they span different number of words. The table shows that the unsupervised RAE can at most capture the syntactic property when the phrases are short. For example, the unsupervised RAE finds *do not want* for the input phrase *do not agree*. However, when the phrase becomes longer, the unsupervised RAE cannot even capture the syntactic property (e.g., the last example in the table). The AvgEmbed model performs similarly to the unsupervised RAE. It finds the similar phrase whose individual words have some similarity with those of the input phrase. For example, the AvgEmbed model finds a similar phrase *wednesday for a* for the input phrase *at a meeting*, since the words {*for, at*} are prepositions and {*wednesday, meeting*} are nouns, and they are near with each other in the monolingual word embedding space. As the AvgEmbed model is just a bag-of-words model and ignores the word-order information, it cannot well capture the semantics of the multiword phrases.[5] In contrast, our BRAE model learns the semantic meaning for each phrase no matter whether it is short or relatively long. This indicates that the proposed BRAE model is effective at learning semantic phrase vector representations.

A question may arise as to how and why the BRAE model is better at learning the phrase vector representations. From the specific examples in Table II, the three models AvgEmbed, Unsupervised RAE, and BRAE show different characteristics. As a bag-of-words model, AvgEmbed tries to capture the average information of the words in the phrase. It ignores the word-order information. Furthermore, the word embedding is learned just using the monolingual data and can only represent the syntax and little semantics of the word. This makes it difficult for the AvgEmbed model to obtain the semantic representation of the phrase. In contrast, the Unsupervised RAE

[5]We also calculate the bilingual phrase similarity for the lexical translation rules with the AvgEmbed model, and apply the similarity as a feature integrated in the baseline BTG-based translation system. We find that the overall performance in BLEU score is 34.9, showing that the AvgEmbed model has no positive effect in improving translation quality compared to the baseline with overall BLEU score 34.82.

Table II. Semantically Similar Phrases in the Training Set for the New Input Phrases

New Phrase	Unsupervised RAE	AvgEmbed	BRAE
military force	core force main force labor force	military action air forces expeditionary forces	military power military strength armed forces
at a meeting	to a meeting at a rate a meeting,	a meeting which meeting on wednesday for a	at the meeting during the meeting at the conference
do not agree	one can accept i can understand do not want	do not want do not meet will not do	do not favor will not compromise not to approve
each people in this nation	each country regards each country has its each other, and	the entire people is part of people people still consist of	every citizen in this country all the people in the country people all over the country

model takes the word-order information into consideration. Thus, this model well captures the syntactic patterns for short phrases, just as the first example *military force* demonstrates. Similar to the AvgEmbed model, the used word embedding is lack of semantics. Moreover, the unsupervised learning algorithm minimizes only the reconstruction error instead of the semantic loss. Consequently, the obtained phrase embedding is not good enough. However, our BRAE model considers all three aspects. First, the BRAE model not only minimizes the reconstruction error of the recursive autoencoder, but also minimizes the semantic loss of the translation equivalents. Second, the constraint-based training algorithm also updates the word embeddings using the counterpart constraints of the translation equivalents, making the word embedding capture much more semantics. Third, the BRAE model attempts to find an optimal binary composition structure with greedy search so that the word-order information is modeled. Therefore, our BRAE model can induce better word vector representations, as Table II shows.

4.3.2. Evaluation on Semantic Similarity using Phrase Vector Representations. With the semantic vector representations for phrases and the vector space transformation function, we can evaluate the BRAE model in the vector space by measuring the semantic similarity between a source phrase and its translation candidates in statistical machine translation. Here, we apply the BRAE model in our baseline BTG-based translation system to lexical translation rule pruning (also known as phrase table pruning), which discards entries whose semantic similarity is very low.

Pruning most of the phrase table without much impact on translation quality is very important for translation, especially in environments where memory and time constraints are imposed. Many algorithms have been proposed to deal with this problem, such as significance pruning [Johnson et al. 2007; Tomeh et al. 2009], relevance pruning [Eck et al. 2007], and entropy-based pruning [Ling et al. 2012; Zens et al. 2012]. These algorithms are based on corpus statistics, including co-occurrence statistics, phrase pair usage, and composition information. For example, significance pruning, which is proven to be a very effective algorithm, computes the probability, p-value, that tests whether a source phrase s and a target phrase t co-occur more frequently in a bilingual corpus than they happen just by chance. The higher the p-value, the more likely the phrase pair is to be spurious.

Our work has the same objective, but instead of using corpus statistics, we attempt to measure the quality of the phrase pair from the view of semantic meaning. Given a phrase pair (s, t), the BRAE model first obtains its semantic phrase representations (p_s, p_t), and then transforms p_s into target semantic space $p_s{}^*$ and p_t into

Table III. Comparison between BRAE-Based Pruning and Significance Pruning of Phrase Table

Method	Threshold	PhraseTable	MT03	MT04	MT05	MT06	MT08	ALL
Baseline		100%	35.81	36.91	34.69	33.83	27.17	34.82
BRAE	0.4	52%	35.94	36.96	35.00	34.71	27.77	**35.16**
	0.5	44%	35.67	36.59	34.86	33.91	27.25	**34.89**
	0.6	35%	35.86	36.71	34.93	34.63	27.34	**35.05**
	0.7	28%	35.55	36.62	34.57	33.97	27.10	**34.76**
	0.8	20%	35.06	36.01	34.13	33.04	26.66	34.04
Significance	8	48%	35.86	36.99	34.74	34.53	27.59	**35.13**
	12	36%	35.59	36.73	34.65	34.17	27.16	**34.72**
	16	25%	35.19	36.24	34.26	33.32	26.55	34.09
	20	18%	35.05	36.09	34.02	32.98	26.37	33.97

Note: Threshold means similarity in BRAE and negative-log-p-value in Significance. "ALL" combines the development and test sets. **Bold numbers** denote that the result is better than or comparable to that of baseline, and the **bold** results are not significantly ($p < 0.05$) better or worse than the baseline according to the pairwise resampling significance test method [Koehn 2004]. $n = 50$ is used for embedding dimensionality.

source semantic space $p_t{}^*$. We finally get two similarities $Sim(p_s{}^*, p_t)$ and $Sim(p_t{}^*, p_s)$. Phrase pairs that have low similarity are more likely to be noisy and more prone to be pruned. In experiments, we discard the phrase pair whose similarity in two directions is smaller than a threshold.[6]

Table III shows the comparison results between our BRAE-based pruning method and the significance pruning algorithm. We can see a common phenomenon in both of the algorithms: for the first few thresholds, the phrase table becomes smaller and smaller while the translation quality is not much decreased, but the performance jumps a lot at a certain threshold (16 for significance pruning, 0.8 for BRAE-based).

Specifically, the significance algorithm can safely discard 64% of the phrase table at its threshold 12 with only 0.1 BLEU loss in the overall test. In contrast, our BRAE-based algorithm can remove 72% of the phrase table at its threshold 0.7 with only 0.06 BLEU loss in the overall evaluation. When the two algorithms use a similar portion of the phrase table (35% in BRAE and 36% in Significance), the BRAE-based algorithm outperforms the significance algorithm on all the test sets except for MT04.

In order to compare these two pruning methods at the same percentage of phrase table, we first sort all the entries in the phrase table by semantic similarities or negative-log-p-value. Then, we respectively choose the same percentage $\{20\%, 30\%, 40\%, 50\%, 60\%\}$ of the phrase table[7] and evaluate their overall BLEU score. Figure 7 gives the results. We can see that the two pruning methods perform worse and worse as the phrase table becomes smaller. Compared to the significance pruning method, our BRAE model performs similarly or worse when retaining 40% or more of the phrase table. When keeping only 20% or 30% of the phrase table, the BRAE model slightly outperforms the significance method. It indicates that our BRAE model is a good alternative for phrase table pruning. Furthermore, our model is much more intuitive because it is directly based on the semantic similarity.

Besides pruning the phrase table with bilingual phrasal similarity, we further extend their usage and adopt $Sim(p_s{}^*, p_t)$ and $Sim(p_t{}^*, p_s)$ as two additional features to

[6]To avoid the situation that all translation candidates for a source phrase are pruned, we always keep the first ten-best according to the semantic similarity.

[7]The two phrase tables pruned by different methods share a large part of the entries. We find that for the settings $\{20\%, 30\%, 40\%, 50\%, 60\%\}$, there are respectively $\{26.6\%, 31.9\%, 42.5\%, 51.5\%, 58.4\%\}$ entries of the two tables are the same.

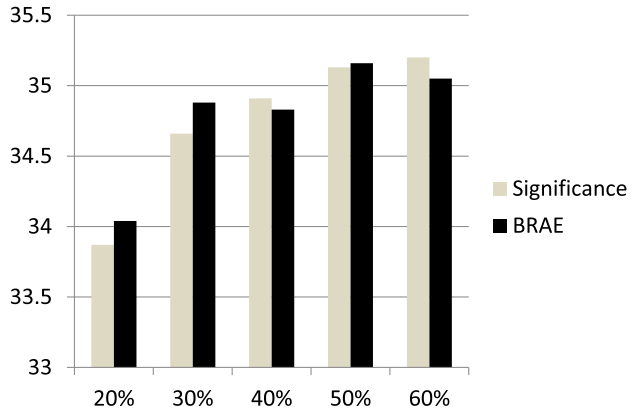

Fig. 7. Comparison between BRAE-based pruning and significance pruning of phrase table in the same percentage.

Table IV. Experimental Results when Integrating the Bilingual Phrasal Similarities in the Baseline BTG-Based Model as Two Features

Method	**MT03**	**MT04**	**MT05**	**MT06**	**MT08**	**ALL**
BTG	35.81	36.91	34.69	33.83	27.17	34.82
BTG-PhrasalSim-25	36.12	37.15	34.93	34.66	27.95	35.31
BTG-PhrasalSim-50	36.43	37.64	35.35	35.53	28.59	**35.84**$^{+}$

Note: "BTG-PhrasalSim" denotes the BTG-based translation model incorporated with the phrasal similarities as features. 25 and 50 denote the dimensionality of the vector space. "ALL" combines the all datasets. "+" means that the model significantly outperforms the baseline with $p < 0.01$.

be integrated into the baseline BTG-based translation model. As dimensionality of the vector embedding space is a hyper-parameter, we try two settings $n = 25$ and $n = 50$. Table IV shows the results.[8] No matter whether $n = 25$ or $n = 50$, the phrasal similarity feature can improve the translation quality. The biggest improvement over the baseline can be up to 1.42 BLEU score (MT08, when $n = 50$). It indicates that the bilingual phrasal similarity is an informative feature.

The specific examples and end-to-end translation results show that our proposed model for phrase semantic vector representations is effective. It indicates that the semantic vector representations for the lexical translation rules are very reliable for use in the type-dependent RNN-based translation model. In the next section, we will present the experimental results of the type-dependent RNN-based model to test how many improvements this model can obtain.

4.4. Evaluation on Type-Dependent RNN-Based Translation Model

4.4.1. Experimental Results. Table V shows the comparison results between the baseline BTG-based translation framework and our type-dependent RNN-based translation model optimized with the SGD algorithm. In this experiment, we keep the 200-best derivation trees for each source sentence in the max-margin training.

As shown in Table V, no matter what the dimensionality of the vector space n is, the RNN-based translation model can significantly improve the translation quality in the overall test data (with gains of more than 1.0 BLEU score). For the specific evaluation

[8]We also test the conventional phrase-based system Moses on the same test set and get the BLEU score 34.63 on the overall dataset, showing that our BTG-based translation system is a good baseline.

Table V. Experimental Results of the Type-Dependent RNN-Based Translation Model Optimized with SGD Algorithm

Method	MT03	MT04	MT05	MT06	MT08	ALL
BTG	35.81	36.91	34.69	33.83	27.17	34.82
BTG-RNN-25	36.64	37.54	36.02	35.36	28.88	**35.96**$^{+}$
BTG-RNN-50	36.83	37.78	35.76	35.73	29.50	**36.29**$^{+}$

Note: 25 and 50 denote the dimensionality of the vector space. "ALL" combines the development and test sets. "+" means that the model significantly outperforms the baseline with $p < 0.01$.

Table VI. Experimental Results of the Type-Dependent RNN-Based Translation Model Optimized with AdaGrad Algorithm

Method	MT03	MT04	MT05	MT06	MT08	ALL
BTG-RNN-25	36.47	37.41	35.27	35.32	28.71	**35.68**$^{+}$
BTG-RNN-50	36.68	37.53	35.59	35.40	28.85	**35.87**$^{+}$

datasets, the largest improvement is up to 2.33 BLEU score (MT08 for dimensionality 50). If we compare Table V with Table IV, we can find that the RNN-based translation model outperforms the one integrating phrasal similarity as features. Specifically, at the same condition $n = 50$, the BTG-RNN-50 system can obtain an improvement of 0.91 BLEU score in MT08 (statistically significant at the level of $p < 0.01$). It demonstrates well the superiority of the RNN-based translation model which directly optimizes the derivation structure by minimizing the semantic gap between the source-language input and its translation candidate.

It should be noted that although the improvement increases when we change the dimensionality from 25 to 50, the training time of the RNN-based model increased a lot (about two times slower). In real application, especially in large-scale experiments, we recommend that dimensionality of 25 may be a good choice.

To test whether our RNN-based translation model is sensitive to the optimization algorithm, we further run the same experiments with an alternative optimization algorithm AdaGrad [Duchi et al. 2011; Socher et al. 2013]. We report the translation results in Table VI. The figures in the table tell us that the RNN-based model optimized with the AdaGrad algorithm does not perform as well as that optimized with the SGD algorithm.[9] However, the model optimized with the AdaGrad algorithm can also significantly outperform the baseline BTG-based system. This indicates that the proposed RNN-based model can significantly improve the translation quality even though it is moderately influenced by the optimization algorithms. As the SGD algorithm performs better, we apply this optimization algorithm in the following experiments.

In the model training procedure, many factors can influence the training efficiency and the final translation quality. The learning rate η is one such factor. In order to have a better understanding of the relation between model training efficiency and translation quality, we try four different learning rates $\{0.001, 0.005, 0.01, 0.05\}$ and compare their performance. As the dimensionality of the embedding space $n = 25$ is more efficient, we choose this setting in this experiment. Table VII reports the results. We cannot get obvious conclusions from the table. Generally, the smaller learning rate (e.g., 0.001) consumes much more training time and leads to slightly better translation performance. To have a good balance between the training efficiency and translation quality, we choose learning rate $\eta = 0.01$ in all the experiments.

[9]Applying the pairwise resampling significance test [Koehn 2004], the results of AdaGrad are not significantly worse than that of SGD on the ALL dataset.

Table VII. Model Training Efficiency and Translation Quality when using Different Learning Rates

Learning Rate	Model Training Time	Translation Quality
0.001	36 hs	36.12
0.005	25 hs	35.71
0.01	16 hs	35.96
0.05	12 hs	35.58

Note: "hs" denotes hours and the translation quality is performed on the overall dataset.

Table VIII. Experimental Results for Different ks of k-Best

Method	RNN Time	MT03	MT04	MT05	MT06	MT08	ALL
BTG		35.81	36.91	34.69	33.83	27.17	34.82
BTG-RNN-k100	12 hs	36.47	37.65	35.55	35.02	28.50	**35.71**$^{+}$
BTG-RNN-k200	16 hs	36.64	37.54	36.02	35.36	28.88	**35.96**$^{+}$
BTG-RNN-k300	25 hs	36.59	37.71	35.65	35.50	29.18	**36.12**$^{+}$
BTG-RNN-k400	36 hs	36.80	38.04	36.06	35.96	29.21	**36.36**$^{+}$
BTG-RNN-k500	52 hs	36.99	37.82	36.17	36.06	29.70	**36.43**$^{+}$

Note: "RNN Time" denotes the training time of the RNN-based translation model. "hs" means hours. "+" means that the model significantly outperforms the baseline with $p < 0.01$.

4.4.2. The Impact of k-Best. As the type-dependent RNN-based model is trained on k-best derivation trees, k of k-best is an important factor that influences the quality of the learned RNN-based model and the final translation performance. Here, we conduct a deep analysis to see how the translation quality is affected by the capacity of k-best.

When we try different ks of k-best, we fix the dimensionality of the vector space to be 25, as we have done in testing the learning rate. We try five different ks ($k = 100, 200, 300, 400, 500$) in our experiments. Table VIII gives the detailed experimental results.

The figures in Table VIII show that the final translation performance overall the test sets can be improved slightly but stably as k becomes larger and larger, although there are several exceptions in the specific datasets (e.g., BLEU score of $k = 200$ on MT05 is higher than that of $k = 300$). The largest gain over the baseline BTG-based translation framework is obtained on the evaluation set NIST MT08 using $k = 500$. We also see that the performance improvement becomes smaller and smaller when k grows. For example, $k = 500$ performs almost the same on the overall dataset compared to $k = 400$. Considering the time complexity, we do not try bigger ks. Generally, the table shows that it benefits much from enlarging the k-best derivation space for the max-margin training in the RNN-based model. Considering both training time and final translation quality, we recommend that $k = 200$ as a good choice for large-scale experiments.

4.4.3. Semantic Gap vs. Translation Quality. BLEU score improvements indicate the success of the proposed model. As our type-dependent RNN-based model is optimized by requiring that correct translations should have small semantic gap with the source sentence, we may wonder whether smaller semantic gaps indeed lead to better translations when testing the RNN-based model (during decoding).

To answer this question, we have designed an experimental setting: for sentences in the test sets, if they are successful in forced decoding with their target references using

Table IX. Reference Translation vs. 1-Best Translation on Test Sets

	MT04	MT05	MT06	MT08
Successful Forced Decoding	297(16.6%)	173(15.99%)	200(20%)	92(13.3%)
Smaller Gap than 1-best	254(85.5%)	146(84.4%)	179(89.5%)	85(92.4%)

the RNN-based translation model, we check whether their semantic gap is smaller than the 1-best translation. We know that the target reference is almost always better than the 1-best translation. Therefore, if the target reference has the smaller semantic gap compared with the 1-best translation, the semantic gap and translation quality will be consistent and our model training is correct.

We conduct the experiment using our type-dependent RNN-based translation model with $n = 50$ for dimensionality and $k = 200$ for k-best. We demonstrate the experimental results in Table IX. In the row *Successful Forced Decoding*, we show the number of sentences that succeed in forced decoding. The figures in parentheses indicate the proportion of the total test set. It is easy to see that only a small number of the test sentences succeed in forced decoding. There are two main reasons. First, the test sets contain many unknown words whose translations are missing. Second, many test sentences are relatively long and the likelihood of combining translation candidates to match reference becomes lower and lower. Among the test sentences with successful forced decoding, the last row in Table IX gives the number of sentences for which the target reference translations have smaller semantic gaps than the 1-best translations. The figures in the parentheses show that for about 85% or even more sentences in all of the test sets, the reference translation has a smaller semantic gap. This indicates that better translation and smaller semantic gap are very consistent. It also demonstrates the correctness of the model training.

The preceding experimental setting has a disadvantage: the sentence pair needs to be successful with forced decoding. We know from Table IX that the percentage of sentence pairs that succeed in forced decoding is very low (lower than 20%). In order to figure out the relationship between semantic gap and translation quality, we design an alternative experimental setting. In this setting, we approximate the translation quality using BLEU scores and figure out the relationship between the semantic gap and BLEU score. The experiment is designed as follows: first, we generate the k-best list on the test set; then, we compute sentence-level BLEU scores and semantic gap for each hypothesis; finally, we plot the two on a scatter plot and compute correlation coefficient.

We randomly choose a subset of 200 source sentences from the test sets and obtain a 10-best list for each source sentence. Each translation hypothesis will be associated with a sentence-level BLEU score and a semantic gap value. To simplify the experiment, we normalize the BLEU score and semantic gap value for each sentence and sort, respectively, the 10-best hypotheses according to the BLEU score (descending order) and the semantic gap value (ascending order). Then, we examine the consistency between the order sorted by the BLEU score and that sorted by the semantic gap value. Figure 8 shows the result. In this figure, a point (x, y) (e.g., $(0.3, 0.5)$) denotes that the translation hypothesis ranks $10 \times x$ with respect to the BLEU score, while it ranks $10 \times y$ with respect to the semantic gap value. The larger area of the point, the more translation hypotheses for different source sentences share the same rank pair. From this figure, we can see that the semantic gap value can reflect the translation quality. We further test the Pearson correlation between the BLEU score and the semantic gap value and get $\gamma = 0.758$.

4.4.4. Human Analysis. To have better intuition, we illustrate some specific translation examples in Figure 9 to compare the RNN-based translation model and the baseline

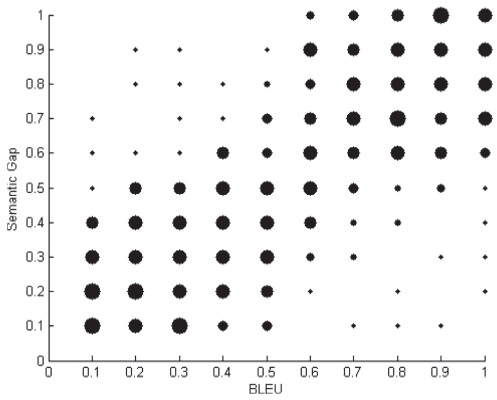

Fig. 8. Relationship between the semantic gap value and the BLEU score on 10-best translation hypotheses for each source sentence.

Example 1:
Source: 今天 我们 正在 面对 ***扩大 的 国际 事态***。
Pinyin: jīntiān wǒmen zhèngzaì miànduì ***kuòdà de guójì shìtài***.
Ref: Today we are facing ***an expanded international situation***.
Baseline: Today we are faced with ***increasing international events***.
RNN model: Today we are faced with ***the expansion of the international situation***.

Example 2:
Source: ***中国 提出*** 未来 15年 “ 重大 科学 研究 计划. ”
Pinyin: ***zhōngguó tíchū*** wèilái 15nián “ zhòngdà kēxué yānjiū jìhuà. ”
Ref: ***China proposes*** “ major research programs ” for next 15 years.
Baseline: ***China made*** “ major scientific research projects ” in the next 15 years.
RNN model: ***China proposed*** “ major scientific research projects ” in the next 15 years.

Example 3:
Source: 斯里兰卡 交战 双方 同意 ***本 月 下旬 在 日内瓦 谈判***.
Pinyin: sīlǐlánkǎ jiāozhàn shuāngfāng tóngyì ***běn yuè xiàxún zài rìnèiwǎ tánpàn***.
Ref: The two belligerent parties in Sri Lanka agreed to ***hold negotiations in Geneva late this month***.
Baseline: Sri Lanka warring parties agree to ***this month in Geneva talks***.
RNN model: Sri Lanka warring parties agree to ***talks in Geneva late this month***.

Fig. 9. Three specific translation examples comparing the RNN-based model with the baseline model.

BTG-based model. It should be noted that the search space for these two models are the same (for each source phrase, both models share the same set of translation candidates). The RNN-based model influences the decoding process by choosing the correct lexical and merging translation rules which lead to small semantic gaps between the translation candidates and the source-language string. The three examples listed in Figure 9 show different aspects of the proposed RNN-based model.

In the first example, the both models obtain the translation candidates for the Chinese phrase *kuòdà de guójì shìtài* with the monotone merging rules. For the single Chinese words *kuòdà* and *shìtài*, the English translations *increasing* and *events* are relatively reasonable. However, for the whole Chinese phrase, the RNN-based model

finds that the English translation *the expansion of the international situation* has a smaller semantic gap with the Chinese phrase than the English phrase *increasing international events* does. As a result, the RNN-based model yields a better translation.

For the second example, the translation candidates *China made* and *China proposed* for the Chinese phrase *zhōngguó tíchū* are both obtained through the lexical translation rules. The baseline BTG-based model lacks sufficient information that can distinguish which is better. In the RNN-based model, the candidate *China proposed* has a smaller semantic gap with *zhōngguó tíchū* than *China made*. From the view of the whole sentence, the translation candidate *China proposed* is more reasonable.

The last example further shows that the RNN-based model has a positive impact on phrase reordering. For the Chinese phrase *běn yuè xiàxún zài rìnèiwǎ tánpàn*, both models utilize two merging rules which combine the partial translations for three Chinese subphrases *běn yuè xiàxún*, *zài rìnèiwǎ*, and *tánpàn*. The baseline BTG-based model applies two monotone merging rules and gets the translation *this month in Geneva talks*. Fortunately, with the guidance of semantics in the vector embedding space, the RNN-based model observes that two swap merging rules can lead to the translation which has a smaller semantic gap with the Chinese phrase. Consequently, the RNN-based model yields a much better translation.

5. CONCLUSION AND FUTURE WORK

This article has presented a novel translation model with recursive neural networks which aims at minimizing the semantic distance gap between the source-language string and its translation candidates. First, we proposed the bilingually-constrained recursive auto-encoders to learn the semantic vector representations for lexical translation rules. Second, we introduced the type-dependent recursive neural networks to model the translation process and designed a max-margin objective function to learn the model parameters. The large-scale experiments on Chinese-to-English translation have shown that our RNN-based translation model can significantly outperform the state-of-the-art baseline.

Currently, we train our RNN-based translation model using k-best derivation trees to simulate the whole derivation space. In the future work, we plan to enhance our type-dependent RNN-based translation model by training it in the whole derivation space so as to obtain a much bigger improvement.

REFERENCES

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

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. 2003. A neural probabilistic language model. *J. Machine Learn. Res. 3*, 1137–1155.

Yoshua Bengio, Holger Schwenk, Jean-Sébastien Senécal, Fréderic Morin, and Jean-Luc Gauvain. 2006. Neural probabilistic language models. In *Innovations in Machine Learning*. Studies in Fuzziness and Soft Computing, Springer, Verlag, Berlin, Heidelberg, 137–186.

David Chiang. 2007. Hierarchical phrase-based translation. *Comput. Linguistics 33*, 2, 201–228.

Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th International Conference on Machine Learning*. 160–167.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *J. Machine Learn. Res. 12*, 2493–2537.

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 ACL*. 1370–1380.

John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *J. Machine Learn. Res. 12*, 2121–2159.

Matthias Eck, Stephen Vogal, and Alex Waibel. 2007. Estimating phrase pair relevance for translation model pruning. In *Proceedings of the Machine Translation Summit XI*.

Michel Galley, Jonathan Graehl, Kevin Knight, Daniel Marcu, Steve DeNeefe, Wei Wang, and Ignacio Thayer. 2006. Scalable inference and training of context-rich syntactic translation models. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 961–968.

Jianfeng Gao, Xiaodong He, Wen-tau Yih, and Li Deng. 2014. Learning continuous phrase represenations for translation modeling. In *Proceedings of the 52nd ACL*. 699–709.

John Howard Johnson, Joel Martin, George Foster, and Roland Kuhn. 2007. Improving translation quality by discarding most of the phrasetable. In *Proceedings of EMNLP*.

Liang Huang, Kevin Knight, and Aravind Joshi. 2006. Statistical syntax-directed translation with extended domain of locality. In *Proceedings of AMTA*. 66–73.

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

Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proceedings of EMNLP*. 388–395.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyes, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*. Association for Computational Linguistics, 177–180.

Peng Li, Yang Liu, and Maosong Sun. 2013. Recursive autoencoders for ITG-based translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

Wang Ling, Joao Graça, Isabel Trancoso, and Alan Black. 2012. Entropy-based pruning for phrase-based machine translation. In *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*. 962–971.

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*. 791–801.

Yang Liu, Qun Liu, and Shouxun Lin. 2006. Tree-to-string alignment template for statistical machine translation. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 609–616.

Tomas Mikolov. 2012. Statistical language models based on neural networks. Ph.D Dissertation. Brno University of Technology.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Proceedings of NIPS*.

Franz Josef Och and Hermann Ney. 2002. Discriminative training and maximum entropy models for statistical machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*. Association for Computational Linguistics, 295–302.

Nathan D. Ratliff, J. Andrew Bagnell, and Martin Zinkevich. 2007. (Approximate) Subgradient methods for structured prediction. In *Proceedings of the International Conference on Artificial Intelligence and Statistics*. 380–387.

Holger Schwenk. 2010. Continuous-space language models for statistical machine translation. *Prague Bullet. Math. Linguistics 93*, 137–146.

Holger Schwenk. 2012. Continuous space translation models for phrase-based statistical machine translation. In *Proceedings of the 24th COLING*. 1071–1080.

Richard Socher, John Bauer, Christopher D. Manning, and Andrew Y. Ng. 2013. Parsing with compositional vector grammars. In *Proceedings of ACL*.

Richard Socher, Christopher D. Manning, and Andrew Y. Ng. 2010. Learning continuous phrase representations and syntactic parsing with recursive neural networks. In *Proceedings of the NIPS-2010 Deep Learning and Unsupervised Feature Learning Workshop*.

Richard Socher, Jeffrey Pennington, Eric H. Huang, Andrew Y. Ng, and Christopher D. Manning. 2011. Semi-supervised recursive autoencoders for predicting sentiment distributions. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. 151–161.

Nadi Tomeh, Nicola Cancedda, and Marc Dymetman. 2009. Complexity-based phrase-table filtering for statistical machine translation. In *Proceedings of Summit XII*. 144–151.

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

Dekai Wu. 1997. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computat. Linguistics 23*, 3, 377–403.

Deyi Xiong, Qun Liu, and Shouxun Lin. 2006. Maximum entropy based phrase reordering model for statistical machine translation. In *Proceedings of ACL-COLING*. 505–512.

Richard Zens, Daisy Stanton, and Peng Xu. 2012. A systematic comparison of phrase table pruning techniques. In *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*. 972–983.

Jiajun Zhang, Shujie Liu, Mu Li, Ming Zhou, and Chengqing Zong. 2014a. Bilingually-constrained phrase embeddings for machine translation. In *Proceedings of the 52th Annual Meeting on Association for Computational Linguistics*.

Jiajun Zhang, Shujie Liu, Mu Li, Ming Zhou, and Chengqing Zong. 2014b. Mind the gap: Machine translation by minimizing the semantic gap in embedding space. In *Proceedings of the 28th AAAI*.

Jiajun Zhang, Feifei Zhai, and Chengqing Zong. 2011. Augmenting string-to-tree translation models with fuzzy use of source-side syntax. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. 204–215.

Jiajun Zhang, Feifei Zhai, and Chengqing Zong. 2013. Syntax-based translation with bilingually lexicalized synchronous tree substitution grammars. *IEEE Trans. Audio, Speech, Lang. Process. 21*, 8, 1586–1597.

Min Zhang, Hongfei Jiang, Aiti Aw, Haizhou Li, Chew Lim Tan, and Sheng Li. 2008. A tree sequence alignment-based tree-to-tree translation model. In *Proceedings of the Annual Meeting of the Association of Computational Linguistics (ACL)*. 559–567.

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

Received April 2014; revised July, August 2014; accepted October 2014