Phrase Pair Rescoring with Term Weightings for Statistical Machine Translation

Bing Zhao Stephan Vogel Alex Waibel
Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA, 15213, USA
{bzhao, vogel+, ahw}@cs.cmu.edu

Abstract

We propose to score phrase translation pairs for statistical machine translation using term weight based models. These models employ *tf.idf* to encode the weights of content and non-content words in phrase translation pairs. The translation probability is then modeled by similarity functions defined in a vector space. Two similarity functions are compared. Using these models in a statistical machine translation task shows significant improvements.

1 Introduction

Words can be classified as content and functional words. Content words like verbs and proper nouns are more informative than function words like "to" and "the". In machine translation, intuitively, the informative content words should be emphasized more for better adequacy of the translation quality. However, the standard statistical translation approach does not take account how informative and thereby, how important a word is, in its translation model. One reason is the difficulty to measure how informative a word is. Another problem is to integrate it naturally into the existing statistical machine translation framework, which typically is built on word alignment models, like the well-known IBM alignment models (Brown et al 1993).

In recent years there has been a strong tendency to incorporate phrasal translation into statistical machine translation. It directly translates an n-gram from the source language into an m-gram in the target language. The advantages are obvious: It has built-in local context modeling, and provides reliable local word reordering. It has multi-word translations, and models a word's conditional fertility given a local context. It captures idiomatic phrase translations and can be easily enriched with bilingual dictionaries. In addition, it can compensate for the segmentation errors made during preprocessing, i.e. word segmentation errors of Chinese. The advantage of using phrase-based translation in a statistical framework has been shown in many studies such as (Koehn et al. 2003; Vogel et al. 2003; Zens et al. 2002; Marcu and Wong, 2002). However, the phrase translation pairs are typically extracted from a parallel corpus based on the Viterbi alignment of some word alignment models. The leads to the question what probability should be assigned to those phrase translations. Different approaches have been suggested as using relative frequencies (Zens et al. 2002), calculate probabilities based on a statistical word-to-word dictionary (Vogel et al. 2003) or use a linear interpolation of these scores (Koehn et al. 2003).

In this paper we investigate a different approach with takes the information content of words better into account. Term weighting based vector models are proposed to encode the translation quality. The advantage is that term weights, such as *tf.idf,* are useful to model the informativeness of words. Highly informative content words usually have high *tf.idf* scores. In information retrieval this has been successfully applied to capture the relevance of a document to a query, by representing both query and documents as term weight vectors and use for example the cosine distance to calculate the similarity between query vector and document vector. The idea now is to consider the source phrase as a "query", and the different target phrases extracted from the bilingual corpus as translation candidates as a relevant "documents". The cosine distance is then a natural choice to model the translation probability.

Our approach is to apply term weighting schemes to transform source and target phrases into term vectors. Usually content words in both source and target languages will be emphasized by large term weights. Thus, good phrase translation pairs will share similar contours, or, to express it in a different way, will be close to each other in the term weight vector space. A similarity function is then defined to approximate translation probability in the vector space.

The paper is structured as follows: in Section 2, our phrase-based statistical machine translation system is introduced; in Section 3, a phrase translation score function based on word translation probabilities is explained, as this will be used as a baseline system; in Section 4, a vector model based on *tf.idf* is proposed together with two similarity functions; in Section 5, length regularization and smoothing schemes are explained briefly; in Section 6, the translation experiments are presented; and Section 7 concludes with a discussion.

2 Phrase-based Machine Translation

In this section, the phrase-based machine translation system used in the experiments is briefly described: the phrase based translation models and the decoding algorithm, which allows for local word reordering.

2.1 Translation Model

The phrase-based statistical translation systems use not only word-to-word translation, extracted from bilingual data, but also phrase-to phrase translations. . Different types of extraction approaches have been described in the literature: *syntax-based*, *word-alignment-based, and genuine phrase alignment models*. The syntax-based approach has the advantage to model the grammar structures using models of more or less structural richness, such as the syntax-based alignment model in (Yamada and Knight, 2001) or the Bilingual Bracketing in (Wu, 1997). Popular word-alignment-based approaches usually rely on initial word alignments from the IBM and HMM alignment models (Och and Ney, 2000), from which the phrase pairs are then extracted. (Marcu and Wong 2002) and (Zhang et al. 2003) do not rely on word alignment but model directly the phrase alignment.

Because all statistical machine translation systems search for a globally optimal translation using the language and translation model, a translation probability has to be assigned to each phrase translation pair. This score should be meaningful in that better translations have a higher probability assigned to them, and balanced with respect to word translations. Bad phrase translations should not win over better word for word translations, only because they are phrases.

Our focus here is not phrase extraction, but how to *estimate* a reasonable probability (or score) to better represent the translation quality of the extracted phrase pairs. One major problem is that most phrase pairs are seen only several times, even in a very large corpus. A reliable and effective estimation approach is explained in section 3, and the proposed models are introduced in section 4.

In our system, a collection of phrase translations is called a *transducer*. Different phrase extraction methods result in different transducers. A manual dictionary can be added to the system as just another transducer. Typically, one source phrase is aligned with several candidate target phrases, with a score attached to each candidate representing the translation quality.

2.2 Decoding Algorithm

Given a set of transducers as the translation model (i.e. phrase translation pairs together with the scores of their translation quality), decoding is divided into several steps.

The first step is to build a lattice by applying the transducers to the input source sentence. We start from a lattice, which has as its only path the source sentence. Then for each word or sequence of words in the source sentence for which we have an entry in the transducer new edges are generated and inserted into the lattice, spanning over the source phrase. One new edge is created for each translation candidate, and the translation score is assigned to this edge. The resulting lattice has then all the information available from the translation model.

The second step is search for a best path through this lattice, but not only based on the translation model scores but applying also the language model. We start with an initial special *sentence begin* hypothesis at the first node in the lattice. Hypotheses are then expanded over the edges, applying the language model to the partial translations attached to the edges. The following algorithm summarizes the decoding process when not considering word reordering:

```
Current node n, previous node n'; edge e
Language model state L, L'
Hypothesis h, h'
Foreach node n in the lattice
  Foreach incoming edge e in n
      phrase = word sequence at e
      n'     = FromNode(e)
      foreach L in n'
         foreach h with LMstate L
           LMcost = 0.0
           foreach word w in phrase
             LMcost += -log p(w|L)
             L' = NewState(L,w)
             L  = L'
           end
           Cost= LMcost+TMcost(e)
           TotalCost=TotalCost(h)+Cost
           h' = (L,e,h,TotalCost)
           store h'in Hypotheses(n,L)
```

The updated hypothesis `h'` at the current node stores the pointer to the previous hypothesis and the edge (labeled with the target phrase) over which it was expanded. Thus, at the final step, one can trace back to get the path associated with the minimum cost, i.e. the best hypothesis.

Other operators such as local word reordering are incorporated into this dynamic programming search (Vogel, 2003).

3 Phrase Pair Translation Probability

As stated in the previous section, one of the major problems is how to assign a reasonable probability for the extracted phrase pair to represent the translation quality.

Most of the phrase pairs are seen only once or twice in the training data. This is especially true for longer phrases. Therefore, phrase pair co-occurrence counts collected from the training corpus are not reliable and have little discriminative power. In (Vogel et al. 2003) a different estimation approach was proposed. Similar as in the IBM models, it is assumed that each source word s_i in the source phrase $\bar{s} = (s_1, s_2, \cdots s_I)$ is aligned to every target word t_j in the target phrase $\bar{t} = (t_1, t_2, \cdots t_J)$ with probability $\Pr(t_j \mid s_i)$. The total phrase translation probability is then calculated according to the following generative model:

$$\Pr(\bar{s} \mid \bar{t}) = \prod_{i=1}^{I} (\sum_{j=1}^{J} \Pr(s_i \mid t_j)) \qquad (1)$$

This is essentially the lexical probability as calculated in the IBM1 alignment model, without considering position alignment probabilities. Any statistical translation can be used in (1) to calculate the phrase translation probability. However, in our experiment we typically see now significant difference in translation results when using lexicons trained from different alignment models.

Also Equation (1) was confirmed to be robust and effective in parallel sentence mining from a very large and noisy comparable corpus (Zhao and Vogel, 2002).

Equation (1) does not explicitly discriminate content words from non-content words. As non-content words such as high frequency functional words tend to occur in nearly every parallel sentence pair, they co-occur with most of the source words in the vocabulary with non-trivial translation probabilities. This noise propagates via (1) into the phrase translations probabilities, increasing the chance that non-optimal phrase translation candidates get high probabilities and better translations are often not in the top ranks.

We propose a vector model to better distinguish between content words and non-content words with the goal to emphasize content words in the translation. This model will be used to *rescore* the phrase translation pairs, and to get a normalized score representing the translation probability.

4 Vector Model for Phrase Translation Probability

Term weighting models such as *tf.idf* are applied successfully in information retrieval. The duality of term frequency (*tf*) and inverse document frequency (*idf*), document space and collection space respectively, can smoothly predict the probability of terms being informative (Roelleke, 2003). Naturally, *tf.idf* is suitable to model content words as these words in general have large *tf.idf* weights.

4.1 Phrase Pair as Bag-of-Words

Our translation model: (*transducer*, as defined in 2.1), is a collection of phrase translation pairs together with scores representing the translation quality. Each phrase translation pair, which can be represented as a triple $\{\bar{s} \rightarrow \bar{t}, p\}$, is now converted into a "Bag-of-Words" D consisting of a collection of both source and target words appearing in the phrase pair, as shown in (2):

$$\{\bar{s} \rightarrow \bar{t}, p\} \Rightarrow D = \{s_1, s_2, \cdots s_I, t_1, t_2, \cdots t_J\} \qquad (2)$$

Given each phrase pair as one document, the whole transducer is a collection of such documents. We can calculate *tf.idf* for each s_i and t_j, and represent source and target phrases by vectors of $\vec{v}_s$ and $\vec{v}_t$ as in Equation (3):

$$\begin{aligned} \vec{v}_s &= \{w_{s_1}, w_{s_2}, \cdots, w_{s_I}\} \\ \vec{v}_t &= \{w_{t_1}, w_{t_2}, \cdots, w_{t_J}\} \end{aligned} \qquad (3)$$

where w_{s_i} and w_{t_j} are *tf.idf* for s_i or t_j respectively.

This vector representation can be justified by word co-occurrence considerations. As the phrase translation pairs are extracted from parallel sentences, the source words s_i and target words t_j in the source and target phrases must co-occur in the training data. The co-occurring words should share similar term frequency and document frequency statistics. Therefore, the vectors $\vec{v}_s$ and $\vec{v}_t$ have similar term weight contours corresponding to the co-occurring word pairs. So the vector representations of a phrase translation pair can reflect the translation quality. In addition, the content words and non-content words are modeled explicitly by using term weights. An over-simplified example would be that a rare word in the source language usually translates into a rare word in the target language.

4.2 Term Weighting Schemes

Given the transducer, it is straightforward to calculate term weights for source and target words. There are several versions of *tf.idf*. The smooth ones are preferred, because phrase translation pairs are rare events collected from training data.

The *idf* model selected is as in Equation (4):

$$idf = \log(\frac{N - df + 0.5}{df + 0.5}) \qquad (4)$$

where N is the total number of documents in the transducer, i.e. the total number of translation pairs, and *df* is the document frequency, i.e. in how many phrase pairs a given word occurs. The constant of 0.5 acts as smoothing.

Because most of the phrases are short, such as 2 to 8 words, the term frequency in the bag of words representation is usually 1, and some times 2. This, in general, does not bring much discrimination in representing translation quality. The following version of *tf* is chosen, so that longer target phrases with more words than average will be slightly down-weighted:

$$tf' = \frac{tf}{tf + 0.5 + 1.5 \cdot len(\vec{v}) / avglen(\vec{v})} \qquad (5)$$

where *tf* is the term frequency, $len(\vec{v})$ is the length in words of the phrase $\vec{v}$, and $avglen(\vec{v})$ is the average length of source or target phrase calculated from the transducer. Again, the values of 0.5 and 1.5 are constants used in IR tasks acting as smoothing.

Thus after a transducer is extracted from a parallel corpus, *tf* and *df* are counted from the collection of the "bag-of-words" phrase alignment representations. For each word in the phrase pair translation its *tf.idf* weight is assigned and the source and target phrase are transformed into vectors as shown in Equation (3). These vectors reserve the translation quality information and also model the content and non-content words by the term weighting model of *tf.idf*.

4.3 Vector Space Alignment

Given the vector representations in Equation (3), a similarity between the two vectors can not directly be calculated. The dimensions I and J are not guaranteed to be the same. The goal is to transform the source vector into a vector having the same dimensions as the target vector, i.e. to map the source vector into the space of the target vector, so that a similarity distance can be calculated. Using the same reasoning as used to motivate Equation (1), it is assumed that every source word s_i contributes some probability mass to each target word t_j. That is to say, given a term weight for t_j, all source term weights are aligned to it with some probability. So we can calculate a transformed vector from the source vectors by calculating weights $w_a^{t_j}$ using a translation lexicon $\Pr(t \mid s)$ as in Equation (6):

$$w_a^{t_j} = \sum_{i=1}^{I} \Pr(t_j \mid s_i) \cdot w_{s_i} \tag{6}$$

Now the target vector and the mapped vector $\vec{v}_a$ have the same dimensions as shown in (7):

$$\begin{aligned} \vec{v}_a &= \{w_a^{t_1}, w_a^{t_2}, \cdots, w_a^{t_J}\} \\ \vec{v}_t &= \{w_{t_1}, w_{t_2}, \cdots, w_{t_J}\} \end{aligned} \tag{7}$$

4.4 Similarity Functions

As explained in section 4.1, intuitively, if $\vec{s}$ and $\vec{t}$ is a good translation pair, then the corresponding vectors of $\vec{v}_a$ and $\vec{v}_t$ should be similar to each other in the vector space.

Cosine distance

The standard cosine distance is defined as the inner product of the two vectors $\vec{v}_a$ and $\vec{v}_t$ normalized by their norms. Based on Equation (6), it is easy to derive the similarity as follows:

$$\begin{aligned} d_{\cos}(v_a^t, v_t) &= \frac{(v_a^t, v_t)}{\|v_a^t\|\|v_t\|} = \frac{1}{\|v_a^t\|\|v_t\|} \sum_{j=1}^{J} w_a^{t_j} w_{t_j} \\ &= \frac{1}{\|v_a^t\|\|v_t\|} \sum_{j=1}^{J} \sum_{i=1}^{I} w_{t_j} P(t_j \mid s_i) w_{s_i} \\ &= \frac{\sum_{j=1}^{J} \sum_{i=1}^{I} w_{t_j} P(t_j \mid s_i) w_{s_i}}{sqrt(\sum_{j=1}^{J} w_{t_j}^2) sqrt(\sum_{j=1}^{J} {w_a^{t_j}}^2)} \end{aligned} \tag{8}$$

where I and J are the length of the source and target phrases; w_{s_i} and w_{t_j} are term weights for source word and target words; $w_a^{t_j}$ is the transformed weight mapped from all source words to the target dimension at word t_j.

BM25 distance

TREC tests show that *bm25* (Robertson and Walker, 1997) is one of the best-known distance schemes. This distance metric is given in Equation (9). The constants of k_1, b, k_3 are set to be 1, 1 and 1000 respectively.

$$\begin{aligned} d_{bm25} &= \sum_{j=1}^{J} w \frac{(k_1+1)w_{t_j}}{(K+w_{t_j})} \frac{(k_3+1)w_a^{t_j}}{(k_3+w_a^{t_j})} \\ w &= idf_{t_j} = (N - df_{t_j} + 0.5)/(df_{t_j} + 0.5) \\ K &= k_1((1-b) + J/avg(l)) \end{aligned} \tag{9}$$

where *avg(l)* is the average target phrase length in words given the same source phrase.

Our experiments confirmed the *bm25* distance is slightly better than the *cosine* distance, though the difference is not really significant. One advantage of *bm25* distance is that the set of free parameters k_1, b, k_3 can be tuned to get better performance e.g. via n-fold cross validation.

4.5 Integrated Translation Score

Our goal is to rescore the phrase translation pairs by using additional evidence of the translation quality in the vector space.

The vector based scores (8) & (9) provide a distinct view of the translation quality in the vector space. Equation (1) provides a evidence of the translation quality based on the word alignment probability, and can be assumed to be different from the evidences in vector space. Thus, a natural way of integrating them together is a *geometric interpolation* shown in (10) or equivalently a linear interpolation in the log domain.

$$d_{\text{int}} = d_{vec}^{\beta}(\vec{t}, \vec{s}) \cdot \Pr^{1-\beta}(\vec{s} \mid \vec{t}) \tag{10}$$

where $d_{vec}(\vec{t}, \vec{s})$ is the score from the *cosine* or *bm25* vector distance, normalized within [0, 1], like a probability.

$$\sum_{\vec{t}} d_{vec}(\vec{t}, \vec{s}) = 1.0$$

The parameter β can be tuned using held-out data. In our cross validation experiments $\beta = 0.5$ gave the best performance in most cases. Therefore, Equation (10) can be simplified into:

$$d_{\text{int}} = d_{vec}(\vec{t}, \vec{s}) \cdot \Pr(\vec{s} \mid \vec{t}) \tag{11}$$

The phrase translation score functions in (1) and (11) are non-symmetric. This is because the statistical lexicon *Pr(s|t)* is non-symmetric. One can easily re-write all the distances by using *Pr(t|s)*. But in our experiments this reverse direction of using *Pr(t|s)* gives trivially difference. So in all the experimental results reported in this paper, the distances defined in (1) and (11) are used.

5 Length Regularization

Phrase pair extraction does not work perfectly and sometimes a short source phrase is aligned to a long target phrase or vice versa. Length regularization can be applied to penalize too long or too short candidate translations. Similar to the sentence alignment work in (Gale and Church, 1991), the phrase length ratio is assumed to be a Gaussian distribution as given in Equation (12):

$$l(\bar{t},\bar{s}) \propto \exp(-0.5 \cdot \frac{(l(\bar{t})/l(\bar{s})-\mu)^2}{\sigma^2}) \qquad (12)$$

where *l(t)* is the target sentence length. Mean μ and variance σ can be estimated using a parallel corpus using a Maximum Likelihood criteria. The regularized score is the product of (11) and (12).

6 Experiments

Experiments were carried out on the so-called large data track Chinese-English TIDES translation task, using the June 2002 test data. The training data used to train the statistical lexicon and to extract the phrase translation pairs was selected from a 120 million word parallel corpus in such a way as to cover the phrases in test sentences. The restricted training corpus contained then approximately 10 million words.. A trigram model was built on 20 million words of general newswire text, using the SRILM toolkit (Stolcke, 2002). Decoding was carried out as described in section 2.2. The test data consists of 878 Chinese sentences or 24,337 words after word segmentation. There are four human translations per Chinese sentence as references. Both NIST score and Bleu score (in percentage) are reported for *adequacy* and *fluency* aspects of the translation quality.

6.1 Transducers

Four transducers were used in our experiments: LDC, BiBr, HMM, and ISA.

LDC was built from the LDC Chinese-English dictionary in two steps: first, morphological variations are created. For nouns and noun phrases plural forms and entries with definite and indefinite determiners were generated. For verbs additional word forms with -s -ed and -ing were generated, and the infinitive form with 'to'. Second, a large monolingual English corpus was used to filter out the new word forms. If they did not appear in the corpus, the new entries were not added to the transducer (Vogel, 2004).

BiBr extracts sub-tree mappings from Bilingual Bracketing alignments (Wu, 1997); HMM extracts partial path mappings from the Viterbi path in the Hidden Markov Model alignments (Vogel et. al., 1996). ISA is an integrated segmentation and alignment for phrases (Zhang et.al, 2003), which is an extension of (Marcu and Wong, 2002).

	LDC	BiBr	HMM	ISA
$N(K)$	425K	137K	349K	263K
$avg(l_{tgt}/l_{src})$	1.80	1.11	1.09	1.20

Table-1 statistics of transducers

Table-1 shows some statistics of the four transducers extracted for the translation task. N is the total number of phrase pairs in the transducer. LDC is the largest one having 425K entries, as the other transducers are restricted to 'useful' entries, i.e. those translation pairs where the source phrase matches a sequence of words in one of the test sentence. Notice that the LDC dictionary has a large number of long translations, leading to a high source to target length ratio.

6.2 *Cosine* vs *BM25*

The normalized *cosine* and *bm25* distances defined in (8) and (9) respectively, are plugged into (11) to calculate the translation probabilities. Initial experiments are reported on the LDC transducer, which gives already a good translation, and therefore allows for fast and yet meaningful experimentation.

Four baselines (Uniform, Base-m1, Base-m4, and Base-m4S) are presented in Table-2.

	NIST	Bleu
Uniform	*6.69*	*13.82*
Base-m1	*7.08*	*14.84*
Base-m4	*7.04*	*14.91*
Base-m4S	*6.91*	*14.44*
cosine	7.17	15.30
bm25	7.19	15.51
bm25-len	**7.21**	**15.64**

Table-2 Comparisons of different score functions

Decoder settings	without word reordering				with word reordering			
Scores (%)	**baseline**		**bm25**		**baseline**		**bm25**	
	NIST	Bleu	NIST	Bleu	NIST	Bleu	NIST	Bleu
LDC	*7.08*	*14.84*	7.21	15.64	*7.13*	*15.10*	7.26	15.98
LDC+ISA	*7.73*	*19.60*	7.99	19.58	*7.86*	*20.80*	8.13	20.93
LDC+ISA+HMM	*7.86*	*19.08*	8.14	20.70	*7.95*	*19.84*	8.19	21.60
LDC+ISA+HMM+BiBr	*7.87*	*19.23*	**8.14**	**21.48**	*7.99*	*20.01*	**8.24**	**22.37**

Table-3 Translation using bm25 rescore function with different decoder settings

In the first uniform probabilities are assigned to each phrase pair in the transducer. The second one (Base-m1) is using Equation (1) with a statistical lexicon trained using IBM Model-1, and Base-m4 is using the lexicon from IBM Model-4. Base-m4S is using IBM Model-4, but we skipped 194 high frequency English stop words in the calculation of Equation (1).

Table-2 shows that the translation score defined by Equation (1) is much better than a uniform model, as expected. *Base-m4* is slightly worse than *Base-m1*.on NIST score, but slightly better using the Bleu metric. Both differences are not statistically significant. The result for *Base-m4S* shows that skipping English stop words in Equation (1) gives a disadvantage. One reason is that skipping ignores too much non-trivial statistics from parallel corpus especially for short phrases. These high frequency words actually account already for more than 40% of the tokens in the corpus.

Using the vector model, both with the *cosine* $d_{\cos}$ and the *bm25* d_{bm25} distance, is significantly better than *Base-m1* and *Base-m4* models, which confirms our intuition of the vector model as an additional useful evidence for translation quality. The length regularization (12) helps only slightly for LDC. Since *bm25's* parameters could be tuned for potentially better performance, we selected *bm25* with length regularization as the model tested in further experiments.

A full-loaded system is tested using the LM020 *with* and *without word-reordering* in decoding. The results are presented in Table-3.

Table-3 shows consistent improvements on all configurations: the individual transducers, combinations of transducers, and different decoder settings of *word-reordering*. Because each phrase pair is treated as a "bag-of-words", the grammar structure is not well represented in the vector model. Thus our model is more tuned towards the *adequacy* aspect, corresponding to NIST score improvement.

Because the transducers of BiBr, HMM, and ISA are extracted from the same training data, they have significant overlaps with each other. This is why we observe only small improvements when adding more transducers.

The final NIST score of the full system is 8.24, and the Bleu score is 22.37. This corresponds to 3.1% and 11.8% relative improvements over the baseline. These improvements are statistically significant according to a previous study (Zhang et.al., 2004), which shows that a 2% improvement in NIST score and a 5% improvement in Bleu score is significant for our translation system on the June 2002 test data.

6.3 Mean Reciprocal Rank

To further investigate the effects of the rescoring function in (11), Mean Reciprocal Rank (MRR) experiments were carried out. MRR for a labeled set is the *mean* of the reciprocal rank of the individual phrase pair, at which the best candidate translation is found (Kantor and Voorhees, 1996).

Totally 9,641 phrase pairs were selected containing 216 distinct source phrases. Each source phrase was labeled with its best translation candidate without ambiguity. The rank of the labeled candidate is calculated according to translation scores. The results are shown in Table-4.

	baseline	cosine	bm25
MRR	0.40	0.58	0.75

Table-4 Mean Reciprocal Rank

The rescore functions improve the MRR from 0.40 to 0.58 using *cosine* distance, and to 0.75 using *bm25*. This confirms our intuitions that good translation candidates move up in the rank after the rescoring.

7 Conclusion and Discussion

In this work, we proposed a way of using term weight based models in a vector space as additional evidences for translation quality, and integrated the model into an existing phrase-based statistical machine translation system. The model shows significant improvements when using it to score a manual dictionary as well as when using different phrase transducers or a combination of all available translation information. Additional experiments also confirmed the effectiveness of the proposed model in terms of of improved Mean Reciprocal Rank of good translations.

Our future work is to explore alternatives such as the reranking work in (Collins, 2002) and include more knowledge such as syntax information in rescoring the phrase translation pairs.

References

A. Stolcke. 2002. SRILM -- An Extensible Language Modeling Toolkit. *In 2002 Proc. Intl. Conf. on Spoken Language Processing*, Denver.

Peter F. Brown, Stephen A. Della Pietra, Vincent J.Della Pietra, and Robert L. Mercer. 1993. *The Mathematics of Statistical Machine Translation: Parameter Estimation*, Computational Linguistics, vol. 19, no. 2, pp. 263–311.

Michael Collins. 2000. Discriminative Reranking for Natural Language Parsing. *Proc. 17th International Conf. on Machine Learning*. pp. 175-182.

William A. Gale and Kenneth W. Church. 1991. A Program for Aligning Sentences in Bilingual Corpora. In *Computational Linguistics*, vol.19 pp. 75-102.

Paul B. Kantor, Ellen Voorhees. 1996. Report on the TREC-5 Confusion Track. *The Fifth Text Retrieval Conference*.

Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical Phrase-Based Translation. *Proceedings of HLT-NAACL*. Edmonton, Canada.

Daniel Marcu and William Wong. 2002. A Phrase-Based, Joint Probability Model for Statistical Machine Translation. *Proceedings of EMNLP-2002*, Philadelphia, PA.

Franz Josef Och and Hermann Ney. 2000. Improved Statistical Alignment Models. *Proceedings of ACL-00*, pp. 440-447, Hongkong, China.

Thomas Roelleke. 2003. A Frequency-based and a Poisson-based Definition of the Probability of Being Informative. *Proceedings of the 26th annual international ACM SIGIR*. pp. 227-234.

S.E. Robertson, and S. Walker. 1997. On relevance weights with little relevance information. *In 1997 Proceedings of ACM SIGIR*. pp. 16-24.

Dekai Wu. 1997. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora, *Computational Linguistics* 23(3): pp.377-404.

K. Yamada and K. Knight. 2001. A Syntax-Based Statistical Translation Model. *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics*. pp. 523-529. Toulouse, France.

Richard Zens, Franz Josef Och and Hermann Ney. 2002. Phrase-Based Statistical Machine Translation. *Proceedings of the 25th Annual German Conference on AI: Advances in Artificial Intelligence*. pp. 18-22.

Stephan Vogel, Hermann Ney, Christian Tillmann. 1996. HMM-based word alignment in statistical translation. In: COLING '96: The 16th Int. Conf. on Computational Linguistics, Copenhagen, Denmark (1996) pp. 836-841.

Stephan Vogel, Ying Zhang, Fei Huang, Alicia Tribble, Ashish Venogupal, Bing Zhao, Alex Waibel. 2003. The CMU Statistical Translation System, *Proceedings of MT-Summit IX*. New Orleans, LA.

Stephan Vogel. 2003. SMT decoder dissected: word reordering, *In 2003 Proceedings of Natural Language Processing and Knowledge Engineering*, (NLP-KE'03) Beijing, China.

Stephan Vogel. 2004. Augmenting Manual Dictionaries for Statistical Machine Translation Systems, *In 2003 Proceedings of LREC*, Lisbon, Portugal. pp. 1593-1596.

Ying Zhang, Stephan Vogel, Alex Waibel. 2004. Interpreting BLEU/NIST Scores : How much Improvement Do We Need to Have a Better System? *In 2004 Proceedings of LREC*, Lisbon, Portugal. pp. 2051-2054.

Ying Zhang, Stephan Vogel, Alex Waibel. 2003. "Integrated Phrase Segmentation and Alignment Algorithm for Statistical Machine Translation," in the Proceedings of NLP-KE'03, Beijing, China.

Bing Zhao, Stephan Vogel. 2002. Adaptative Parallel Sentences Mining from web bilingual news collection. In *2002 IEEE International Conference on Data Mining*, Maebashi City, Japan.