Mind the Gap: Machine Translation by Minimizing the Semantic Gap in Embedding Space

Jiajun Zhang[1], **Shujie Liu**[2], **Mu Li**[2], **Ming Zhou**[2] **and Chengqing Zong**[1]
[1]National Laboratory of Pattern Recognition, CASIA, Beijing, P.R. China
{jjzhang,cqzong}@nlpr.ia.ac.cn
[2]Microsoft Research Asia, Beijing, P.R. China
{shujliu,muli,mingzhou}@microsoft.com

Abstract

The conventional statistical machine translation (SMT) methods perform the decoding process by compositing a set of the translation rules which are associated with high probabilities. However, the probabilities of the translation rules are calculated only according to the cooccurrence statistics in the bilingual corpus rather than the semantic meaning similarity. In this paper, we propose a Recursive Neural Network (RNN) based model that converts each translation rule into a compact real-valued vector in the semantic embedding space and performs the decoding process by minimizing the semantic gap 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. Extensive experiments on Chinese-to-English translation show that our RNN-based model can significantly improve the translation quality by up to 1.68 BLEU score.

Introduction

The conventional statistical machine translation (SMT) models, such as phrase-based models (Koehn et al. 2007), formal syntax-based models (Chiang 2007; Xiong, Liu, and Lin 2006) and linguistically syntax-based models (Liu, Liu, and Lin 2006; Huang, Knight, and Joshi 2006; Galley et al. 2006; Zhang et al. 2008), perform the decoding process and generate the translation result by compositing a set of translation rules which are associated with high probabilities. The probabilities of the translation rules (e.g. the phrasal translation probabilities and the lexical weights in phrase-based and formal syntax-based models) are all computed based on the cooccurrence statistics of the rule's source- and target-sides in the bilingual corpus. However, the cooccurrence statistics is much biased to the bilingual corpus and is not sufficient to show whether the source- and target-sides in a translation rule are in the same meaning, especially for the low frequent but correct translation rules. Accordingly, the conventional SMT models cannot guarantee that the generated translations are in the most similar semantic meanings with the source-side inputs.

Copyright © 2014, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

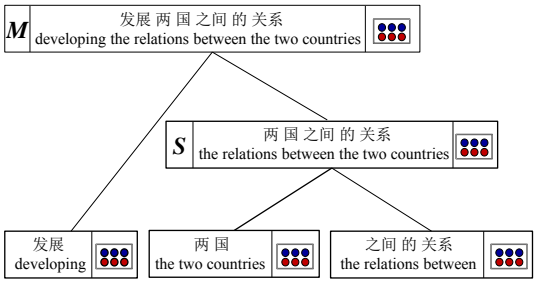

Figure 1: An example for the RNN-based translation model with (string and semantic vector) representations for source- and target-side in each node. The leaf nodes are translation rules (phrase pairs), and the nonterminals ***S*** and ***M*** determines which network will be applied to combine the children 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.

Aiming at retaining the semantic meaning during the translation process, we propose a Recursive Neural Network (RNN) based translation model. Like the previous SMT models, the RNN-based model induces the translation rules from the bitexts. Unlike them, the RNN-based model learns how to represent each lexical translation rule with two compact semantic vectors, and learns how to perform decoding using the merging type (*swap* or *monotone*) dependent recursive neural networks that attempt to find the best translation candidate having the minimal semantic gap with the source string.

Fig. 1 shows an example for our RNN-based translation model. The overall objective of our model is to search the best hidden derivation tree with the minimal sum of the semantic gaps in each node. The nodes in Fig. 1 are divided into two groups: the leaf nodes and the nonterminal nodes. Our RNN-based model designs two sub-models to handle the leaf nodes and the nonterminal nodes respectively.

Since the leaf nodes denote the basic translation rules which are directly induced from the bitexts, both of the

source- and target-sides of the leaf nodes are grammatical and the semantic gap between them should be a fixed value. We apply the bilingually-constrained recursive auto-encoders (BRAE) (Zhang et al. 2014) to semantically embed each source and target grammatical phrase with compact real-valued vectors and find the fixed semantic gaps. The BRAE is learned by minimizing the semantic gap between the high-quality translation equivalents and maximizing the semantic gap between non-translation pairs simultaneously. With the learned BRAE model, each translation rule is represented with 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).

Given the leaf nodes in tuples, another sub-model is proposed to learn how to composite any two children recursively. Following the bracketing transduction grammars (Wu 1997), we adopt two types of composition operators: monotone and swap. We then design two type-dependent networks for monotone composition and swap composition respectively. The networks learn four functions: two vector composition functions for the source and target language strings respectively, and two transformation functions that transform the semantic vectors between the source-side embedding space and the target-side embedding space. The source-side vector composition function takes the source-side vectors in the left and right children as input, and outputs a vector to represent the semantic meaning of the combined source string. The source-side vector composition function is the same for both monotone and swap operator. As target-side strings are combined in different ways according to monotone or swap composition, two target-side vector composition functions are involved: one for monotone operator and the other for swap operator. The four 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 kbest trees generated by the conventional SMT models.

With the learned RNN-based translation model, we conduct the large-scale experiments on Chinese-to-English translation. The experimental results show that our RNN-based model can significantly outperform the state-of-the-art, with an improvement up to 1.68 BLEU score.

Related Work

In the 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 the language model (Duh et al. 2013; Vaswani et al. 2013), more context usage for target language word prediction and the sparsity problem in translation probability estimation (Mikolov et al. 2010; Auli et al. 2013; Kalchbrenner and Blunsom 2013; Liu et al. 2013; Zou et al. 2013), and the phrase reordering problem (Li, Liu, and Sun 2013).

For language modelling in the statistical machine translation, rather rather depending on the Markov assumptions, the recurrent neural network based language model (Duh et al. 2013; Vaswani et al. 2013) represents the sequence of words with continuous vectors and can make full use of the whole history information before the current word in target language. This model has shown a significant reduction of the perplexity of the language model and can improve the translation quality.

Besides the target-side history words, more source-side context can lead to better prediction of target word translation. The works (Mikolov et al. 2010; Auli et al. 2013; Kalchbrenner and Blunsom 2013; Liu et al. 2013) maps both of the source-side context and the target-side history into a real-valued vector, and utilizes the continuous vector to better predict the target word generation. Zou et al. (2013) learn word embeddings with bilingual constraints and augment the lexical translation probability.

Phrase reordering is an important problem in statistical machine translation. Instead of using lexical words as concrete features, (Li, Liu, and Sun 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 any source- and target-side phrases, and the model to perform decoding by minimizing the semantic gap between a source string and its translation candidate.

RNN-based Translation Framework

This section presents our proposed RNN-based translation framework. First, we introduce the formal syntax-based baseline translation system, namely the bracketing transduction grammar (BTG) based system. Then, we propose the bilingually-constrained recursive auto-encoders to semantically embed each phrasal translation rules with compact real-valued vectors. Finally, we present the RNN-based translation model with the objective to minimize the semantic gap between a source string and its translation candidates.

BTG-based Translation Model

The BTG-based translation (Wu 1997; Xiong, Liu, and Lin 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.

During decoding, the source language sentence is first divided into phrases (sequence of words), then the lexical translation rule $A \rightarrow (x, y)$ translates each source phrase x into target phrase y and forms a block A. The monotone merging rule $A \rightarrow [A^l, A^r]$ (or the swap merging rule $A \rightarrow \langle A^l, A^r \rangle$) combines the two neighboring blocks into a bigger one until the whole source sentence is covered.

The lexical translation rule $A \rightarrow (x, y)$ plays the same role as the phrasal translation pairs (tuples consisting of a source phrase and its target translation hypothesis) in the conventional phrase-based translation models (Koehn et al. 2007). The monotone merging rule $A \rightarrow [A^l, A^r]$ combines the two consecutive 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.

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). 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_i's are feature functions, such as the bidirectional phrasal translation probabilities, the language model and the reordering model. λ's are feature weights.

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 known that the lexical translation rule 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 apply the 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 BRAE model for learning the way of semantic composition.

Word Vector Representations Recently, the word vector representations are typically learned with the Deep Neural Networks (DNN), which convert a word into a dense, low dimensional, real-valued vector (Bengio et al. 2003; 2006; Collobert and Weston 2008; Mikolov et al. 2013). 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:

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

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 Eq. 2. The Recursive Auto-encoder (RAE) learns the vector representation of the phrase by recursively combining two children vectors in a bottom-up manner (Socher et al. 2011). Fig. 2 illustrates an instance of a RAE applied to a binary tree, in which a standard auto-encoder (in box) is re-used at each node. For two children $c_1 = x_1$ and $c_2 = x_2$, the standard auto-encoder computes the parent vector y_1 as follows:

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

Where $W^{(1)} \in \mathbb{R}^{n \times 2n}$, $[c_1; c_2] \in \mathbb{R}^{2n \times 1}$, and $f = tanh(\cdot)$.

The standard auto-encoder then reconstructs the children:

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

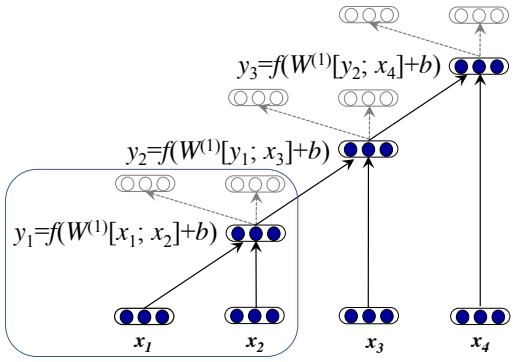

Figure 2: A recursive auto-encoder for a four-word phrase. The empty nodes are the reconstructions of the input.

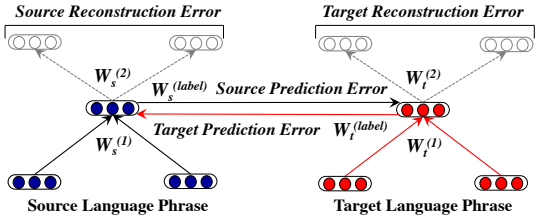

Figure 3: An illustration of the bilingual-constrained recursive auto-encoders. The two phrases are translation equivalents induced with forced decoding with the baseline SMT.

Finally the standard auto-encoder tries to minimize the reconstruction errors between inputs and reconstructions:

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

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

The BRAE model Without supervision, the above unsupervised method can only induce general representations of the multi-word phrases. Although no gold semantic phrase vector representations exist for supervision, we know the fact that the translation equivalents should share the same semantic meaning and thus should share the same semantic vector representation ideally. Therefore, the source phrase and the target phrase in a translation equivalent can supervise each other to induce their semantic meanings. Accordingly, we adopt our proposed Bilingually-constrained Recursive Auto-encoders (Zhang et al. 2014) (Fig. 3 shows the network structure). For a phrase pair (s, t), two kinds of errors are involved:

1. **reconstruction error** $E_{rec}(s, t; \theta)$: how well the learned vectors p_s and p_t represent the phrase s and t?

$$E_{rec}(s, t; \theta) = E_{rec}(s; \theta) + E_{rec}(t; \theta) \quad (6)$$

2. **semantic error** $E_{sem}(s, t; \theta)$: what is the semantic distance between the learned vector representations p_s and p_t?

Since word embeddings for two languages are learned separately and locate in different vector space, we do not

enforce the phrase embeddings in two languages to be in the same semantic vector space. We suppose there is a transformation between the two semantic embedding spaces. Thus, the semantic distance is bidirectional:

$$E_{sem}(s,t;\theta) = E_{sem}(s|t,\theta) + E_{sem}(t|s,\theta) \quad (7)$$

Where $E_{sem}(s|t,\theta) = E_{sem}(p_t, f(W_s^l p_s + b_s^l))$ and $E_{sem}(s|t,\theta)$ is then calculated as follows:

$$E_{sem}(s|t,\theta) = \frac{1}{2}||p_t - f(W_s^l p_s + b_s^l)||^2 \quad (8)$$

We then further enhance the semantic error with both translation equivalents and non-translation pairs [1], and the corresponding max-semantic-margin error becomes:

$$\begin{aligned} E^*_{sem}(s|t,\theta) = max\{0, E_{sem}(s|t,\theta) \\ - E_{sem}(s|t',\theta) + 1\} \end{aligned} \quad (9)$$

$E^*_{sem}(t|s,\theta)$ can be calculated in exactly the same way. For the phrase pair (s,t), the joint error is:

$$E(s,t;\theta) = \alpha E_{rec}(s,t;\theta) + (1-\alpha)E_{sem}(s,t;\theta) \quad (10)$$

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

$$J_{BRAE} = \frac{1}{N}\sum_{(s,t)\in(S,T)} E(s,t;\theta) + \frac{\lambda}{2}||\theta||^2 \quad (11)$$

The parameters θ can be divided into the source-side parameters θ_s and the target-side parameters θ_t. As seen from Fig. 3 that if the target phrase representation p_t is available, the optimization of the source-side parameters becomes a supervised learning problem. We apply the Stochastic Gradient Descent (SGD) algorithm to optimize each parameter. Word vector representations θ_L are initialized with a DNN toolkit Word2Vec (Mikolov et al. 2013) using the large-scale monolingual data, and other parameters are randomly initialized.

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 apply 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, and 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 pre-defined 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.

[1]For each translation equivalent, we randomly change the words in the target phrase and obtain a non-translation pair.

After parameter training, the BRAE model can learn a semantic vector representation for each source and target phrase respectively. Thus, each lexical translation rule can be represented with two semantic compact vectors.

RNN-based Translation Model

With translation rules represented as the semantic compact vectors, we propose the RNN-based model to find for a test source language sentence the best derivation tree using a CKY algorithm. For ease of exposition, we first describe how to score an existing derivation tree in which each node consists of the string and vector representations.

Scoring Derivation Trees with RNN Assuming we are given a derivation tree as shown in Fig 1. We define the representations of the leaf nodes (lexical translation rules) as (sp, tp, sv, tv) in which sp is the source phrase, tp is the target phrase, sv and tv are the semantic vector representations for sp and tp respectively. sv and tv are learned using the BRAE model. We then define the representations of the non-terminal nodes (merging rules) as $(type, sp', tp', sv', tv')$ where $type$ denotes how the two children are combined (*monotone* or *swap*) to generate this current node. sp' is a source phrase, tp' is a translation candidate (may be not in grammar) which is different from tp (normal natural language phrase). sv' and tv' are both learned with the type-dependent recursive neural networks.

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

$$sv' = f(W^s[sv^l; sv^r] + b^s) \quad (12)$$

in which, sv^l and sv^r are semantic vector representations of the source phrases 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 of the target phrase tv', the weight matrix and bias term depend on the type of the merging rule:

$$tv' = f(W^{type}[tv^l; tv^r] + b^{type}) \quad (13)$$

where tv^l and tv^r 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.

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

$$S^{node}_{gap}(sv, tv) = S^{node}_{gap}(sv|tv) + S^{node}_{gap}(tv|sv) \quad (14)$$

where $S^{node}_{gap}(sv|tv) = S^{node}_{gap}(tv, f(W^l_s sv + b^l_s))$, and $S^{node}_{gap}(\cdot)$ is computed with 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):

$$S_{RNN} = \sum_{node} S_{gap}^{node}(sv, tv) \tag{15}$$

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

Max-Margin Training Objective Given the bilingual sentences in the training data, we can perform forced decoding for the source sentence to find the gold derivation trees $goldTs$ which lead to exactly the corresponding translation reference. At the same time, we can decode the source sentences of the training data with the baseline BTG-based translation model and find the kbest (k can be 100, 200, 500, ...) derivation trees $kbestTs$. Ideally, we want the RNN score of the gold derivation tree $S_{RNN}(goldT)$ is much smaller than that of the kbest tree $S_{RNN}(kbestT)$.

We first define a structured margin loss $\Delta(goldT, kbestT)$. The discrepancy between the gold tree and the kbest tree is measured by counting the number of nodes $N(kbest)$ having different source phrase spans in the kbest tree with that in the gold tree.

$$\Delta(goldT, kbestT) = \sum_{d \in N(kbestT)} \kappa \mathbf{1}\{d \notin N(goldT)\} \tag{16}$$

As different derivation trees have different number of nodes in statistical machine translation, we normalize the above margin loss as follows:

$$\Delta^*(goldT, kbestT) = \Delta(goldT, kbestT) \times \frac{N(goldT)}{N(kbestT)} \tag{17}$$

Following (Socher et al. 2013b), we set $\kappa = 0.1$ in the experiments. And we require that the RNN score of the gold derivation tree will be smaller up to a margin to the kbest derivation trees:

$$S_{RNN}(goldT) \leq S_{RNN}(kbestT) - \Delta^*(goldT, kbestT) \tag{18}$$

This leads to the regularized function over all the m bilingual sentence pairs with successful forced decoding:

$$\begin{aligned} J_\theta &= \frac{1}{m}\sum_i r_i(\theta) + \frac{\lambda}{2}||\theta||^2, where \\ r_i(\theta) &= \max_{\substack{gt \in goldTs \\ kt \in kbestTs}} (S_{RNN}(gt) + \Delta(gt, kt) \\ &\quad - S_{RNN}(kt)) \end{aligned} \tag{19}$$

Intuitively, to minimize this objective, the RNN score of the gold derivation tree gt is decreased and the RNN score of the kbest derivation tree kt is increased.

We follow (Socher et al. 2013b) and adopt the diagonal variant of AdaGrad (Duchi, Hazan, and Singer 2011) to optimize the above objective function for parameter training.

Experiments

With the learned RNN-based model, the baseline BTG-based translation framework will be enhanced by the semantic continuous vector grammars. The RNN score of the derivation tree will serve as another informative feature (besides the model features of the baseline) to search for the optimal translation candidate during decoding.

Hyper-Parameter Settings

The hyper-parameters in the BRAE model and the RNN-based model include the dimensionality of the word embedding n in Eq. 2, the balance weight α in Eq. 10, λs in Eq. 11 and Eq. 19.

For the dimensionality n, we have tried two settings $n = 25, 50$ in our experiments. We draw α from 0.05 to 0.5 with step 0.05, and λs from $\{10^{-6}, 10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}\}$. The overall score of the BRAE and RNN-based model is employed to guide the search procedure. Finally, we choose $\alpha = 0.15$, $\lambda = 10^{-2}$.

SMT Setup

The SMT evaluation is conducted on Chinese-to-English translation. The bilingual training data from LDC [2] 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 bilingual training data. The NIST MT03 is used as the development data. NIST MT05, MT06 and MT08 (news data) are used as the test data. Case-insensitive BLEU is employed as the evaluation metric. The statistical significance test is performed by the re-sampling approach (Koehn 2004). In order to get the best performance for each system (including the baseline), we run MERT four times with different initial parameters and choose the parameters with the highest BLEU.

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 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 bilingual phrase pairs (length ranging from 1 to 7) are obtained [3]. For max-margin training in the 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.

Experimental Results and Analysis

Experimental Results Table 1 shows the comparison results between the baseline BTG-based translation frame-

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

[3] Training the BRAE model is very efficient, and it can run in a laptop.

Method	MT03	MT05	MT06	MT08	ALL
BTG	35.81	34.69	33.83	27.17	33.72
BTG-RNN-25	36.47	35.27	35.32	28.71	$\mathbf{34.85}^{+}$
BTG-RNN-50	36.68	35.59	35.40	28.85	$\mathbf{35.17}^{+}$

Table 1: Experimental results of the RNN-based translation model. 25 and 50 denotes the dimensionality of the vector space. "ALL" combines the development and test sets. "+" means that the model significantly outperforms the baseline with $p < 0.01$.

work and the RNN augmented translation model. In this experiment, we keep 200-best derivation trees for each source sentences in the max-margin training.

As shown in Table 1, no matter what the dimensionality of the vector space n is, the RNN augmented 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 data sets, the largest improvement can be up to 1.68 BLEU score (MT08 for the dimensionality 50). We also see that the translation quality can be improved slightly if the dimensionality of the semantic vector space is enlarged from 25 to 50. It should be noted that, the training time of the RNN-based model increased a lot when we change the dimensionality of 25 to 50 (about 2 times slower). Therefore, we believe that the dimensionality of 25 is a good choice, especially in the large-scale experiments.

Analysis The experimental results above have successfully shown the effectiveness of our proposed RNN-based translation model. As the RNN-based translation model attempts to minimize the semantic gap between the source string and its translation candidates, we here conduct a human evaluation to test whether the RNN augmented translation model performs much better than the baseline BTG-based framework on retaining the semantic meaning of the source sentences.

We randomly choose a subset 200 sentences from MT06 test set, and we then ask two bilingual speakers (Rater 1 and Rater 2) to compare the results of two translation systems BTG-RNN-50 and BTG. Table 2 presents the comparison statistics. The figures in Table 2 show that the two raters both report that on about 40% of sentences the system BTG-RNN-50 performs better. The inter-rater agreement Cohen Kappa κ is 0.52 in the evaluation. These results indicate that our RNN augmented translation model does very well in retaining the semantic meaning of the source sentence.

As the RNN-based model is trained on the kbest derivation trees, k of kbest 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 the kbest.

When we try different ks of kbest, we fix the dimensionality of the vector space to be 25. It is due to two reasons. For one hand, the time complexity of the max-margin model training is not too high. For the other hand, setting the dimensionality to be 25 has shown pretty good in Table 1. We

Rater	$<$	$=$	$>$
Rater 1	38	83	79
Rater 2	45	71	84

Table 2: Human evaluation results when comparing the translation system BTG-RNN-50 with the baseline system BTG on 200 sentences subset of MT06. "$>$" means the number of sentences on which the system BTG-RNN-50 performs better than the baseline system BTG in keeping the semantic meaning of the source sentence.

Method	MT03	MT05	MT06	MT08	ALL
BTG	35.81	34.69	33.83	27.17	33.72
BTG-RNN-k100	36.35	35.11	35.03	28.40	$\mathbf{34.52}^{+}$
BTG-RNN-k200	36.47	35.27	35.32	28.71	$\mathbf{34.85}^{+}$
BTG-RNN-k300	36.66	35.45	35.48	28.69	$\mathbf{34.96}^{+}$
BTG-RNN-k400	36.73	35.61	35.67	28.77	$\mathbf{35.11}^{+}$
BTG-RNN-k500	36.87	35.74	35.65	28.92	$\mathbf{35.38}^{+}$

Table 3: Experimental results for different ks of kbest. "+" means that the model significantly outperforms the baseline with $p < 0.01$.

try five different ks ($k = 100, 200, 300, 400, 500$) in the experiments. Table 3 gives the detailed experimental results.

The figures in Table 3 show that the final translation performance can be improved slightly but stably as the k becomes larger and larger. The largest gains over the baseline BTG-based translation framework can be up to 1.84 BLEU score (MT06 for $k = 400$). This indicates that it benefits much from enlarging the kbest derivation space for the max-margin training in the RNN-based model.

Conclusion and Future Work

This paper has presented an augmented translation model with the recursive neural networks which aim at minimizing the semantic distance gap between the source language string and its translation candidates. First, we presented the bilingually-constrained recursive auto-encoders to learn the semantic vector representation for each lexical translation rule. 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 augmented translation model can significantly outperform the baseline.

Currently, we train our RNN-based translation model using kbest 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 a larger derivation space (e.g. derivation forest) so as to obtain a much bigger improvement.

Acknowledgments

We thank Feifei Zhai and anonymous reviewers for their valuable comments. The research work has been partially

funded by the Natural Science Foundation of China under Grant No. 61333018 and 61303181, and Hi-Tech Research and Development Program (863 Program) of China under Grant No. 2012AA011102.

References

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

Bengio, Y.; Ducharme, R.; Vincent, P.; and Jauvin, C. 2003. A neural probabilistic language model. *Journal of Machine Learning Research* 3:1137–1155.

Bengio, Y.; Schwenk, H.; Senécal, J.-S.; Morin, F.; and Gauvain, J.-L. 2006. Neural probabilistic language models. In *Innovations in Machine Learning*. 137–186.

Chiang, D. 2007. Hierarchical phrase-based translation. *computational linguistics* 33(2):201–228.

Collobert, R., and Weston, J. 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.

Duchi, J.; Hazan, E.; and Singer, Y. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *The Journal of Machine Learning Research* 2121–2159.

Duh, K.; Neubig, G.; Sudoh, K.; and Tsukada, H. 2013. Adaptation data selection using neural language models: Experiments in machine translation. In *51st Annual Meeting of the Association for Computational Linguistics*, 678–683.

Galley, M.; Graehl, J.; Knight, K.; Marcu, D.; DeNeefe, S.; Wang, W.; and Thayer, I. 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*, 961–968. Association for Computational Linguistics.

Huang, L.; Knight, K.; and Joshi, A. 2006. Statistical syntax-directed translation with extended domain of locality. In *Proceedings of AMTA*, 66–73.

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

Koehn, P.; Hoang, H.; Birch, A.; Callison-Burch, C.; Federico, M.; Bertoldi, N.; Cowan, B.; Shen, W.; Moran, C.; Zens, R.; et al. 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*, 177–180. Association for Computational Linguistics.

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

Li, P.; Liu, Y.; and Sun, M. 2013. Recursive autoencoders for itg-based translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

Liu, L.; Watanabe, T.; Sumita, E.; and Zhao, T. 2013. Additive neural networks for statistical machine translation. In *51st Annual Meeting of the Association for Computational Linguistics*, 791–801.

Liu, Y.; Liu, Q.; and Lin, S. 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*, 609–616. Association for Computational Linguistics.

Mikolov, T.; Karafiát, M.; Burget, L.; Cernockỳ, J.; and Khudanpur, S. 2010. Recurrent neural network based language model. In *INTERSPEECH*, 1045–1048.

Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G.; and Dean, J. 2013. Distributed representations of words and phrases and their compositionality. In *Proceedings of NIPS*.

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

Socher, R.; Pennington, J.; Huang, E. H.; Ng, A. Y.; and Manning, C. D. 2011. Semi-supervised recursive autoencoders for predicting sentiment distributions. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 151–161.

Socher, R.; Perelygin, A.; Wu, J. Y.; Chuang, J.; Manning, C. D.; Ng, A. Y.; and Potts, C. 2013b. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

Vaswani, A.; Zhao, Y.; Fossum, V.; and Chiang, D. 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.

Wu, D. 1997. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computational linguistics* 23(3):377–403.

Xiong, D.; Liu, Q.; and Lin, S. 2006. Maximum entropy based phrase reordering model for statistical machine translation. In *Proceedings of ACL-COLING*, 505–512.

Zhang, M.; Jiang, H.; Aw, A.; Li, H.; Tan, C. L.; and Li, S. 2008. A tree sequence alignment-based tree-to-tree translation model. In *ACL*, 559–567.

Zhang, J.; Liu, S.; Li, M.; Zhou, M.; and Zong, C. 2014. Bilingually-constrained phrase embeddings for machine translation. In *Proceedings of the 52th Annual Meeting on Association for Computational Linguistics*. Association for Computational Linguistics.

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