Iterative Learning of Parallel Lexicons and Phrases from Non-Parallel Corpora

Meiping Dong† Yang Liu†‡ Huanbo Luan† Maosong Sun†‡ Tatsuya Izuha‡ Dakun Zhang‡
†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
‡Jiangsu Collaborative Innovation Center for Language Competence, Jiangsu, China
hello@163.com, {liuyang2011,sms}@tsinghua.edu.cn, luanhuanbo@gmail.com
†‡Toshiba Corporation Corporate Research & Development Center
tatsuya.izuha@toshiba.co.jp
‡Toshiba (China) R&D Center
zhangdakun@toshiba.com.cn

Abstract

While parallel corpora are an indispensable resource for data-driven multilingual natural language processing tasks such as machine translation, they are limited in quantity, quality and coverage. As a result, learning translation models from non-parallel corpora has become increasingly important nowadays, especially for low-resource languages. In this work, we propose a joint model for iteratively learning parallel lexicons and phrases from non-parallel corpora. The model is trained using a Viterbi EM algorithm that alternates between constructing parallel phrases using lexicons and updating lexicons based on the constructed parallel phrases. Experiments on Chinese-English datasets show that our approach learns better parallel lexicons and phrases and improves translation performance significantly.

1 Introduction

Parallel corpora, which are collections of parallel texts, play a critical role in data-driven multilingual natural language processing (NLP) tasks such as statistical machine translation (MT) and cross-lingual information retrieval. For example, in statistical MT, parallel corpora serve as the central source for estimating translation model parameters [Brown et al., 1993; Koehn et al., 2003; Chiang, 2005]. It is widely accepted that the quantity, quality, and coverage of parallel corpora have an important effect on the performance of statistical MT systems.

Despite the apparent success of data-driven multilingual NLP techniques, the availability of large-scale, wide-coverage, high-quality parallel corpora still remains a major challenge. For most language pairs, parallel corpora are nonexistent. Even for the top handful of resource-rich languages, the available parallel corpora are usually unbalanced because the major sources are government documents or news articles.

As a result, learning translation models from non-parallel corpora has attracted intensive attention from the community [Koehn and Knight, 2002; Fung and Cheung, 2004; Munteanu and Marcu, 2006; Quirk et al., 2007; Ueffing et al., 2007; Haghhighi et al., 2008; Bertoldi and Federico, 2009; Cettolo et al., 2010; Daumé III and Jagarlamudi, 2011; Ravi and Knight, 2011; Nuhn et al., 2012; Dou and Knight, 2012; Klementiev et al., 2012; Zhang and Zong, 2013; Dou et al., 2014]. Most existing approaches focus on learning word-based models: either bilingual lexicons or IBM models. Based on canonical correlation analysis (CCA), Haghhighi et al. [2008] leverage orthographic and context features to induce word translation pairs. Ravi and Knight [2011] cast training IBM models on monolingual data as a decipherment problem. However, word-based models are not expressive enough to capture non-local dependencies and therefore are insufficient to yield high quality translations.

Recently, several authors have moved a step further to learn phrase-based models from non-parallel corpora [Klementiev et al., 2012; Zhang and Zong, 2013]. Zhang and Zong [2013] propose to use a parallel lexicon to retrieve parallel phrases from non-parallel corpora. They show that their approach can learn new translations and improve translation performance. However, their approach is unidirectional: only using lexicons to identify parallel phrases. In fact, it is possible to learn better lexicons from extracted phrase pairs in a reverse direction, which potentially constitutes a “find-one-get-more” loop [Fung and Cheung, 2004].

In this paper, we propose an iterative approach to learning bilingual lexicons and phrases jointly from non-parallel corpora. Given two sets of monolingual phrases that might contain parallel phrases, we develop a generative model based on IBM model 1 [Brown et al., 1993], which treats the mapping between phrase pairs as a latent variable. The model is trained using a Viterbi EM algorithm. Experiments on Chinese-English datasets show that iterative training significantly improves the quality of learned bilingual lexicons and phrases and benefit end-to-end MT systems.

2 Preliminaries

We begin with a brief introduction to IBM model 1, which is the core component of our generative model.

Let \( f = f_1, \ldots, f_J \) be a foreign sentence with \( J \) words. We use \( f_j \) to denote the \( j \)-th word in the foreign sentence. Similarly, \( e = e_1, \ldots, e_I \) is an English sentence with \( I \) words and \( e_i \) is the \( i \)-th word. \( f \) and \( e \) denote single foreign and English words, respectively. A word alignment \( a = a_1, \ldots, a_J \)
indicates the correspondence between $f$ and $e$. Brown et al. [1993] restrict that each foreign word is aligned to exactly one English word: $a_t \in \{0, 1, \ldots, J\}$, where $e_0$ is defined to be an empty English word.

According to Brown et al. [1993], IBM model 1 is defined as follows:

$$P(f|e; \theta) = \sum_a P(f, a|e; \theta) \tag{1}$$
$$= \sum_a \lambda_j \frac{p(J|I)}{(I+1)^j} \prod_{j=1}^J p(f|e_{a_j}) \tag{2}$$

where the set of model parameters $\theta$ consists of length probabilities $p(J|I)$ and translation probabilities $p(f|e)$:

$$\forall I : \sum_f p(J|I) = 1 \tag{3}$$
$$\forall e : \sum_f p(f|e) = 1 \tag{4}$$

Brown et al. [1993] indicate that the right-hand side of Eq. (2) is a sum of terms each of which is a monomial in the translation probabilities. Therefore, we have

$$P(f|e; \theta) = \frac{p(J|I)}{(I+1)^J} \prod_{j=1}^J \sum_{e_i} p(f|e_i) \tag{5}$$

Given a set of parallel sentences $D = \{(f^{(s)}, e^{(s)})\}_{s=1}^S$, the training objective is to maximize the log-likelihood of the training data:

$$J(\theta) = \sum_{s=1}^S \log P(f^{(s)}|e^{(s)}) - \sum_e \lambda_e \left( \sum_f p(f|e) - 1 \right) \tag{6}$$

Brown et al. [1993] use the EM algorithm to estimate translation probabilities, $c(f|e; D)$, which is the expected number of times that $e$ connects to $f$ on the training data, is calculated as

$$\sum_{s=1}^S \frac{p(f|e)}{\sum_{f'} p(f'|e^{(s)})} \sum_{j=1}^{J^{(s)}} \delta(f, f^{(s)}_{j}) \sum_{i=0}^{I^{(s)}} \delta(e, e^{(s)}_i) \tag{7}$$

where $J^{(s)}$ and $J^{(t)}$ are lengths of $e^{(s)}$ and $f^{(t)}$, respectively. Then, the new translation probabilities can be obtained by normalizing these counts:

$$p(f|e) = \frac{c(f|e; D)}{\sum_{f'} c(f'|e; D)} \tag{8}$$

3 Model

In this work, we are interested in learning IBM Model 1 from non-parallel corpora: a set of foreign strings $F = \{f^{(i)}\}_{i=1}^T$ and a set of English strings $E = \{e^{(s)}\}_{s=1}^S$. Our goal is twofold:

1. Extract a parallel corpus from non-parallel corpora $F$ and $E$ using IBM model 1.
2. Train IBM model 1 from the extracted parallel corpus.

Due to data sparsity, a long foreign sentence in $F$ can hardly have an English translation in $E$. Alternatively, we assume that a short foreign string, say a phrase with up to 7 words, in $F$ can potentially have an English translation in $E$ [Munteanu and Marcu, 2006; Quirk et al., 2007; Cettolo et al., 2010]. Therefore, in this work, $F$ and $E$ are defined to be sets of phrases. We can easily obtain such monolingual phrase sets by collecting $n$-grams.

We introduce phrase matching to denote the correspondence between phrases in $F$ and $E$. A foreign phrase $f$ is said to match an English phrase $e$ if they are translations of each other. More formally, we use $M$ to denote a set of all possible matchings and $m$ a matching. Following Brown et al. [1993], we restrict that each foreign phrase matches exactly one English phrase: $m = m_1, \ldots, m_T$, where $m_i \in \{0, 1, \ldots, S\}$. Note that we allow a foreign phrase to connect to an empty English phrase $e^{(0)}$.

The joint model is defined as

$$P(F,E; \theta) = \sum_m P(F, m; E; \theta) \tag{9}$$
$$= \sum_m \frac{P(T|S)}{(S+1)^T} \prod_{t=1}^T P(f^{(t)}|e^{(m_t)}; \theta) \tag{10}$$

We refer to $P(f^{(t)}|e^{(m_t)}; \theta)$ as phrase translation model, which can be divided into two categories: empty and non-empty. The probability of a foreign phrase given an empty English phrase is defined to be a model parameter satisfying

$$\sum_{f \in F} p(f|e^{(0)}) = 1 \tag{11}$$

where $F$ is the set of all possible foreign phrases. Note that $F$ in the training data is just a subset of $F$. As $F$ cannot be fully enumerated, we set $p(f|e^{(0)})$ to a fixed number $c$, which is a hyper-parameter to be optimized on the held-out data.

For non-empty English phrases, we use IBM Model 1 as follows:

$$P(f^{(t)}|e^{(m_t)}; \theta) = \frac{p(J^{(t)}|I^{(m_t)})}{(I^{(m_t)}+1)^J} \prod_{j=1}^{J^{(t)}} \sum_{i=0}^{I^{(m_t)}} p(f^{(t)}_j|e^{(m_t)}_i) \tag{12}$$

Note that the length model $p(J^{(t)}|I^{(m_t)})$ plays an important role in learning IBM model 1 from non-parallel corpora since the model needs to choose among source phrases with various lengths.

Therefore, the phrase translation model is defined as

$$P(f^{(t)}|e^{(m_t)}; \theta)$$

$\delta(m_t, 0) e +$

$$\left(1 - \delta(m_t, 0)\right) \frac{p(J^{(t)}|I^{(m_t)})}{(I^{(m_t)}+1)^J} \prod_{j=1}^{J^{(t)}} \sum_{i=0}^{I^{(m_t)}} p(f^{(t)}_j|e^{(m_t)}_i) \tag{13}$$
4 Training

4.1 The Viterbi EM Algorithm

Given monolingual corpora $F$ and $E$ and a parallel lexicon $d$, the training objective is given by

$$ J(\theta) = P(F|E; \theta) - \sum_f \sum_e \sigma(f, e, d) \log \frac{\sigma(f, e, d)}{p(f|e)} $$

where $d$ is a parallel lexicon and $\sigma(f, e, d)$ checks whether $(f, e)$ exists in the lexicon:

$$ \sigma(f, e, d) = \begin{cases} 1 & \text{if } (f, e) \in d \\ 0 & \text{otherwise} \end{cases} $$

It is natural to use the EM algorithm to estimate the model parameters. The expected count of the length model $c(J|I; F, E)$ is given by

$$ E_{F, m|E; \theta} \left[ \sum_{t=1}^T (1 - \delta(m_t, 0)) \delta(J^{(t)}, J) \delta(f^{(m_t)}, I) \right] $$

The expected count of the translation model $c(f|e; F, E)$ is given by

$$ E_{F, m|E; \theta} \left[ \sum_{t=1}^T (1 - \delta(m_t, 0)) \frac{p(f|e)}{\sum_{j=0}^{J(m_t)} \sum_{i=0}^{F(m_t)} p(f^{(i)}|e^{(j)})} \times \sum_j \delta(f^{(i)}, e) \sum_{j=0}^{J(m_t)} \delta(e^{(j)}, e) \right] + \sigma(f, e, d) $$

Unfortunately, calculating the two expectations in Eq. (17) and (18) on the training data is intractable due to the exponential search space of phrase matching.

Instead, we use a Viterbi EM algorithm as shown in Figure 1. The algorithm takes a set of foreign phrases $F$, a set of English phrases $E$, and a parallel lexicon $d$ as input (line 1). After initializing model parameters (line 2), the algorithm calls the procedure $\text{ALIGN}(F, E, \theta)$ to compute the Viterbi matching between $F$ and $E$ (line 4). Then, the algorithm updates the model by normalizing counts collected from the Viterbi matching (line 5). This process terminates after $K$ iterations and returns the final matching and model.

1: procedure VITERBIEM($F$, $E$, $d$)
2: Initialize $\theta(0)$
3: for all $k = 1, \ldots, K$ do
4:    $\tilde{m}^{(k)} \leftarrow \text{ALIGN}(F, E, \theta^{(k-1)})$
5:    $\theta^{(k)} \leftarrow \text{UPDATE}(F, E, d, \tilde{m}^{(k)})$
6: end for
7: return $\tilde{m}^{(K)}, \theta^{(K)}$
8: end procedure

Figure 1: A Viterbi EM algorithm for learning IBM model 1 from non-parallel corpora. $F$ and $E$ are sets of foreign and English phrases, $\theta^{(k)}$ is the set of parameters at the $k$-th iteration, and $\tilde{m}^{(k)}$ is the Viterbi matching between $F$ and $E$.

4.2 Computing Viterbi Matching

Given a set of foreign phrases $F$ and a set of English phrases $E$, the Viterbi matching is defined as

$$ \tilde{m} = \arg\max_{m} \left\{ P(F, m|E; \theta) \right\} $$

$$ = \arg\max_{m} \left\{ \prod_{t=1}^T P(f^{(t)}|e^{(m_t)}; \theta) \right\} $$

It is clear that computing the Viterbi matching for individual foreign phrases is independent. We only need to focus on finding the most probable English phrase for each foreign phrase:

$$ \tilde{m_s} = \arg\max_{s \in \{0, 1, \ldots, S\}} \left\{ P(f^{(s)}|e^{(s)}; \theta) \right\} $$

For the empty English phrase (i.e., $s = 0$), the translation probability is simply $\epsilon$. For non-empty English phrases, the decision rule is

$$ \tilde{m_t} = \arg\max_{s \in \{1, \ldots, S\}} \left\{ \frac{p(f^{(t)}|i^{(s)})}{(i^{(s)}+1)J^{(s)}} \prod_{j=1}^{J^{(s)}} p(f^{(j)}|e^{(i^{(j)})}) \right\} $$

Therefore, the Viterbi matching for a foreign phrase can be determined by

$$ \tilde{m_t} = \begin{cases} \tilde{m_t} & \text{if } P(f^{(t)}|e^{(m_t)}; \theta) > \epsilon \\ 0 & \text{otherwise} \end{cases} $$

We can see that $\epsilon$ determines whether a foreign phrase is unaligned or not. This is important for preventing a foreign phrase that has no counterparts on the English side from connecting to a wrong English phrase.

Computing $\tilde{m_t}$ in Eq. (22) is computationally expensive if $S$ is very large. In practice, we use information retrieval (IR) techniques to speed up the computation, which has been suggested by Zhang and Zong [2013]. While standard cross-lingual IR focuses on the relevance between queries and documents and ignores function words (e.g., punctuations), our goal is to find translations in the set of English phrases. Both content and function words are important for indexing and retrieval. Therefore, we use translation probabilities instead of term weights such as tf.idf in retrieval for indexing.
We build a term-document incidence matrix \( \mathcal{M} \in \mathbb{R}^{|V_f| \times |S|} \) for all foreign words, where \( V_f \) is the foreign vocabulary. Each element in the matrix \( \mathcal{M}(f, s) \) stores the probability that \( f \) given the English words in \( e^{(s)} \). For example, suppose the 621-th English phrase is “the program has been implemented” and \( p(“chengxu”|“program”) = 0.58 \), we set \( \mathcal{M}(“chengxu”, 621) = 0.58 \). If another English word is also a translation of “chengxu”, the matrix retains the highest translation probability. Hence, English phrases can be efficiently retrieved in a monolingual way.

For efficiency, we use a coarse-to-fine approach to computing the Viterbi matching for non-empty English phrases. First, the term-document incidence matrix is used to filter most unlikely source phrases and return a coarse set of candidates. Then, phrase translation probabilities are exactly calculated using Eq. (12).

### 4.3 Updating Model Parameters

Given the Viterbi matching, the count of the length model \( c(J|I; F, E) \) is simply

\[
\sum_{t=1}^{T} (1 - \delta(\hat{m}_t, 0)) \delta(J^{(t)}, J) \delta(f^{(\hat{m}_t)}), l)
\]

(24)

The count of the translation model \( c(f|e; F, E) \) is given by

\[
\sum_{t=1}^{T} (1 - \delta(\hat{m}_t, 0)) \frac{p(f|e)}{\sum_{i=0}^{I^{(m_t)}} p(f|e^{(m_t)})}
\]

\[
\times \sum_{j=1}^{J} \delta(f, f^{(j)}) \sum_{i=0}^{I^{(m_t)}} \delta(e, e^{(m_t)})
\]

\[
+ \sigma(f, e, d)
\]

(25)

### 5 Experiments

#### 5.1 Matching Evaluation

To measure how well our approach identifies parallel phrases from non-parallel corpora \( F \) and \( E \), we define the matching accuracy as

\[
\sum_{t=1}^{T} \delta(\hat{m}_t, m_t^*) \]

(26)

where \( T \) is number of all foreign phrases, \( \hat{m} \) is the Viterbi matching predicted by a model, and \( m^* \) is the gold-standard matching.

<table>
<thead>
<tr>
<th>Dev</th>
<th>Chinese</th>
<th>English</th>
</tr>
</thead>
<tbody>
<tr>
<td>Phrases</td>
<td>2,000</td>
<td>4,000</td>
</tr>
<tr>
<td>Vocabulary</td>
<td>2,994</td>
<td>4,366</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Test</th>
<th>Chinese</th>
<th>English</th>
</tr>
</thead>
<tbody>
<tr>
<td>Phrases</td>
<td>20,000</td>
<td>40,000</td>
</tr>
<tr>
<td>Vocabulary</td>
<td>10,315</td>
<td>13,233</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Dict</th>
<th>Chinese</th>
<th>English</th>
</tr>
</thead>
<tbody>
<tr>
<td>Entries</td>
<td>1,000</td>
<td></td>
</tr>
<tr>
<td>Vocabulary</td>
<td>567</td>
<td>798</td>
</tr>
</tbody>
</table>

Table 1: Statistics of the development set (Dev), test set (Test), and parallel dictionary (Dict).

<table>
<thead>
<tr>
<th>iteration</th>
<th>log-likelihood</th>
<th>accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>-88488</td>
<td>38.50</td>
</tr>
<tr>
<td>2</td>
<td>-36238</td>
<td>49.75</td>
</tr>
<tr>
<td>3</td>
<td>-28568</td>
<td>51.12</td>
</tr>
<tr>
<td>4</td>
<td>-26813</td>
<td>51.70</td>
</tr>
<tr>
<td>5</td>
<td>-25752</td>
<td>52.00</td>
</tr>
<tr>
<td>6</td>
<td>-25281</td>
<td>52.15</td>
</tr>
<tr>
<td>7</td>
<td>-25277</td>
<td>52.20</td>
</tr>
<tr>
<td>8</td>
<td>-25264</td>
<td>52.25</td>
</tr>
<tr>
<td>9</td>
<td>-25253</td>
<td>52.30</td>
</tr>
<tr>
<td>10</td>
<td>-24642</td>
<td>52.35</td>
</tr>
</tbody>
</table>

Table 2: Effect of empty translation probability \( \epsilon \) on matching accuracy on the development set.

Table 3 shows the log-likelihood and matching accuracy on the development set. We use the development set to optimize the hyperparameter \( \epsilon \). Table 2 shows the effect of empty translation probability \( \epsilon \) on accuracy. The Viterbi EM algorithm runs for 10 iterations. We find that \( \epsilon \) does not have a significant impact on accuracy. Therefore, we set \( \epsilon = e^{-30} \) in the following experiments.

Table 3: Log-likelihood and matching accuracy on the development set.
Table 4: Effect of dictionary size on matching accuracy on the development set.

| |d| | accuracy |
|---|---|---|
| 0 | 0.05 |
| 10 | 0.55 |
| 50 | 2.45 |
| 100 | 6.70 |
| 500 | 34.20 |
| 1,000 | 52.35 |

Table 5: Effect of noises on matching accuracy on the development set.

<table>
<thead>
<tr>
<th>noises</th>
<th>accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>59.30</td>
</tr>
<tr>
<td>100</td>
<td>57.65</td>
</tr>
<tr>
<td>500</td>
<td>55.30</td>
</tr>
<tr>
<td>1,000</td>
<td>54.95</td>
</tr>
<tr>
<td>1,500</td>
<td>53.45</td>
</tr>
<tr>
<td>2,000</td>
<td>52.35</td>
</tr>
</tbody>
</table>

Table 4 shows the effect of dictionary size on matching accuracy. Clearly, using larger seed dictionaries improves accuracy since the dictionary with 3,000 entries covers most words in the development set. 

Table 5 shows the effect of noises on matching accuracy. Without noises, our model is able to achieve an accuracy of 59.30%.

Our final result on the test set is 40.18% as shown in Figure 2. The Viterbi EM algorithm ran for 70 iterations, the dictionary size is 1,000, and the empty translation probability $\epsilon = e^{-30}$.

We analyzed the errors made by our model and found that most wrongly identified phrases are actually very close to the gold-standard phrases. For example, given a Chinese phrase “ta biaoashi jingji”, our model identifies an English phrase “he says economy” while the gold-standard phrase is “he said economy”. Therefore, although the matching accuracy is relatively low, the identified phrase pairs are still valuable for machine translation, as shown in the following section.

5.2 Translation Evaluation

We follow Zhang and Zong [2013] to evaluate our approach on domain adaptation for machine translation. Given an out-domain parallel corpus $L$ and an in-domain non-parallel corpus $U$, the task is to maximize the translation performance on unseen in-domain text.

Our experiment runs as follows:

1. Train IBM model 1 on the out-domain parallel corpus $L$ and obtain initial model parameters $\theta$;

2. Identify parallel phrase pairs $P$ from the in-domain non-parallel corpus $U$ using $\theta$;

3. Evaluate the combined parallel corpora $L \cup P$ by training phrase-based models and calculating the BLEU score on unseen in-domain test set;

4. Update model parameters $\theta$ on $L \cup P$;

5. Go to step 2 or terminate if the number of iterations reaches the predefined limit (e.g., 5).

The out-domain parallel corpus $L$ (financial articles from FTChina) we used consists of 7,360 pairs of Chinese-English phrases with 41,279 Chinese words and 41,123 English words. The average lengths of Chinese and English phrases are 5.61 and 5.59. The in-domain non-parallel corpus $U$ (news reports from LDC) consists of 2,935,265 Chinese phrases and 3,961,087 English phrases. The average lengths of Chinese and English phrases are 5.67 and 5.91.

We used Moses [Koehn and Hoang, 2007] to learn phrase-based translation models on the combined parallel corpus $L \cup P$ and evaluate translation performance on NIST datasets. We used the SRILM toolkit [Stolcke, 2002] to train a 4-gram language model on the Xinhua portion of the GIGAWORD corpus, which contains 238M English words. The NIST 2002 MT Chinese-English dataset is our the development set and the NIST 2005 dataset is the test set. The evaluation metric is case-insensitive BLEU4 [Papineni et al., 2002].

Table 6 lists the statistics of constructed parallel corpus $L \cup P$ and resulting BLEU scores. At iteration 0, only the out-domain parallel corpus $L$ is used. Then, our system iteratively extracts parallel phrases from the in-domain non-parallel corpus $U$. We find that the constructed parallel corpus keeps growing by including new phrase pairs that do not exist in $L$. The size of translation probability table of IBM Model 1 also increases significantly, suggesting that our approach is able to learn translations of unseen words. As a result, our approach achieves statistically significant improvements than using the starting out-domain parallel corpus.

Table 7 shows some example learned parallel phrases. Note that these phrases do not exist in the out-domain parallel corpus $L$. Words that are unseen in $L$ are highlighted in...
Table 6: Results on domain adaptation for machine translation. At iteration 0, we are given a small out-domain parallel corpus with 7,360 phrase pairs. Our system iteratively learns IBM model 1 from the combination of the out-domain parallel corpus and in-domain non-parallel corpus and yields a set of matched in-domain parallel phrases as a byproduct. The quality of constructed parallel corpus is measured by training phrase-based translation models using Moses and evaluating on NIST datasets. Our approach achieves statistically significant improvement than using starting out-domain parallel corpus.

<table>
<thead>
<tr>
<th>iteration</th>
<th>corpus size</th>
<th>Chinese Vocab.</th>
<th>English Vocab.</th>
<th>tTable size</th>
<th>BLEU</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>7,360</td>
<td>4,149</td>
<td>4,134</td>
<td>8,494</td>
<td>7.68</td>
</tr>
<tr>
<td>1</td>
<td>43,401</td>
<td>7,794</td>
<td>6,638</td>
<td>31,381</td>
<td>11.21</td>
</tr>
<tr>
<td>2</td>
<td>102,396</td>
<td>11,843</td>
<td>9,338</td>
<td>64,748</td>
<td>12.65</td>
</tr>
<tr>
<td>3</td>
<td>135,290</td>
<td>13,334</td>
<td>10,474</td>
<td>77,924</td>
<td>12.77</td>
</tr>
<tr>
<td>4</td>
<td>148,177</td>
<td>14,059</td>
<td>11,017</td>
<td>82,599</td>
<td>13.23</td>
</tr>
<tr>
<td>5</td>
<td>153,681</td>
<td>14,447</td>
<td>11,256</td>
<td>85,279</td>
<td>13.40</td>
</tr>
</tbody>
</table>

Table 7: Example learned parallel phrases. These phrases do not appear in the out-domain parallel corpus $L$. Words that are unseen in $L$ are highlighted in bold.

<table>
<thead>
<tr>
<th>id</th>
<th>Chinese phrase</th>
<th>English phrase</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>qi zhuyao chanpin shi shiwu</td>
<td>its main products are food</td>
</tr>
<tr>
<td>2</td>
<td>meiguo jiding de zhanlue jihua</td>
<td>the set us strategic plan</td>
</tr>
<tr>
<td>3</td>
<td>quanqiuhua de qushi</td>
<td>the trends of globalization</td>
</tr>
<tr>
<td>4</td>
<td>qianghua lianhe zuozhan gongneng</td>
<td>strengthening joint combat</td>
</tr>
<tr>
<td></td>
<td></td>
<td>functions</td>
</tr>
<tr>
<td>5</td>
<td>yisilanjiao, tianzhujiatuo he jiduijiao</td>
<td>islam, catholicism, and christianity</td>
</tr>
</tbody>
</table>

6 Related Work

Joint Parallel Sentence and Lexicon Extraction. Advocating the “find-one-get-more” principle, Fung and Cheung [2004] propose an iterative framework to extract parallel sentences and lexicons from non-parallel corpora via bootstrapping and EM. They first use similarity measures to match documents in non-parallel corpora and then extract parallel sentences and new translations from these documents by exploiting bootstrapping on top of IBM model 4. While similar in spirit to their idea, our work assumes that parallel phrases exist in monolingual corpora and develops a new model to capture the correspondence between monolingual phrases in a principled way.

Parallel Sub-Sentential Fragment Extraction. Observing that comparable corpora often contain parallel words or phrases, a number of authors have proposed to extract such parallel fragments from noisy parallel sentences [Munteanu and Marcu, 2006; Quirk et al., 2007; Cettolo et al., 2010]. They first extract candidate parallel sentences from comparable corpora and then identify parallel fragments from the noisy candidate parallel sentences. This approach assumes that comparable corpora are readily available and focuses on finding parallel fragments within comparable sentence pairs, our approach directly builds a generative model on fragments, which can be $n$-grams extracted from monolingual corpora.

Extracting Parallel Phrases using Lexicons. Our work is also similar to [Zhang and Zong, 2013] that exploits a bilingual lexicon to retrieve parallel phrases from monolingual corpora. The major difference is that their approach is non-iterative. Our approach uses a Viterbi EM algorithm and can improve the learned model iteratively. Experiments show that our iterative approach significantly outperforms the non-iterative approach (see Table 6: 13.40 vs. 11.21).

Translation as Decipherment. Another interesting line of research casts translation with monolingual corpora as a decipherment problem [Ravi and Knight, 2011; Nuhn et al., 2012; Dou and Knight, 2012]. They develop word-based generative models on monolingual corpora and use sampling methods for efficient training. While our approach is also based on word-based translation model, it constructs parallel corpus in the EM loop, which significantly reduces the training complexity.

Transductive Learning on Monolingual Corpora. A number of authors leverage transductive learning to make full use of monolingual data [Ueffing et al., 2007; Bertoldi and Federico, 2009]. They use an existing translation model to translate unseen source-language text. Then, the input and their translations constitute a pseudo parallel corpus. This process iterates until convergence. Zhang and Zong [2013] indicate that this approach can hardly learn new translations consisting of words unseen in the seed parallel corpus.

7 Conclusion

We have presented an iterative approach to learning parallel lexicons and phrases from non-parallel corpora. Experiments show that iterative learning can offer valuable parallel phrases to machine translation systems. In the future, we plan to extend our approach to include more sophisticated alignment models such as IBM models 2-5 and HMM [Vogel et al., 1996]. It is also interesting to scale up to web text and build a never-ending parallel lexicon and phrase mining system.
Acknowledgements
Yang Liu and Maosong Sun are supported by the National Natural Science Foundation of China (No. 61331013 and No. 61432013), the 863 Program (2015AA015407) and Toshiba Corporation Corporate Research & Development Center. Huanbo Luan is supported by the National Natural Science Foundation of China (No. 61303075). This research is also supported by the Singapore National Research Foundation under its International Research Centre@Singapore Funding Initiative and administered by the IDM Programme.

References