Query Lattice for Translation Retrieval

Meiping Dong[†], Yong Cheng[‡], Yang Liu[†], Jia Xu[‡], Maosong Sun[†], Tatsuya Izuha[◇], Jie Hao[#]

[†]State Key Laboratory of Intelligent Technology and Systems
Tsinghua National Laboratory for Information Science and Technology
Department of Computer Sci. and Tech., Tsinghua University, Beijing, China
hellodmp@163.com, {liuyang2011,sms}@tsinghua.edu.cn

[‡]Institute for Interdisciplinary Information Sciences
Tsinghua University, Beijing, China
chengyong3001@gmail.com, xu@tsinghua.edu.cn

[◇]Toshiba Corporation Corporate Research & Development Center
tatsuya.izuha@toshiba.co.jp

[#]Toshiba (China) R&D Center
haojie@toshiba.com.cn

Abstract

Translation retrieval aims to find the most likely translation among a set of target-language strings for a given source-language string. Previous studies consider the single-best translation as a query for information retrieval, which may result in translation error propagation. To alleviate this problem, we propose to use the *query lattice*, which is a compact representation of exponentially many queries containing translation alternatives. We verified the effectiveness of query lattice through experiments, where our method explores a much larger search space (from 1 query to 1.24×10^{62} queries), runs much faster (from 0.75 to 0.13 second per sentence), and retrieves more accurately (from 83.76% to 93.16% in precision) than the standard method based on the query single-best. In addition, we show that query lattice significantly outperforms the method of (Munteanu and Marcu, 2005) on the task of parallel sentence mining from comparable corpora.

1 Introduction

Translation retrieval aims to search for the most probable translation candidate from a set of target-language strings for a given source-language string. Early translation retrieval methods were widely used in example-based and memory-based translation systems (Sato and Nagao, 1990; Nirenburg et al., 1993; Baldwin and Tanaka, 2000; Baldwin, 2001). Often, the document set is a list of translation records that are pairs of source-language and target-language strings. Given an input source string, the retrieval system returns a translation record of maximum similarity to the input on the source side. Although these methods prove to be effective in example-based and memory-based translation systems, they heavily rely on parallel corpora that are limited both in size and domain.

More recently, Liu et al. (2012) have proposed a new translation retrieval architecture that depends only on monolingual corpora. Given an input source string, their system retrieves translation candidates from a set of target-language sentences. This can be done by combining machine translation (MT) and information retrieval (IR): machine translation is used to transform the input source string to a coarse translation, which serves as a query to retrieve the most probable translation in the monolingual corpus. Therefore, it is possible for translation retrieval to have access to a huge volume of monolingual corpora that are readily available on the Web.

However, the MT + IR pipeline suffers from the *translation error propagation problem*. Liu et al. (2012) use 1-best translations, which are inevitably erroneous due to the ambiguity and structural divergence of natural languages, as queries to the IR module. As a result, translation mistakes will be

Corresponding author: Jia Xu. Tel: +86-10-62781693 Ext 1683. Homepage: iiis.tsinghua.edu.cn/∼xu

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 2031–2041, Dublin, Ireland, August 23-29 2014.

propagated to the retrieval process. This situation aggravates when high-accuracy MT systems are not available for resource-scarce languages.

In this work, we propose to use *query lattice* in translation retrieval to alleviate the translation error propagation problem. A query lattice is a compact representation of exponentially many queries. We design a retrieval algorithm that takes the query lattice as input to search for the most probable translation candidate from a set of target-language sentences. As compared with Liu et al. (2012), our approach explores a much larger search space (from 1 query to 1.24×10^{62} queries), runs much faster (from 0.75 second per sentence to 0.13), and retrieves more accurately (from 83.76% to 93.16%). We also evaluate our approach on extracting parallel sentences from comparable corpora. Experiments show that our translation retrieval system significantly outperforms a state-of-the-art parallel corpus mining system.

2 Related Work

Our work is inspired by three research topics: retrieving translation candidates from parallel corpus, using lattice to compactly represent exponentially many alternatives, and using lattice as query in information retrieval.

1. *Translation Retrieval using Parallel Corpus*. The idea of retrieving translation candidates from existing texts originated in example-based and memory-based translation (Sato and Nagao, 1990; Nirenburg et al., 1993; Baldwin and Tanaka, 2000; Baldwin, 2001). As these early efforts use a parallel corpus (e.g., translation records that are pairs of source-language and target-language strings), they focus on calculating the similarity between two source-language strings. In contrast, we evaluate the translational equivalence of a given source string and a target string in a large monolingual corpus.

2. *Lattice in Machine Translation*. Lattices have been widely used in machine translation: considering Chinese word segmentation alternatives (Xu et al., 2005), speech recognition candidates (Matsoukas et al., 2007), SCFG (Dyer et al., 2008) and so on in the decoding process, minimum bayes risk decoding (Tromble et al., 2008), minimum error rate training (Macherey et al., 2008), system combination (Feng et al., 2009), just to name a few. In this work, we are interested in how to use a lattice that encodes exponentially many translation candidates as a single query to retrieve similar target sentences via an information retrieval system.

3. *Query Lattice in Information Retrieval*. The use of lattices in information retrieval dates back to Moore (1958). Most current lattice-based IR systems often treat lattices as conceptional hierarchies or thesauri in formal concept analysis (Priss, 2000; Cheung and Vogel, 2005). In spoken document retrieval, however, lattices are used as a compact representation of multiple speech recognition transcripts to estimate the expected counts of words in each document (Saraclar and Sproat, 2004; Zhou et al., 2006; Chia et al., 2010). Our work is significantly different from previous work that uses the bag-of-words model because translation retrieval must take structure and dependencies in text into account to ensure translational equivalence.

3 Query Lattice for Translation Retrieval

3.1 Translation Retrieval

Let $\mathbf{f}$ be a source-language string, $\mathbf{E}$ be a set of target-language strings, the problem is how to find the most probable translation $\hat{\mathbf{e}}$ from $\mathbf{E}$. Note that $\mathbf{E}$ is a monolingual corpus rather than a parallel corpus. Therefore, string matching on the source side (Sato and Nagao, 1990; Nirenburg et al., 1993; Baldwin and Tanaka, 2000; Baldwin, 2001) does not apply here.

We use $P(\mathbf{e}|\mathbf{f})$ to denote the probability that a target-language sentence $\mathbf{e}$ is the translation of a source-language sentence $\mathbf{f}$. As suggested by Liu et al. (2012), it can be decomposed into two sub-models by

introducing a coarse translation $\mathbf{q}$ as a hidden variable:

$$P(\mathbf{e}|\mathbf{f}) = \sum_{\mathbf{q}\in\mathbf{Q}(\mathbf{f})} P(\mathbf{q},\mathbf{e}|\mathbf{f}) \quad (1)$$

$$= \sum_{\mathbf{q}\in\mathbf{Q}(\mathbf{f})} P(\mathbf{q}|\mathbf{f}) \times P(\mathbf{e}|\mathbf{q},\mathbf{f}) \quad (2)$$

where $P(\mathbf{q}|\mathbf{f})$ is a **translation** sub-model, $P(\mathbf{e}|\mathbf{q},\mathbf{f})$ is a **retrieval** sub-model, and $\mathbf{Q}(\mathbf{f})$ is the set of all possible translations of the sentence $\mathbf{f}$. Note that $\mathbf{q}$ actually serves as a **query** to the retrieval sub-model.

To take advantage of various translation and retrieval information sources, we use a log-linear model (Och and Ney, 2002) to define the conditional probability of a query $\mathbf{q}$ and a target sentence $\mathbf{e}$ conditioned on a source sentence $\mathbf{f}$ parameterized by a real-valued vector $\boldsymbol{\theta}$:

$$P(\mathbf{q},\mathbf{e}|\mathbf{f};\boldsymbol{\theta}) = \frac{\exp(\boldsymbol{\theta}\cdot\mathbf{h}(\mathbf{q},\mathbf{e},\mathbf{f}))}{\sum_{\mathbf{q}'\in\mathbf{Q}(\mathbf{f})}\sum_{\mathbf{e}'\in\mathbf{E}}\exp(\boldsymbol{\theta}\cdot\mathbf{h}(\mathbf{q}',\mathbf{e}',\mathbf{f}))} \quad (3)$$

where $\mathbf{h}(\cdot)$ is a vector of feature functions and $\boldsymbol{\theta}$ is the corresponding feature weight vector.

Accordingly, the decision rule for the latent variable model is given by

$$\hat{\mathbf{e}} = \arg\max_{\mathbf{e}\in\mathbf{E}} \left\{ \sum_{\mathbf{q}\in\mathbf{Q}(\mathbf{f})} \exp(\boldsymbol{\theta}\cdot\mathbf{h}(\mathbf{q},\mathbf{e},\mathbf{f})) \right\} \quad (4)$$

As there are exponentially many queries, it is efficient to approximate the summation over all possible queries by using maximization instead:

$$\hat{\mathbf{e}} \approx \arg\max_{\mathbf{e}\in\mathbf{E}} \left\{ \max_{\mathbf{q}\in\mathbf{Q}(\mathbf{f})} \left\{ \boldsymbol{\theta}\cdot\mathbf{h}(\mathbf{q},\mathbf{e},\mathbf{f}) \right\} \right\} \quad (5)$$

Unfortunately, the search space is still prohibitively large since we need to enumerate all possible queries. Liu et al. (2012) split Eq. (5) into two steps. In the first step, a translation module runs to produce the 1-best translation $\hat{\mathbf{q}}$ of the input string $\mathbf{f}$ as a query:

$$\hat{\mathbf{q}} \approx \arg\max_{\mathbf{q}\in\mathbf{Q}(\mathbf{f})} \left\{ \boldsymbol{\theta}_t\cdot\mathbf{h}_t(\mathbf{q},\mathbf{e},\mathbf{f}) \right\} \quad (6)$$

where $\mathbf{h}_t(\cdot)$ is a vector of translation features and $\boldsymbol{\theta}_t$ is the corresponding feature weight vector. In the second step, a monolingual retrieval module takes the 1-best translation $\hat{\mathbf{q}}$ as a query to search for the target string $\hat{\mathbf{e}}$ with the highest score:

$$\hat{\mathbf{e}} \approx \arg\max_{\mathbf{e}\in\mathbf{E}} \left\{ \boldsymbol{\theta}_r\cdot\mathbf{h}_r(\hat{\mathbf{q}},\mathbf{e},\mathbf{f}) \right\} \quad (7)$$

where $\mathbf{h}_r(\cdot)$ is a vector of retrieval features and $\boldsymbol{\theta}_r$ is the corresponding feature weight vector.

Due to the ambiguity of translation, however, state-of-the-art MT systems are still far from producing high-quality translations, especially for distantly-related languages. As a result, the 1-best translations are usually erroneous and potentially introduce retrieval mistakes.

A natural solution is to use n-best lists as queries:

$$\hat{\mathbf{e}} \approx \arg\max_{\mathbf{e}\in\mathbf{E}} \left\{ \max_{\mathbf{q}\in\mathbf{N}(\mathbf{f})} \left\{ \boldsymbol{\theta}\cdot\mathbf{h}(\mathbf{q},\mathbf{e},\mathbf{f}) \right\} \right\} \quad (8)$$

where $\mathbf{N}(\mathbf{f}) \subset \mathbf{T}(\mathbf{f})$ is the n-best translations of the input source sentence $\mathbf{f}$.

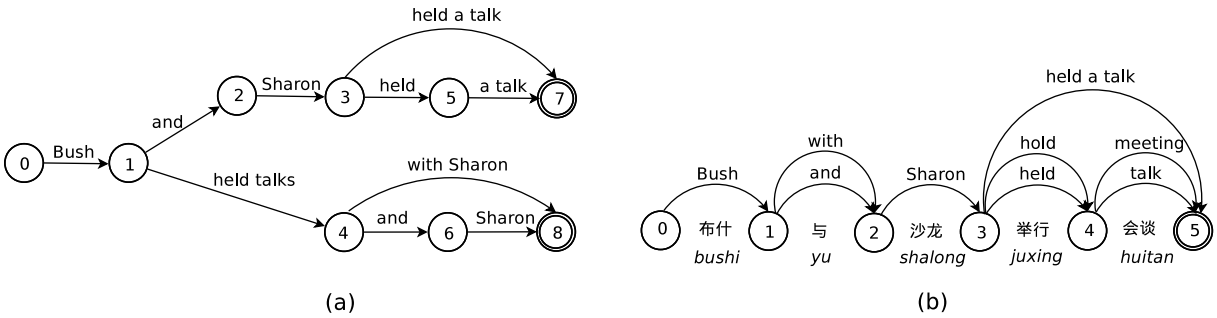

Figure 1: Two kinds of query lattices: (a) search graph that is generated *after* phrase-based decoding and (b) translation option graph that is generated *before* decoding. Translation option graph is more compact and encodes more translation candidates.

Although using n-best lists apparently improves the retrieval accuracy over using 1-best lists, there are two disadvantages. First, the decision rule in Eq. (8) requires to enumerate all the n translations and retrieve for n times. In other words, the time complexity increases linearly. Second, an n-best list only accounts for a tiny fraction of the exponential search space of translation. To make things worse, there are usually very few variations in n-best translations because of spurious ambiguity - a situation where multiple derivations give similar or even identical translations.

Therefore, we need to find a more elegant way to enable the retrieval module to explore exponentially many queries without sacrificing efficiency.

3.2 Query Lattice

We propose to use **query lattice** to compactly represent exponentially many queries. For example, given a source sentence "*bushi yu shalong juxing huitan*", we can use the **search graph** produced by a phrase-based translation system (Koehn et al., 2007) as a lattice to encode exponentially many derivations.

Figure 1(a) shows a search graph for the example source sentence. Each edge is labeled with an English phrase as well as the corresponding translation feature value vector. Node 0 denotes the starting node. Node 7 and node 8 are two ending nodes. Each path from the starting node to an ending node denotes a query. Paths that reach the same node in the lattice correspond to recombined hypotheses that have equivalent feature histories (e.g., coverage, last generated target words, the end of last covered source phrase, etc) in phrase-based decoding.

However, there are two problems with using search graph as query lattice. First, it is computationally expensive to run a phrase-based system to generate search graphs. The time complexity for phrase-based decoding with beam search is $O(n^2b)$ (Koehn et al., 2007), where n is the length of source string and b is the beam width. Moreover, the memory requirement is usually very high due to language models. As a result, translation is often two orders of magnitude slower than retrieval. Second, a search graph has too many "duplicate" edges due to different reordering, which increase the time complexity of retrieval (see Section 3.3). For example, in Figure 1(a), the English phrase "Sharon" occurs two times due to different reordering.

Alternatively, we propose to use **translation option graph** as query lattice. In a phrase-based translation system, translation options that are phrase pairs matching a substring in the input source string are collected *before* decoding. These translation options form a query lattice with monotonic reordering. Figure 1(b) shows an example translation option graph, in which nodes are sorted according to the positions of source words. Each edge is labeled with an English phrase as well as the corresponding translation feature value vector.

We believe that translation option graph has three advantages over search graph:

1. *Improved efficiency in translation*. Translation option graph requires no decoding.

2. *Improved efficiency in retrieval*. Translation option graph has no duplicate edges.

Algorithm 1 Retrieval with lattice as query.

procedure LATTICERETRIEVE($\mathbf{L}(\mathbf{f}), \mathbf{E}, k$)
  $Q \leftarrow$ GETWORDS($\mathbf{L}(\mathbf{f})$) ▷ Get distinct words in the lattice to form a coarse query
  $\mathbf{E}_k \leftarrow$ RETRIEVE($\mathbf{E}, Q, k$) ▷ Retrieve top-k target sentences using the coarse query
  for all $\mathbf{e} \in \mathbf{E}_k$ **do**
    FINDPATH($\mathbf{L}(\mathbf{f}), \mathbf{e}$) ▷ Find a path with the highest score
  end for
  SORT($\mathbf{E}_k$) ▷ Sort retrieved sentences according the scores
  return $\mathbf{E}_k$
end procedure

Algorithm 2 Find a path with the highest score.

procedure FINDPATH($\mathbf{L}(\mathbf{f}), \mathbf{e}$)
  for $v \in \mathbf{L}(\mathbf{f})$ in topological order **do**
    $path(v) \leftarrow \emptyset$ ▷ Initialize the Viterbi path at node v
    $score(v) \leftarrow 0$ ▷ Initialize the Viterbi score at node v
    for $u \in IN(v)$ **do** ▷ Enumerate all antecedents
      $p \leftarrow path(u) \cup \{e_{u \rightarrow v}\}$ ▷ Generate a new path
      $s \leftarrow score(u) +$ COMPUTESCORE($e_{u \rightarrow v}$) ▷ Compute the path score
      if $s > score(v)$ **then**
        $path(v) \leftarrow p$ ▷ Update the Viterbi path
        $score(v) \leftarrow s$ ▷ Update the Viterbi score
      end if
    end for
  end for
end procedure

3. *Enlarged search space*. Translation option graph represents the entire search space of monotonic decoding while search graph prunes many translation candidates.

In Figure 1, the search graph has 9 nodes, 10 edges, 4 paths, and 3 distinct translations. In contrast, the translation option graph has 6 nodes, 9 edges, 10 paths, and 10 distinct translations. Therefore, translation option graph is more compact and encodes more translation candidates.

Although translation option graph ignores language model and lexcialized reordering models, which prove to be critical information sources in machine translation, we find that it achieves comparable or even better retrieval accuracy than search graph (Section 4). This confirms the finding of Liu et al. (2012) that language model and lexicalized reordering models only have modest effects on translation retrieval.

3.3 Retrieval with Query Lattice

Given a target corpus $\mathbf{E}$ and a query lattice $\mathbf{L}(\mathbf{f}) \subset \mathbf{Q}(\mathbf{f})$, our goal is to find the target sentence $\hat{\mathbf{e}}$ with the highest score $\boldsymbol{\theta} \cdot \mathbf{h}(\mathbf{q}, \mathbf{e}, \mathbf{f})$:

$$\hat{\mathbf{e}} \approx \arg\max_{\mathbf{e} \in \mathbf{E}} \left\{ \max_{\mathbf{q} \in \mathbf{L}(\mathbf{f})} \left\{ \boldsymbol{\theta} \cdot \mathbf{h}(\mathbf{q}, \mathbf{e}, \mathbf{f}) \right\} \right\} \quad (9)$$

Due to the exponentially large search space, we use a coarse-to-fine algorithm to search for the target sentence with the highest score, as shown in Algorithm 1. We use an example to illustrate the basic idea. Given an input source sentence "*bushi yu shalong juxing le huitan*", our system first generates a query lattice like Figure 1(a). It is non-trivial to directly feed the query lattice to a retrieval system. Instead, we would like to first collect all distinct words in the lattice: {"*Bush*", "*and*" , "*Sharon*", "*held*", "*a*", "*talk*", "*talks*", "*with*"}. This set serves as a coarse single query and the retrieval system returns a list of target sentences that contain these words:

			Chinese	English
Training			1.21M	1.21M
Dev	in-domain	query	5K	
		document		2.23M
	out-of-domain	query	5K	
		document		2.23M
Test	in-domain	query	5K	
		document		2.23M
	out-of-domain	query	5K	
		document		2.23M

Table 1: The datasets for the retrieval evaluation. The training set is used to train the phrase-based translation model and language model for Moses (Koehn et al., 2007). The development set is used to optimize feature weights using the minimum-error-rate algorithm (Och, 2003). A development set consists of a query set and a document set. The test set is used to evaluate the retrieval accuracy. To examine the effect of domains on retrieval performance, we used two development and test sets: in-domain and out-domain.

President Bush gave a talk at a meeting
Bush held a meeting with Sharon
Sharon and Bush attended a meeting held at London

Note that as a retrieval system usually ignores the structural dependencies in text, the retrieved sentences (scored by retrieval features) are relevant but not necessarily translations of the input. Therefore, we can match each retrieved sentence against the query lattice to find a path with the highest score using additional translation features. For example, the Viterbi path for "*Bush held a meeting with Sharon*" in Figure 1(a) is "*Bush held talks with Sharon*". The translation features of matched arcs in the path are collected to compute the overall score according to Eq. (9). Finally, the algorithm returns a sorted list:

Bush held a meeting with Sharon
President Bush gave a talk at a meeting
Sharon and Bush attended a meeting held at London

More formally, the input of Algorithm 1 are a query lattice $\mathbf{L}(\mathbf{f})$, a target corpus $\mathbf{E}$, and a parameter k (line 1). The function GETWORDS simply collects all the distinct words appearing in the lattice (line 2), which are used for constructing a coarse boolean query Q. Then, the function RETRIEVE runs to retrieve the top-k target sentences $\mathbf{E}_k$ in the target corpus $\mathbf{E}$ only using standard IR features according to the query Q (line 3). These first two steps eliminate most unlikely candidates and return a coarse set of target sentence candidates efficiently. [1] Then, a procedure FINDPATH$(\mathbf{L}(\mathbf{f}), \mathbf{e})$ runs to search for the translation with the highest score for each candidate (lines 4-6). Finally, the algorithm returns the sorted list of target sentences (lines 7-9).

Algorithm 2 shows the procedure FINDPATH$(\mathbf{L}(\mathbf{f}), \mathbf{e})$, which searches for the path with higher score using a Viterbi-style algorithm. The function COMPUTESCORE scores an edge according to the Eq. (9) which linearly combines the translation and retrieval features.

Generally, the lattice-based retrieval algorithm has a time complexity of $O(k|E|)$, where $|E|$ is the number of edges in the lattice.

4 Experiments

In this section, we try to answer two questions:

1. Does using query lattices improve translation retrieval accuracy over using n-best lists?

2. How does translation retrieval benefit other end-to-end NLP tasks such as machine translation?

[1] In our experiments, we set the parameter k to 500 as a larger value of k does not give significant improvements but introduce more noises.

Accordingly, we evaluated our system in two tasks: translation retrieval (Section 4.1) and parallel corpus mining (Section 4.2).

4.1 Evaluation on Translation Retrieval

4.1.1 Experimental Setup

In this section, we evaluate the accuracy of translation retrieval: given a query set (i.e., source sentences), our system returns a sorted list of target sentences. The evaluation metrics include precision@n and recall.

The datasets for the retrieval evaluation are summarized in Table 1. The **training set**, which is used to train the phrase-based translation model and language model for the-state-of-the-art phrase-based system Moses (Koehn et al., 2007), contains 1.21M Chinese-English sentences with 32.0M Chinese words and 35.2M English words. We used the SRILM toolkit (Stolcke, 2002) to train a 4-gram language model on the English side of the training corpus. The **development set**, which is used to optimize feature weights using the minimum-error-rate algorithm (Och, 2003), consists of **query set** and a **document set**. We sampled 5K parallel sentences randomly, in which 5K Chinese sentences are used as queries and half of their parallelled English sentences(2.5K) mixed with other English sentences(2.3M) as the retrieval document set. As a result, we can compute precision and recall in a noisy setting. The **test set** is used to compute retrieval evaluation metrics. To examine the effect of domains on retrieval performance, we used two data sets: **in-domain** and **out-domain**. The in-domain development and test sets are close to the training set while the out-domain data sets are not.

We compare three variants of translation retrieval: 1-best list, n-best list, and lattice. For query lattice, we further distinguish between search graph and translation option graph. They are generated by Moses with the default setting.

We use both translation and retrieval features in the experiments. The translation features include phrase translation probabilities, phrase penalty, distance-based and lexicalized reordering models, language models, and word penalty. Besides the conventional IR features such as term frequency and inverse document frequency, we use five additional featured derived from BLEU (Papineni et al., 2002): the n-gram matching precisions between query and retrieved target sentence ($n = 1, 2, 3, 4$) and brevity penalty. These features impose structural constraints on retrieval and ensure translation closeness of retrieved target sentences. The minimum-error-rate algorithm supports a variety of loss functions. The loss function we used in our experiment is $1-$P@n. Note that using translation option graph as query lattice does not include language models and distance-based lexicalized reordering models as features.

4.1.2 Evaluation Results

Table 2 shows the results on the in-domain test set. The "# candidates" column gives the number of translation candidates explored by the retrieval module for each source sentence on average. The lattices, either generated by search graph or by translation options, contain exponentially many candidates. We find that using lattices dramatically improves the precisions over using 1-best and n-best lists. All the improvements over 1-best and n-best lists are significant statistically. The 1-best, n-best, and the search graph lattice share with the same translation time: 5,640 seconds for translating 5,000 queries. Note that the translation time is zero for the translation option graph because it does not need phrase-based decoding. For retrieval, the time cost for the n-best list method generally increases linearly. As the search graph lattice contains many edges, the retrieval time increases by an order of magnitude as compared with 100-best list. An interesting finding is that using translation options as a lattice contains more candidates and consumes much less time for retrieval than using search graph as a lattice. One possible reason is that a search graph generated by Moses usually contains many redundant edges. For example, Figure 1 is actually a search graph and many phrases occurs multiple times in the lattice (e.g., "and" and "Sharon"). In contrast, a lattice built by translation options hardly has any redundant edges but still represents exponentially many possible translations. We can also see that the lattice constructed by search graph considering language model can benefit the precision much, especially when n is little. But this advantage decreases with n increasing and the time consumed by translation options as lattice is much less than the search graph as lattice. Besides, the margin between them is not too large so we can

method	# candidates	P@n					time	
		n=1	n=5	n=10	n=20	n=100	translation	retrieval
1-best	1	87.40	91.40	92.24	92.88	93.64	5,640	82
10-best	10	89.84	93.20	93.96	94.36	95.56	5,640	757
100-best	100	90.76	94.32	95.00	95.76	96.76	5,640	7,421
lattice (graph)	1.20×10^{54}	93.60	96.08	96.28	96.52	96.80	5,640	89,795
lattice (options)	4.14×10^{62}	93.28	95.84	95.96	96.16	96.84	0	307

Table 2: Results on the in-domain test set. We use the minimum-error-rate training algorithm (Och, 2003) to optimize the feature with the respect to $1-$P@n.

method	# candidates	P@n					time	
		n=1	n=5	n=10	n=20	n=100	translation	retrieval
1-best	1	67.32	76.60	79.40	81.80	83.76	3,660	92
10-best	10	72.68	80.96	83.36	85.84	88.76	3,660	863
100-best	100	78.60	85.76	87.76	89.64	92.16	3,660	8,418
lattice (graph)	1.51×10^{61}	84.32	89.40	90.68	91.56	92.44	3,660	67,205
lattice (options)	1.24×10^{65}	81.92	88.00	89.80	91.24	93.16	0	645

Table 3: Results on the out-of-domain test set.

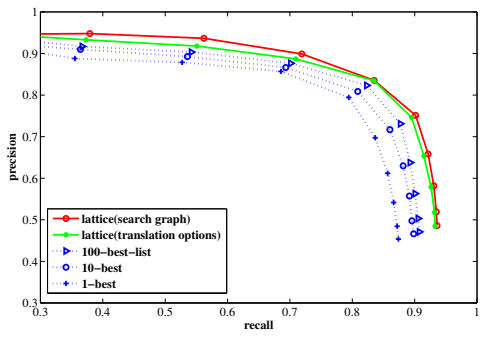

Figure 2: In-domain Precision-Recall curves.

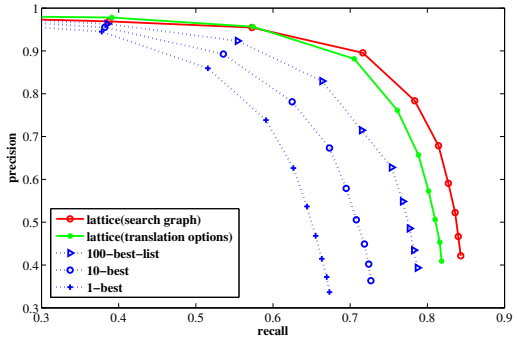

Figure 3: Out-domain Precision-Recall curves.

abandon some little precision for obtain the large time reducing. Therefore, using translation options as lattices seems to be both effective and efficient.

Table 3 shows the results on the out-of-domain test set. While the precisions for all methods drop, the margins between lattice-based retrieval and n-best list retrieval increase, suggesting that lattice-based methods are more robust when dealing with noisy datasets.

Figures 2 and 3 show the Precision-Recall curves on the in-domain and out-of-domain test sets. As the query set is derived from parallel sentences, recall can be computed in our experiments. The curves show that using lattices clearly outperforms using 1-best and n-best lists. The margins are larger on the out-of-domain test set.

4.2 Evaluation on Parallel Corpus Mining

In this section, we evaluate translation retrieval on the parallel corpus mining task: extracting a parallel corpus from a comparable corpus.

4.2.1 Experimental Setup

The comparable corpus for extracting parallel sentences contains news articles published by Xinhua News Agency from 1995 to 2010. Table 4 shows the detailed statistics. There are 1.2M Chinese and 1.7M English articles.

We re-implemented the method as described in (Munteanu and Marcu, 2005) as the baseline system.

language	articles	sentences	words	vocabulary
Chinese	1.2M	18.5M	441.2M	2.1M
English	1.7M	17.8M	440.2M	3.4M

Table 4: The Xinhua News Comparable Corpus from 1995 to 2010

Munteanu and Marcu (2005)			*this work*		
English words	Chinese words	BLEU	English words	Chinese Words	BLEU
5.00M	4.12M	22.84	5.00M	3.98M	25.44
10.00M	8.20M	25.10	10.00M	8.17M	26.62
15.00M	12.26M	25.41	15.00M	12.49M	26.49
20.00M	16.30M	25.56	20.00M	16.90M	26.87

Table 5: Comparison of BLEU scores using parallel corpora extracted by the baseline and our system. Given a comparable corpus (see Table 4), both systems extract parallel corpora that are used for training phrase-base models (Koehn et al., 2007). The baseline system is a re-implementation of the method described in (Munteanu and Marcu, 2005). Our system uses translation option graph as query lattice. Our system significantly outperforms the baseline for various sizes.

It assigned a score to each sentence pair using a classifier. Our system used translation option graph as query lattices due to its simplicity and effectiveness. For each source sentence in the comparable corpus, our system retrieved the top target sentence together with a score.

To evaluate the quality of extracted parallel corpus, we trained phrase-based models on it and ran Moses on NIST datasets. The development set is the NIST 2005 test set and the test set is the NIST 2006 test set. The final evaluation metric is case-insensitive BLEU-4.

4.2.2 Evaluation Results

Table 5 shows the comparison of BLEU scores using parallel corpora extracted by the baseline and our system. We find that our system significantly outperforms the baseline for various parallel corpus sizes. This finding suggests that using lattice to compactly represent exponentially many alternatives does help to alleviate the translation error propagation problem and identify parallel sentences of high translational equivalence.

5 Conclusion

In this work, we propose to use query lattice to address the translation error propagation problem in translation retrieval. Two kinds of query lattices are used in our experiments: search graph and translation option graph. We show that translation option graph is more compact and represents a much larger search space. Our experiments on Chinese-English datasets show that using query lattices significantly outperforms using n-best lists in the retrieval task. Moreover, we show that translation retrieval is capable of extracting high-quality parallel corpora from a comparable corpus. In the future, we plan to apply our approach to retrieving translation candidates directly from the Web, which can be seen as a huge monolingual corpus.

Acknowledgments

This research is supported by the 973 Program (No. 2014CB340501), the National Natural Science Foundation of China (No. 61331013 and No. 61033001), the 863 Program (No. 2012AA011102), Toshiba Corporation Corporate Research & Development Center, and the Singapore National Research Foundation under its International Research Centre @ Singapore Funding Initiative and administered by the IDM Programme.

References

T. Baldwin and H. Tanaka. 2000. The effects of word order and segmentation on translation retrieval performance. In *Proceedings of COLING*.

Timothy Baldwin. 2001. Low-cost, high-performance translation retrieval: Dumber is better. In *Proceedings of ACL*, pages 18–25, Toulouse, France, July. Association for Computational Linguistics.

Karen Cheung and Douglas Vogel. 2005. Complexity reduction in lattice-based information retrieval. *Information Retrieval*, pages 285–299.

Tee Kiah Chia, Khe Chai Sim, Haizhou Li, and Hwee Tou Ng. 2010. Statistical lattice-based spoken document retrieval. *ACM Transactions on Information Systems*, 28(1).

Christopher Dyer, Smaranda Muresan, and Philip Resnik. 2008. Generalizing word lattice translation. In *Proceedings of ACL-HLT*, pages 1012–1020, Columbus, Ohio, June. Association for Computational Linguistics.

Yang Feng, Yang Liu, Haitao Mi, Qun Liu, and Yajuan Lü. 2009. Lattice-based system combination for statistical machine translation. In *Proceedings of EMNLP*, pages 1105–1113, Singapore, August. Association for Computational Linguistics.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of ACL - Demo and Poster Sessions*, pages 177–180, Prague, Czech Republic, June. Association for Computational Linguistics.

Chunyang Liu, Qi Liu, Yang Liu, and Maosong Sun. 2012. THUTR: A translation retrieval system. In *Proceedings of COLING - Demo and Poster Sessions*, pages 321–328, Mumbai, India, December. The COLING 2012 Organizing Committee.

Wolfgang Macherey, Franz Och, Ignacio Thayer, and Jakob Uszkoreit. 2008. Lattice-based minimum error rate training for statistical machine translation. In *Proceedings of EMNLP*, pages 725–734, Honolulu, Hawaii, October. Association for Computational Linguistics.

Spyros Matsoukas, Ivan Bulyko, Bing Xiang, Kham Nguyen, Richard Schwartz, and John Makhoul. 2007. Integrating speech recognition and machine translation. In *Proceedings of ICASSP*, volume 4, pages IV–1281. IEEE.

C.N. Moore. 1958. A mathematical theory of the use of language symbols in retrieval. In *ICSI 1958*.

Dragos Munteanu and Daniel Marcu. 2005. Improving machine translation performance by exploiting non-parallel corpora. *Computational Linguistics*, 31(4):477–1504.

S. Nirenburg, C. Domashnev, and D.J. Grannes. 1993. Two approaches to matching in example-based machine translation. In *TMI 1993*.

Franz Josef Och and Hermann Ney. 2002. Discriminative training and maximum entropy models for statistical machine translation. In *Proceedings of ACL*, pages 295–302, Philadelphia, Pennsylvania, USA, July. Association for Computational Linguistics.

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

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of ACL*, pages 311–318, Philadelphia, Pennsylvania, USA, July. Association for Computational Linguistics.

Uta Priss. 2000. Lattice-based information retrieval. *Knwoledge Organization*, 27(3):132–142.

Murat Saraclar and Richard Sproat. 2004. Lattice-based search for spoken utterance retrieval. In Daniel Marcu Susan Dumais and Salim Roukos, editors, *HLT-NAACL*, pages 129–136, Boston, Massachusetts, USA, May. Association for Computational Linguistics.

S. Sato and M. Nagao. 1990. Toward memory-based translation. In *Proceedings of COLING*.

Andreas Stolcke. 2002. Srilm: an extensible language modeling toolkit. In *Proceedings of ICSLP*.

Roy Tromble, Shankar Kumar, Franz Och, and Wolfgang Macherey. 2008. Lattice Minimum Bayes-Risk decoding for statistical machine translation. In *Proceedings of EMNLP*, pages 620–629, Honolulu, Hawaii, October. Association for Computational Linguistics.

Jia Xu, Evgeny Matusov, Richard Zens, and Hermann Ney. 2005. Integrated chinese word segmentation in statistical machine translation. In *Proceedings of IWSLT 2005*, pages 141–147, Pittsburgh, PA, October.

Zheng-Yu Zhou, Peng Yu, Ciprian Chelba, and Frank Seide. 2006. Towards spoken-document retrieval for the internet: Lattice indexing for large-scale web-search architectures. In *Proceedings of HLT-NAACL*, pages 415–422, New York City, USA, June. Association for Computational Linguistics.