Hierarchical Sub-sentential Alignment with IBM Models for Statistical Phrase-based Machine Translation

Hao Wang[†] and Yves Lepage[†]

In this paper, we describe a novel method for joint word alignment and symmetrization. Based on initial parameters from simple IBM models, we synchronously parse the parallel sentence pair under the framework of bracket transduction grammar constraints. Our 2-phase method can achieve nearly the same run-time as `fast_align` while delivering better alignments on distantly-related language pairs such as English–Japanese. We show how to integrate this method into a standard phrase-based SMT pipeline. Although the alignment quality results are mixed, by forcing all words to be aligned (*1-to-many/many-to-1*), our method significantly reduces the phrase table size with no difference in translation quality and even outperforms `fast_align` in some end-to-end translation experiments.

Key Words: *word alignment, symmetrization, machine translation, parsing*

1 Introduction

Word alignment is the fundamental issue in many tasks of natural language processing. For instance, state-of-the-art statistical machine translation (SMT) methods require word-to-word aligned data on a parallel corpus, for the various purposes of extracting reusable translation fragments, e.g., word pairs (Brown, Cocke, Pietra, Pietra, Jelinek, Mercer, and Roossin 1988), phrase pairs (Koehn, Och, and Marcu 2003), hierarchical rules (Chiang 2007), tree-to-string correspondences (Liu, Liu, and Lin 2006) and tree-to-tree templates (Zhang, Jiang, Aw, Sun, Li, and Tan 2007). Since word alignment is the basis to perform these tasks, the reliability and accuracy of the word aligner directly affect the end-to-end MT performance in most cases (Fraser and Marcu 2007), which seems intuitive.

A standard phrase-based SMT pipeline can be summarized as a serial process: *align-extract-tune-decode*. Word alignment is the essential preprocessing step for phrase extraction. Most of the word alignment methods are based on the generative models, like IBM models (Brown, Pietra, Pietra, and Mercer 1993) and, besides, they can also be augmented with an HMM-based model (Vogel, Ney, and Tillmann 1996). In general, we can group these models into two types, as the pioneering work (Liang, Taskar, and Klein 2006): sequence-based models (IBM model 1, 2 and HMM model) and fertility-based models (IBM model 3, 4 and 5). Intuitively, the high-level

[†] Graduate School of Information, Production and Systems, Waseda University

fertility-based models become intractable from simpler sequence-based models, which is not easy to implement. As a result, even when using `GIZA++`[1], the most widely used aligner at present, which consists of various IBM models and their extensions, IBM model 4 is selected as default to find a trade-off between alignment quality and efficiency.

There exist several problems in the presented models. For sequence-based models, the EM algorithm (Dempster, Laird, and Rubin 1977) is applied to find the optimal parameters. Recently, some researchers also work on Bayesian word alignment, using techniques such as Gibbs sampling (Mermer and Saraclar 2011). Typically to say, the training process runs in both directions, then yields two sets of parameters, and eventually two directional alignments. Since sequence-based models restrict to *many-to-1* alignments, they are prone to produce discontinuous alignments. For example, under the definition of IBM models for alignment, we can map several indices of the source side to the same index of the target side, and it is possible that some indices on the target side are not being mapped. Because the phrase-based SMT relies more on alignments of multi-word units (*many-to-many*), called phrase pairs, which commonly cannot be correctly extracted during processing, especially when these phrase pairs are needed to build the translation hypothesis during decoding. Intuitively, this will have a significant influence on the final quality of the translation.

Due to the drawback of discontinuity in sequence-based models, high-level fertility-based models were developed. These models are expected to generate more word alignments to fill the empty cells between cells with high confident in alignment matrices. There is no doubt that these models generate state-of-the-art alignments but the fact that IBM model 3 and above are often criticized for their complexity, which is much time-consuming during training, even within the highly optimized implementations in `GIZA++`. To deal with this problem, Dyer, Chahuneau, and Smith (2013) employ a variation of sequence-based models rather than fertility-based models to generate word alignment. They showed that in the implementation (`fast_align`[2]) can yield comparable results on some language pairs in end-to-end translation experiments. `Fast_align` yields less alignment. Wang and Lepage (2016a) showed that the number of word alignments output by `fast_align` is 19% less with `GIZA++`. Moreover, word-position-featured method is not really suitable for the distantly-related language pairs, Ding, Utiyama, and Sumita (2015) demonstrated that `fast_align` does not outperform `GIZA++` on Japanese-English and English-German experiments.

[1] http://www.statmt.org/moses/giza/`GIZA++`.html
[2] https://github.com/clab/fast_align

However, both sequence-based and fertility-based models still generate asymmetrical alignments and mono-directional alignments are not suitable for the bilingual tasks, like a phrase-based SMT. In order to obtain symmetric alignment, Och and Ney (2003) proposed to train models in both forward and reverse directions, whereafter merging the outcome of mono-directional alignments with some symmetrization heuristics. Among these heuristics, the *grow-diag-final-and* heuristic (GDFA) has been shown to be the most effective for phrase extraction for phrased-based SMT (Wu and Wang 2007). Contrary to introducing heuristics, there exists some work (Liang et al. 2006) which tries to train a balanced model by maximizing the agreement between two directional word alignments. Such work is beyond the scope of our focus because it would lead to more time during training. It is, therefore, natural to explore other methods for word alignment symmetrization.

Another challenge in word alignment, especially for distinctly-related language pairs, is modeling the permutations of words between the source and target sentences while, theoretically, IBM models have not studied such a problem. Due to the diversity of natural languages, the word orders of the source and target sentences are quite different, e.g., English and Japanese. For some free-order languages, there is even no significant relationship between the current alignment and previous alignment in sequence from left to right, seems in contrast with the presumption of HMM model and IBM model 2 under this case. In this work, we prefer to model our problem with Bracket Transduction Grammar (BTG) (Wu 1995). BTG is an effective method that can capture the reordering of words across bilingual sentence pair. In fact, constraints of BTG were better matched to the IBM decoding model (Zens, Ney, Watanabe, and Sumita 2004). Differing with the IBM models, BTG has shown its efficiency in a way that constrained the search space of distortion in word alignment (Zhang and Gildea 2005; Haghighi, Blitzer, DeNero, and Klein 2009; Riesa and Marcu 2010).

In this paper, rather than reparameterizing IBM models, we expect to achieve better translation results addressing the reason of discontiguous word alignment directly. To address this problem, we concentrate our attention on hierarchical alignment with BTG parsing. Instead of using synchronous parsing to search for Viterbi BTG alignments as in (Li, Yang, and Sun 2012), we propose to employ the bi-partitioning algorithm presented in (Lardilleux, Yvon, and Lepage 2012) to generate symmetrical BTG block alignments (*1-to-many/many-to-1*) with our beam search implementation. In particular, our method starts with an adjacency matrix which is initialized using the lexicon translation model obtained via fast running EM estimation with IBM models; then we apply the bi-partitioning algorithm. Our two-phase method takes a radically different strategy compared with previous BTG-based unsupervised/supervised methods

for word alignment.

To summarize, we propose a novel method for joint word alignment and symmetrization, which can be regarded as a hybridization of BTG parsing and IBM models. In Section 2, we introduce the previous related works and notions in word alignment. We also describe each alignment method briefly. In Section 3, we justify the proposed method with mathematical principles and give multiple comparisons with state-of-the-art methods in Section 4. We also report statistics on speed, alignment scores, translation table sizes and translation scores (BLEU and RIBES) in end-to-end translation experiments, which we discuss in detail in this article.

2 Related works

2.1 Viterbi alignment and symmetrization

State-of-the-art word alignment models contain a large number of parameters (e.g., word translation probabilities) that need to estimate in addition to the desired hidden alignment variables. The basic idea of the previous methods is to develop a model where the word alignment is a hidden variable (Och and Ney 2003), by applying some statistical estimation to obtain the most possible/Viterbi alignments. The problem of translation can be defined as: $Pr(\mathbf{e}|\mathbf{f}) = \sum_{\mathbf{a}} p(\mathbf{e}, \mathbf{a}|\mathbf{f})$. Here we use the symbol $p(\cdot)$ to denote general probability distributions. $\mathbf{a}$ is a "hidden" variable which is mapping from a source position i to a target position a_i. It is always possible to find a best alignment by maximizing the likelihood on the given parallel training corpus. Under sequence-based models, we have

$$p(\mathbf{e}, \mathbf{a}|\mathbf{f}) = \prod_{j=1}^{n} \sum_{i=0}^{m} \theta(e_j|f_{a_j}) \delta(a_j = i|j, m, n) \tag{1}$$

where $\mathbf{f} = (f_1, \ldots, f_m)$ is the source sentence and $\mathbf{e} = (e_1, \ldots, e_n)$ is the target sentence. The translation parameters $\theta(e_j|f_{a_j})$ are parameterized by an appropriate local conditional probability distributions.

$$\delta(a_j = i|j, m, n) = \begin{cases} p_0 & i = 0 \\ (1 - p_0) \times h(a_j = i|j, m, n) & i \neq 0 \end{cases} \tag{2}$$

For modeling the distortion parameters $\delta(a_j = i|j, m, n)$, IBM models are the popular models. There are summarized as follows as (Liang et al. 2006):

$$h(a_j = i|j, m, n, a_{j-1} = i') \propto \begin{cases} 1 & \text{IBM 1} \\ \phi(\frac{i}{m}, \frac{j}{n}) & \text{IBM 2} \\ \phi(i, i') & \text{HMM} \end{cases} \tag{3}$$

In the formula, we write i' to denote the last alignment point $(i', j-1)$ in history. Among these models, IBM model 1 assumes a uniform distribution for $p(a)$, which effectively means that the word order of the sentences are considered irrelevant. This is clearly not true in real translated sentences of most language pairs. However, a_j and a_{j-1} tend to be strongly correlated, for example, this is the case in English-French. Most research on word alignment has assumed some version of a word order model to capture this dependency. Perhaps the simplest version (Dyer et al. 2013) is used in IBM model 2 where the observation is made that $i/m \approx j/n$. In other words, sentences tend to have the same order of words in both languages. This is a very rough approximation. (Vogel et al. 1996) proposed instead to directly model $P(a_j - a_{j-1} = x|m)$. This describes the length x of the "jump" in the source sentence when moving one word forward in the target sentence, conditioned on the source sentence length n.

Based on the above defined model, a Viterbi alignment model returns mono-directional alignments ($\mathbf{f} \rightarrow \mathbf{e}$ or vice-versa) which maximize the following formula.

$$\hat{\mathbf{a}} = \hat{a}_1^n = \underset{a_j \in \{\text{all alignments}\}}{\text{argmax}} \sum_{\mathbf{j=1}}^{n} p(e_j, a_j = i|f_i) \tag{4}$$

This processing explains the reason why Viterbi alignment methods generate discontinuity in word alignments. Although the influence of discontinuity can be reduced by combining two sets of alignments $\hat{\mathbf{a}} = \hat{a}_1^n$ ($\mathbf{f} \rightarrow \mathbf{e}$) and $\hat{\mathbf{b}} = \hat{b}_1^m$ ($\mathbf{e} \rightarrow \mathbf{f}$) into one alignment matrix A using the *grow-diag-final-and* algorithm, sometimes, there may still exist some unaligned target words. Given the merged alignment, we can easily factor the lexical translation probabilities. Obviously, the existence of unaligned target words makes the process of phrase extraction to produce more translation fragments, and it is reasonable to allow *null-to-1* alignments when iterating over all possible boundaries in target side. This strategy largely increases the size of extracted phrase table enormously at the same time. To alleviate this problem, we can force to align each source word with a target word and vice versa by forbidding that any source/target word be unaligned in the given sentence pair. In our experiments, we showed our strategy is effective to reduce the size of phrase table.

2.2 BTG-based word alignment

Bracket Transduction Grammar (BTG) is an effective method to restrict the exploration of all the possible permutations. There has been some interest in using BTGs for the purpose of alignment (Zhang and Gildea 2005; Wang, Knight, and Marcu 2007; Xiong, Zhang, and Li 2010; Neubig, Watanabe, Mori, and Kawahara 2012). Initially, the BTG formalism (Wu 1995) offers a special case of synchronous context-free grammars, which are quite suitable as the weak constraint regarding synchronous parsing. In particular, Haghighi et al. (2009), Riesa and Marcu (2010) showed that BTG, which captures structural coherence between parallel sentences, helps in word alignment. During parsing, a BTG builds a synchronous parse tree for both the source and the target sentence, assuming that the trees have the same underlying structure (BTG tree) but that order of constituents may differ in the two languages. Hence, each leaf in the BTG tree stands for a word-to-word correspondence. There are three types of rule in a BTG:

$$S : \gamma \rightarrow [X_1 X_2] \quad | \quad I : \gamma \rightarrow < X_1 X_2 > \quad | \quad T : \gamma \rightarrow (f, e) \tag{5}$$

where X_1, X_2 and γ are non-terminal symbols, f and e are terminal strings, and [ ] denotes the same order for the two non-terminals in two languages, <> denotes the inversion case. These formalisms are slightly different with Inversion Transduction Grammar (ITG) (Wu 1997), where the lexical rule $\gamma \rightarrow (f, e)$ does not include any source/target empty rules. In our case, we force to output the phrase pair (f, e) as a single block alignment which the length is one at least and three at most to prevent *null* blocks.

The complexity of word alignment grows exponentially with the length of the source and the target sentences. BTG models provide a natural, alternative method to reduce the search space in aligning. By estimating the joint word alignment relation directly, it eliminates the need for any of the conventional heuristics. The biggest barrier to applying BTG for Viterbi alignment is the time complexity of naïve CYK parsing ($O(n^6)$), which makes it hard to deal with long sentences or large grammars in practice.

Most of the previous research attempts to reduce the computational complexity of BTG parsing with some pruning methods. Zhang and Gildea (2005) propose *tic-tac-toe* pruning by extending BTG with the additional lexical information based on IBM model 1 Viterbi probability. Haghighi et al. (2009) investigate pruning based on the posterior predictions from two joint estimated models. Li et al. (2012) present a simple beam search algorithm for searching the Viterbi BTG alignments. Lardilleux et al. (2012) propose to binary segment the alignment matrix recursively to compute BTG-like alignments based on word level association scores but have not reported the alignment performance independently. Cherry and Lin (2007) presented a

phrasal BTG to the joint phrasal translation model and reported the results of the word alignment evaluation experiment. Neubig, Watanabe, Sumita, Mori, and Kawahara (2011) incorporated a non-parametric Bayesian method with joint phrase alignment and extraction model. Given it is not able to explain many translation phenomena that extracting phrase using word-to-word alignment as a pivot, Cohn and Haffari (2013) proposed to learn the BTG phrasal model using a recursive Bayesian method. Kamigaito, Tamura, Takamura, Okumura, and Sumita (2016) modified the bidirectional agreement constraints and applied a more complex version (BTG-style agreement) to train the BTG model jointly. Differing from the previous works, in this paper, we propose a BTG-forest-based word alignment as the heuristic to explore probable alignment points in alignment matrix. We will discuss the details in Section 3.

2.3 Hierarchical sub-sentential alignment and *Ncut*

Hierarchical sub-sentential alignment (HSSA) (Lardilleux et al. 2012) was first introduced as a complement for `Anymalign`[3]. Given the soft alignment matrix built using the parameters output by `Anymalign`, the HSSA method takes all the cells in the soft alignment matrix into consideration and relies on a precise criterion to determine a good partition in a similar way as image segmentation. HSSA makes use of an unsupervised clustering criterion called *normalized cuts* (Shi and Malik 2000; Zha, He, Ding, Simon, and Gu 2001), or *Ncut* for short, to recursively segment the matrix into two parts. During the segmentation processing, HSSA is supervised by the BTG constraint to decide the search scope of next level on the diagonal (rep: anti-diagonal) corresponding exactly to the case of *straight* (rep: *inverted*.) HSSA terminates when all words in the source and target sentences are aligned and generate symmetric alignments at the same time.

Consider a source phrase $A : X\overline{X}$ and a target phrase $B : Y\overline{Y}$, which can be split at source index i and j in target side in a dichotomic way. The sub-spans X, $\overline{X}$ in source side are corresponding to Y, $\overline{Y}$ or $\overline{Y}$, Y. If we record the segmentation option as $\gamma \in \{0, 1, 2\}$ (diagonal, anti-diagonal and terminal), the association within groups $asso(A, B)$ and the risk of cutting *cut* at point (i, j) is defined by the underlying formula followed the definition of (Shi and Malik 2000):

$$asso(A, B) = \sum_{f \in A} \sum_{e \in B} w(f, e) \tag{6}$$

[3] https://anymalign.limsi.fr/

where $w(f, e)$ stands for the weighted score that f is aligned with e.

$$cut(i,j|\gamma=0) = \underbrace{asso(X,\overline{Y})}_{\text{left}} + \underbrace{asso(\overline{X},Y)}_{\text{right}}, \qquad straight \tag{7}$$

$$cut(i,j|\gamma=1) = \underbrace{asso(X,Y)}_{\text{left}} + \underbrace{asso(\overline{X},\overline{Y})}_{\text{right}}, \qquad inverted \tag{8}$$

The optimal bi-partitioning of such a matrix (graph) is the one that minimizes this *cut* value. However, the minimum cut criterion favors cutting small sets of isolated nodes in the graph, which is counterintuitive, so (Shi and Malik 2000) propose *Ncut* as a measure for total normalized association within groups for a given partition. In our case, *Ncut* can be defined as:

$$Ncut(i,j|\gamma) = \frac{cut(i,j|\gamma)}{cut(i,j|\gamma)+2\times cut_{\text{left}}(i,j|\bar{\gamma})} + \frac{cut(i,j|\bar{\gamma})}{cut(i,j|\bar{\gamma})+2\times cut_{\text{right}}(i,j|\bar{\gamma})}$$

Each possible splitting point (i, j) in the matrix divides the parent matrix into 4 sub-matrices (XY, $X\overline{Y}$, $\overline{X}Y$, $\overline{XY}$). Either the two sub-matrices on the diagonal (XY, $\overline{XY}$) or the two sub-matrices on the anti-diagonal ($X\overline{Y}$, $\overline{X}Y$) will be explored recursively on the next layer, referring to γ equals 0 or 1. Hence, recursive segmentation eventually consists in determining these indices (i, j) which minimize $Ncut(i, j|\gamma)$ or $Ncut(i, j|\bar{\gamma})$ over all possible indices. Hence, the criterion *Ncut* that we seek in our recursive partition algorithm, minimizing the disassociation between the blocks unaligned and maximizing the association within the blocks aligned. Since the the worst case time complexity of top-down HSSA is cubic $O(m \times n \times \min(m, n))$ and the best case is $O(m \times n \times \log \min(m, n))$ in the length of the input sentence pair, it is faster than the original BTG method $O(n^6)$.

3 Joint alignment and symmetrization model

In this paper, we propose a novel joint model for the unsupervised training of a word aligner by applying IBM models with HSSA. The next sections show how to combine various IBM models with HSSA method to obtain symmetric word alignments.

3.1 Building the soft alignment matrices

Given a source sentence $\mathbf{f} = f_1^m = f_1, \ldots, f_i, \ldots, f_m$ and a target sentence $\mathbf{e} = e_1^n = e_1, \ldots, e_j, \ldots, e_n$, alignment associations between a source sentence $\mathbf{f}$ and a target sentence $\mathbf{e}$ can be regarded as a contingency matrix (Matusov, Zens, and Ney 2004; Moore 2005; Liu, Xia,

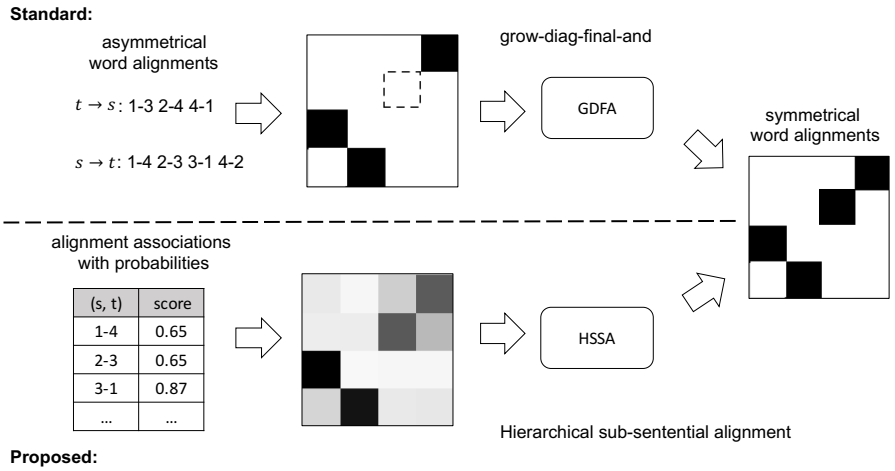

Fig. 1 Comparison of standard bidirectional word alignment pipeline and our proposed joint method pipeline.

Xiao, and Liu 2009), noted as $\mathcal{M}$, in which m is the length of source sentence in words and n for target side. For example, in (Liu et al. 2009), a weighted matrix was employed for extracting more candidates of phrase pairs, which consists of the cells that corresponding to a arbitrary word pair. Each of these cells has been assigned with a probability score to measure the confidence of aligning the two words. Following this definition, we define a function w which measuring the probability of alignment between any source and target word pair (f_i, e_j) but use it for BTG parsing. The symmetric alignment between word f_i and e_j presents a weighted cell (i, j) in the alignment matrix.

Let $\mathcal{M}$ be such a soft alignment matrix (a weighted adjacent matrix in fact) which is used to represent the graph of the sentence pair, in which each pair of words in such a graph is connected, with weight on the edges between the nodes. Formally, when given a source word f and a target word e, we define a soft link $l = (f, e)$ to exist if f and e are probable translation. There are plenty of ways to define the weight of $l(f, e)$ and perhaps the most simple way to inference is using the posterior IBM model 1 probabilities. Since the sparsity of the data for counting, Laplace smoothing is used here to handle the unseen alignments and remove the small values, with assigned a smoothing parameter $p_0 = 10^{-4}$ and force $l(f, e) >= p_0$.

$$l(f,e) = \begin{cases} p_0 & \text{if } l = \varepsilon \\ \sqrt{l(f|e) \times l(e|f)} & \text{otherwise} \end{cases} \tag{9}$$

After computed the posterior probabilities with the EM algorithm, the symmetrical score $l(f, e)$ can be obtained by taking the geometric mean of the lexical translation probabilities in both directions $l(f|e)$ and $l(e|f)$ and we can approximate $w(f_i, e_j)$ by $l(f, e)$. As our task is not only aiming at better results but also at saving time in alignment computation, the simpler the faster. In (Haghighi et al. 2009; Liu, Liu, and Lin 2010), more features to achieve better results. However, their work focuses on supervised BTG models, while our joint model is purely unsupervised. Other works like training with agreement (Liang et al. 2006) perhaps give better initial parameters but also require more time for training and result in a computationally expensive process. Moore (2005) pointed that IBM model 1 has many disadvantages, it is either too sensitive to rare words or over-weights frequent words (like function words). For this reason, incorporating variational Bayes (VB) (Riley and Gildea 2012) into our model is necessary. We implemented VB for the translation probabilities in M step and assume the distribution of the target vocabulary to be a Dirichlet distribution, with a symmetric Dirichlet prior as α so that we have

$$l(e|f) \sim \text{Dirichlet}(\alpha) \tag{10}$$

Here, we assume the prior $\alpha = 0.01$. As an alternative, other probability models for estimation can also be used, and we can change the original IBM model with the max operator (as (Zhang and Gildea 2005)). Given the asymmetrical alignments in both directions, it is easy to reestimate the Viterbi probabilities. This reestimation processing can significantly reduce the size of the trained models. Some heuristic like *grow-diag-final-and* can also be applied before re-estimating. We found that it does yield better results in our alignment experiments.

Until now, we have not discussed any effect of position information yet. As we known, in IBM model 2 and HMM model, for a given word pair (f_i, e_j), position information (i, j) is a very important term. In addition, our joint model takes the position information as a complementary component. It is expected to work under the condition that the sentence pair contains multiple possible word translation pairs for identical (f, e), i.e., the case when $(f_i, e_j) = (f_{i'}, e_{j'})$. An effective solution is to define the score w which connecting two nodes f_i and e_j, as the product of a feature translation term (translation probability) and spatial proximity term (relative position similarity) as in (Shi and Malik 2000):

$$w(f_i, e_j) = e^{\frac{\theta(f_i, e_j)}{\sigma_\theta}} \times \begin{cases} p_0 & \text{otherwise} \\ e^{\frac{\delta(i,j,m,n)}{\sigma_\delta}} & \text{if } h(i,j,m,n) < r \end{cases} \tag{11}$$

where w measures the strength of the translation link between a source word and a target word

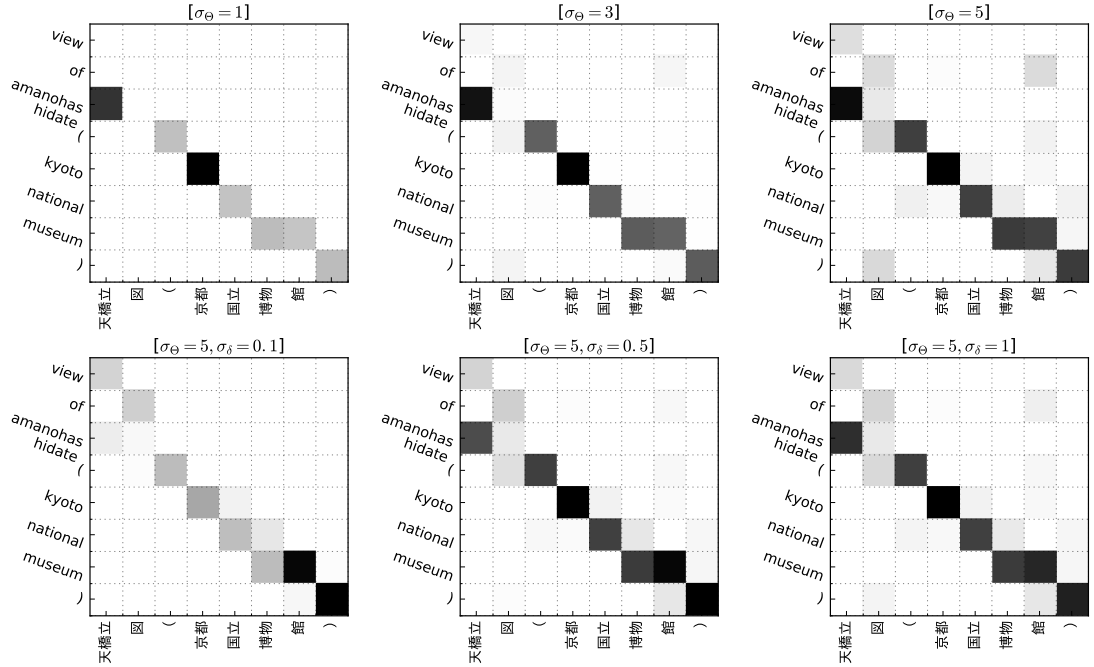

Fig. 2 Grey-scale map of soft alignment matrix. Three sub-graphs without spatial proximity term (top), three with spatial proximity term (bottom).

(f_i, e_j). $\theta(f_i, e_j) = \log(l(i,j))$ is a translation model and $\delta(i,i,m,n) = \log(1 - h(i,j,m,n))$ is a distortion model. r is a threshold, which depends on language. It usually should take a value in the range $[0.5, 0.9]$. σ_θ and σ_δ are hyper-parameters. To compute the value of $h(i,j,m,n)$, we assume $h(i,j,m,n) = |i/m - j/n|$. Although this is not necessary but in our case, we adjust values to a specified range $w(j,i) \in [{p_0}^2, 1)$. Since *Ncut* is a normalized score, it does not require any normalization term. The hyper-parameters σ_θ and σ_δ are fixed at the beginning of experimentation by maximizing the *Recall* in the preliminary experiments. Since tuning such hyper-parameters for each language pairs is expensive, we just show that it is possible to obtain reasonable results without careful tuning. Figure 2 shows how matrices change when changing these hyper-parameters. Since we also want to address the influence of imported distortion sub-model, we performed the comparable experiment. The results are found in Table 5 and Table 6.

Algorithm 1 Top-Down Parsing

function TopDownParsing($< \mathbf{f}^K, \mathbf{e}^K >, \mathbf{m}$) ▷ Where $\mathbf{f}$ - source sentence, $\mathbf{e}$ - target sentence, τ - probability model, K - corpus size.
  $\mathcal{M} \leftarrow$ InitializeSoftMatrix($< \mathbf{f}, \mathbf{e} >, \tau$) ▷ build soft alignment matrix
  $S_0 \leftarrow$ {InitializeHypo($0, |\mathbf{f}|, 0, |\mathbf{e}|$)}
  $S_{final} \leftarrow \{\}$
  for $i = 0$ to $\min(|\mathbf{f}|, |\mathbf{e}|)$ **do**
    if $S_i = \{\}$ **then** ▷ no new hypothesis
      continue
    end if
    for all *hypo* $\in$ ExpandHypos($S_i, \mathcal{M}$) **do** ▷ expand current hypothesis
      $S_{i+1} \leftarrow S_{i+1} \cup hypo$
      if *hypo*.coverage $=$ *true* **then** ▷ check whether all words have been aligned
        $S_{final} \leftarrow S_{final} \cup hypo$
      end if
    end for
    $S_{i+1} \leftarrow top_k(S)$ ▷ prune the beam, return the top-k hypotheses
  end for
  $\tilde{D} = \underset{D \in S_{\text{final}}}{\text{argmax}}\ Score(D_{F_{avg}}|\mathcal{M})$ ▷ select the best-1 as final symmetrical alignments
  return $\tilde{D}$
end function

3.2 Top-down Parsing and Reranking

In (Lardilleux et al. 2012), 1-best parsing was employed to find the optimal *Ncut* at each layer. However, experimentally, we found that the strategy of 1-best parsing used in the original HSSA does not generate the best global derivation. Hence, we define a scoring function $Score()$ aiming to find the best derivation $\tilde{D}$ with the minimal value:

$$\tilde{D} = \underset{D}{\text{argmin}}\ Score(D_{Ncut}|\mathbf{f}, \mathbf{e}) = \underset{D}{\text{argmin}} Score(D_{Ncut}|\mathcal{M}) \tag{12}$$

D_{Ncut} stands for the parser derivation obtained according to $Ncut$. Due to an equivalent way of writing $Ncut$ is the function of arithmetic mean of F-measure, as described in (Wang and Lepage 2016b), in sub-matrices (X, Y) and $(\bar{X}, \bar{Y})$, notes F_{avg}:

$$Ncut(i, j|\gamma) \propto 1 - \frac{F_1(X,Y)+F_1(\overline{X},\overline{Y})}{2} = 1 - F_{avg} \tag{13}$$

Where we compute the F_1 score as:

$$F_1(X, Y) = 1 - \frac{asso(\overline{X},Y)+asso(X,\overline{Y})}{2\times asso(X,Y)+asso(\overline{X},Y)+asso(X,\overline{Y})} \tag{14}$$

Algorithm 2 Expand Hypotheses

function EXPANDHYPOS($S_i, \mathcal{M}$) ▷ Where S_i-previous stack, $\mathcal{M}$-soft matrix
  $S \leftarrow \{\}$
  while S_i **do**
    $hypo \leftarrow S_i.pop()$ ▷ pop the item from the previous stack
    for all $block \in hypo.blocks$ **do**
      $S' \leftarrow \{\}$
      $\{i_1, j_1, i_3, j_3\} \leftarrow block$ ▷ four corner indices in sub-matrix
      for all $\{i_2, j_2\} \in \mathcal{M}_{(i_1,j_1,i_3,j_3)}$ **do** ▷ search for all possible partitions
        for $\gamma \in [0, 1]$ **do**
          $score = \text{ComputeScore}(\gamma, i_2, j_2, \mathcal{M}_{(i_1,j_1,i_3,j_3)})$
          if $\gamma = 0$ **then** ▷ straight case in BTG
            $block_1, block_2 \leftarrow \text{DiagonalMatrices}(\mathcal{M}_{(i_1,j_1,i_3,j_3)}, i_2, j_2)$
          else ▷ inverted case in BTG
            $block_1, block_2 \leftarrow \text{AntiDiagonalMatrices}(\mathcal{M}_{(i_1,j_1,i_3,j_3)}, i_2, j_2)$
          end if
          $S \leftarrow S \cup \{i_2, j_2, score, \gamma, block_1, block_2\}$
        end for
      end for
    end for
    for all $\{i_2, j_2, score, \gamma\, block_0, block_1\} \in top_k(S')$ **do**
      $new\_hypo = hypo.update([block_1, block_2], i_2, j_2, \gamma, score)$ ▷ expand current hypothesis
      $S \leftarrow S \cup new\_hypo$ ▷ add new hypothsis into new stack
    end for
  end while
  return S
end function

With this interpretation, minimizing *Ncut* is equivalent to maximizing the value F_{avg}. Intuitively, it suffices to replace *Ncut* with F_{avg} to derive the following formula, the probability of a parsing tree or the probability of a sequence of derivation $D = \{d_0, \ldots, d_K\}$ and best word alignment $\hat{\mathbf{a}}$ based on the derivation $\tilde{D}$ can be defined as,

$$\tilde{D} = \underset{D}{\text{argmax}}\ Score(D_{F_{avg}}|\mathcal{M}) = \underset{d_k \in D}{\text{argmax}} \prod_{k=0}^{K} F_{avg}(d_k) \tag{15}$$

$$\hat{\mathbf{a}} = Proj(\tilde{D}) \tag{16}$$

The use of *Ncut* or F_{avg} is equivalent. Hence the derivation either $D_{F_{avg}}$ or D_{Ncut} obtained should be same. $Proj()$ is a projection function A projection function which produces the final

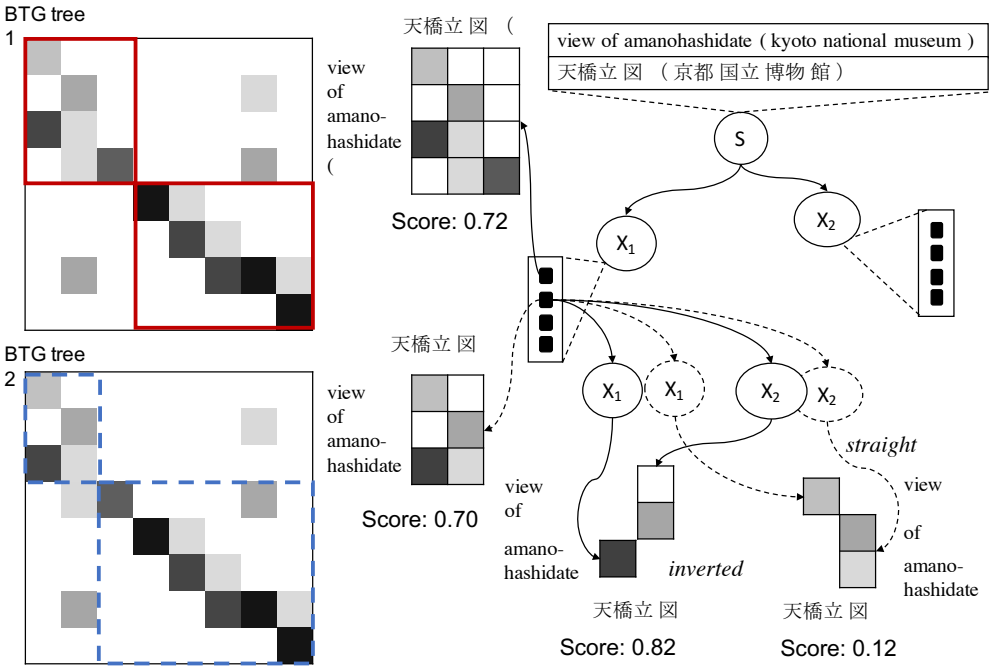

Fig. 3 Hierarchical sub-sentential alignment with beam search as Top-down BTG forest parsing.

word-to-word alignment from the leaves of the BTG parser tree. Let d_k denote the operation of derivation at step k, in which d_k is defined as a triple $< i, j, \gamma >$. During parsing, where i stands for the index of the splitting point on the source side and j stands for the index of the splitting point on the target side. We implement our top-down parser with beam search see Figure 3. The incremental top-down BTG parsing algorithm used is presented in **Algorithm** 1.

We consider that the incremental parser has a parser hypothesis at each step. We define the hypothesis as a four-tuple $\langle P, Q, v, c \rangle$. P is a stack of the unsolved blocks. Q is a list of the previous derivations $\{d_0, \ldots, d_{k-1}\}$. A block denoted by $([i_0, i_1), [j_0, j_1))$ covers the source words from f_{i_0} to f_{i_1-1} and the target words from e_{j_0} to e_{j_1-1}. v records the current score. We set c to *true* on termination (stack P is empty). At the beginning, the initial hypothesis contains only a block which covers all the words in the source and target sentences. Then, we split the block in each step, and decide the node type (*straight* or *inverted*) when the splitting point is determined according to the defined score function. $top_k(S)$ returns the first k-th hypotheses from stack S in terms of their scores. The computational complexity of the top-down parsing algorithm is $O(n \times m \times k \times \log \min(m, n))$ for sentence lengths m, n, beam size k. $\log \min(m, n)$ stands for the parsing depth. For each iteration, each hypothesis in the history will be used to generate

Table 1 Statistics of Hansard Corpus and KFTT Corpus for alignment evaluation in our experiment (M: million, K: thousand).

	# of	Hansard Corpus (en-fr)	KFTT Corpus (en-ja)
train	lines	1,130,551	331,109
	tokens	20.02 M/23.61 M	5.97 M/6.12 M
	types	68.0 K/86.6 K	138 K/114 K
test	lines	447	1,235
	tokens	7,020/7,761	30,822/34,366
	types	1,732/1,943	4,990/4,908

Table 2 Statistics of data from WMT08 Shared tasks, KFTT Corpus and Europarl Corpus v7 for translation evaluation used in our experiments.

		WMT 08		KFTT	Europarl v7	
	# of	en-fr	en-de	en-ja	es-pt	en-fr
train	lines	1.28 M	1.26 M	330 K	183.3 K	183.3 K
	tokens	32.2 M/33.5 M	29.3 M/31.6 M	5.91 M/6.09 M	5.27 M/5.02 M	4.95 M/5.23 M
dev	lines	2,000	2,000	1,166	1,000	1,000
	tokens	53.1 K/55.1 K	5.31 K/48.8 K	24.3 K/26.8 K	36.4 K/34.5 K	35.3 K/36.3 K
test	lines	2,000	2,000	1,160	2,000	2,000
	tokens	54.3 K/56.2 K	26.7 K/28.5 K	5.43 K/5.02 K	59.6 K/58.8 K	57.6 K/61.8 K

new hypotheses as shown in **Algorithm** 2. **Algorithm** 1 terminates when no new hypothesis is generated or has reached the maximum number of iterations $\min(m, n)$.

To reduce the time complexity in calculating the value of $asso(A, B)$, we make use of a specialized data structure for fast computation. For each built soft alignment matrix, a summed area table (SAT) is created for fast calculating the summation of cells in the corresponding soft alignment matrix $\mathcal{M}(m, n)$. This preprocessing step is to build a new $(m+1, n+1)$ matrix $\mathcal{M}'$, where each entry is the sum of the sub-matrix to the upper-left of that entry. Any arbitrary sub-matrix sum can be calculated by looking up and combining only 4 entries in the SAT. For instance, assume that A, B extends from point (i_0, j_0) to point (i_1, j_1). We have,

$$asso(A, B) = \sum_{i=i_0+1}^{i_1} \sum_{=j_0+1}^{j_1} w(i, j) = \mathcal{M}'(i_1, j_1) - \mathcal{M}'(i_0, j_1) - \mathcal{M}'(i_1, j_0) + \mathcal{M}'(i_0, j_0) \quad (17)$$

The time complexity here is reduced from $O(m \times n)$ to $O(1)$ when calculating the summation of all cells in the block of (A, B).

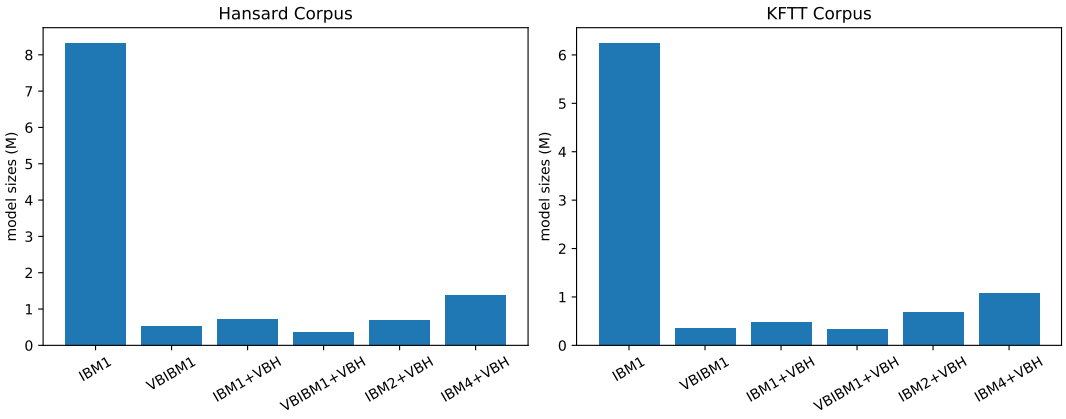

Fig. 4 Comparsion of model sizes (# of entries) with different initialization methods.

4 Experiments

4.1 Data

We first describe the data used in our experiments. The data sets for word alignment and translation tasks are from the different corpora. For word alignment subtasks, we use the Hansard Corpus[4] from 2003 NAACL shared task (Mihalcea and Pedersen 2003) for English–French and KFTT Corpus[5] for English–Japanese. Table 1 gives some counts on the training set and test set for word alignment evaluation and the training set includes the test set.

For the translation task, we conduct experiments in several language pairs: English–French (en–fr), English–German (en–de), Spanish-Portuguese (es–pt), English–Japanese (en–ja) and Japanese–English (ja–en). The English–Japanese and Japanese–English subtasks use the KFTT corpus. For English–French, Spanish-Portuguese and English–German, we make use of four different corpora, two large corpora (English–French and English–German) from the WMT 2008 Shared Task[6] and two smaller corpora (English–French and Spanish-Portuguese) from the Europarl Corpus[7]. For translation evaluation, training, development, test sets are independent. Table 2 gives some statistics on the training, development and test sets.

[4] http://web.eecs.umich.edu/ mihalcea/wpt/index.html#resources
[5] http://www.phontron.com/kftt/
[6] http://www.statmt.org/wmt08/shared-task.html
[7] http://www.statmt.org/europarl/

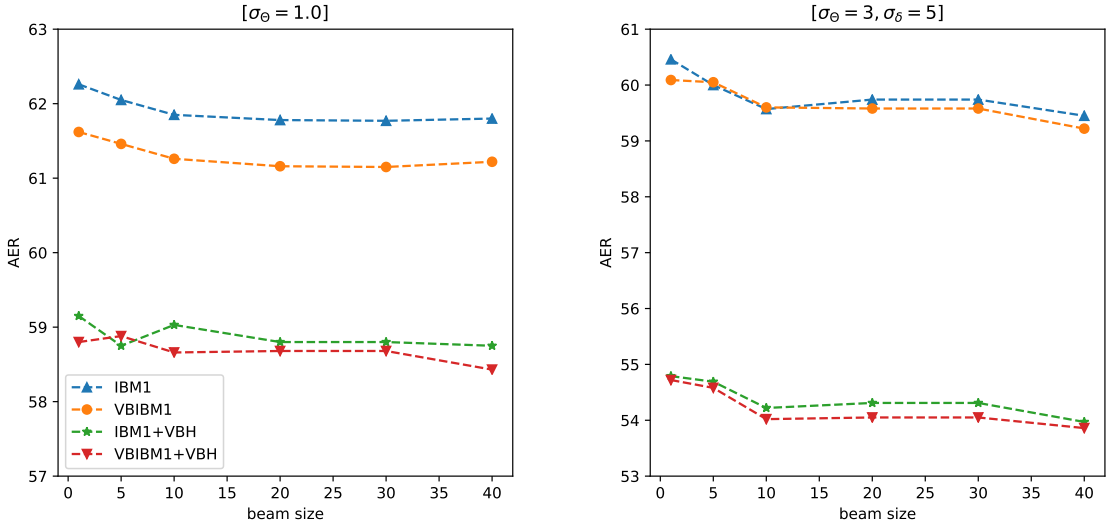

Fig. 5 Detemining the proper value for beam size acording to AER on the KFTT corpus. Larger beam sizes have lower AER. The default beam size in `Hieralign` is setted as 10 given the trade-off between accuracy and speed.

4.2 Experiment setup

For word alignment, we preprocess the data by lowercasing. In the case of `GIZA++` and `fast_align`, we train word alignments in both directions with the default settings, likes the standard bootstrap for IBM model 4 alignment in `GIZA++` as $\{1^5H^53^34^3\}^8$ and 5 iterations in `fast_align`. We then symmetrize the word alignments using *grow-diag-final-and* and evaluate with the final obtained alignments. For our implementation, `Hieralign`[9], in order to limit the run-time to that of `fast_align`, we apply 5 iterations of parameter updates by EM with IBM model 1. Since reestimating the Viterbi probability is very fast when an initial word alignment is given for reference, we also employ various methods to compute $l(f, e)$, like IBM1, IBM1 using variational Bayes (VBIBM1), IBM1 Viterbi with heuristic (IBM1+VBH), IBM1 using variational Bayes Viterbi with heuristic (VBIBM1+VBH), IBM2 Viterbi with heuristic (IBM2+VBH)[10], IBM4 Viterbi with heuristic (IBM4+VBH), etc.. Figure 4 illustrates the size of obtained models using different initialization methods.

Some comparison with other BTG alignment methods is necessary to confirm the advantages

[8] $1^5H^53^34^3$ denotes 5 iterations for IBM model 1, 5 iterations for HMM model , 3 iterations for IBM model 3, 3 iterations for IBM model 4

[9] https://github.com/wang-h/Hieralign

[10] We use the variation of IBM model 2 (Dyer et al. 2013), which is fast and relatively simple model.

Table 3 Wall-clock time (minutes:seconds) required to obtain the symmetric alignments. The total time also includes the time for symmetrizing asymmetric alignments for `GIZA++` and `fast_align` with the standard symmetrization heuristic: *grow-diag-final-and*. "online" means that Hieralign can run in the "online" mode (a trained model is given). This provides an "online" service: when new sentences are input, it directly runs the second step to output word alignments.

	Hansard Corpus	KFTT Corpus
`pialign` + `pialign:itgstats.pl`	≫240:00	199:46
`GIZA++` + `Moses:train-model.perl-3`	228:36	51:38
`fast_align` + `fast_align:atools`	6:58	1:40
`Hieralign` (beamsize=1)	6:43	1:33
`Hieralign` (beamsize=10)	8:36	2:03
`Hieralign` (online, beamsize=1)	1:36	0:22
`Hieralign` (online, beamsize=10)	3:21	1:13

Table 4 Effect of beam size on alignment quality on Japanese-English data in terms of match number, precision, recall, AER and running time of HSSA. "Test" means the total count of word-to-word alignments in the file output by each method. "Matches" means the count of "true positives" alignment found in the output of each method.

	Test	Matches	Prec	Rec	AER	CPU time (sent./sec.)
Ref	33,377					
Best-1	43,017	15,879	36.91	47.57	58.43	3,389
Best-5	43,052	15,969	37.09	47.84	58.21	1,602
Best-10	43,150	16,030	37.15	48.03	58.11	993
Best-20	43,137	**16,036**	**37.17**	**48.05**	**58.08**	662
Best-40	43,137	16,007	37.11	47.96	58.16	336

IBM1+VBH with beam search ($\sigma_\theta = 1$).

of our proposed method. For this consideration, we use another open-sourced BTG-based word aligner, `pialign` (Neubig et al. 2011) [11], which uses hierarchical BTG models (hier). For `pialign`, we run it with 8 threads and train the model with batch size 40 and only taking 1 sample during parameter inference. We extract phrases directly from the word-to-word alignment (1-to-many, many-to-1 and many-to-many) with traditional heuristic (Koehn et al. 2003) for translation. Our BTG-style parsing is on the basic of top-down bi-partitioning. This is different from previous CKY-based models (Xiong et al. 2010; Neubig et al. 2011). In the work (Neubig et al. 2011), they employed another heuristic in phrase extraction rather the standard one (Koehn et al. 2003). To make our work traceable, we compare the phrase-table size extracted with the simple

[11] http://www.phontron.com/pialign/

heuristic-based phrase extraction (Koehn, Axelrod, Birch, Callison-Burch, Osborne, Talbot, and White 2005) with the model-based phrase extraction approach in (Neubig et al. 2011). For scoring phrases, four features are used in each phrase table for all experiments: the conditional phrase probabilities in both directions, $p(f|e)$ and $p(e|f)$, lexical weighting probabilities in both directions, $p_{lex}(f|e)$ and $p_{lex}(e|f)$ (Koehn et al. 2003)[12].

For translation evaluation in all experiments, the phrase-based SMT systems are standard statistical machine translation systems built by using the `Moses`[13] toolkit (Koehn, Hoang, Birch, Callison-Burch, Federico, Bertoldi, Cowan, Shen, Moran, Zens, et al. 2007) with Minimum Error Rate Training (Och 2003) and a 5-gram language model learnt using `KenLM` (Heafield 2011). We use the training set for training translation, lexical reordering, and target language models, the development set for tuning the parameters of the log-linear model in decoding and the test set for evaluation. For phrase extraction, we employ the traditional heuristic (Koehn et al. 2003) for all methods used in experiment. In our experiment, the maximum length of phrases entered into phrase table is limited to 7. Before that, *grow-diag-final-and* was used for `GIZA++`, `fast_align`. Specially for `pialign`, we make use of `pialign:itgstats.pl` [14] to extract the final word alignments used for phrase extraction. The baselines are the PB-SMT systems with the default distortion limit set to 6. We also filter the data to remove the long sentences (more than 100 words) from the training set. We do the same preprocessing (lowercasing and tokenization) using the scripts provided in `Moses` [15] and baseline processing as WAT[16] for English-Japanese and Japanese-English. Finally, we conduct translation experiments and compare `Hieralign`, `fast_align`, `pialign` and `GIZA++` for performance and time.

4.3 Alignment and speed evaluation

We evaluate the performance of our proposed method and report the performance of various alignment methods in terms of precision, recall and alignment error rate (AER) as defined in (Och and Ney 2003). Figure 5 shows that we found the alignment scores improve until a beam size of 20. Larger sizes do not help. In a real situation, given the result in Table 4 and Figure 5 shown, a beam size of 10 should give a reasonable trade-off between time and accuracy (achieved comparable running time and accuracy with `fast_align`). Since our implementation does not require any

[12] In the experiment reported in (Neubig et al. 2011), three additional features are employed (the joint probability of the phrase, the average posterior probability of a span and the uniform phrase penalty)
[13] http://www.statmt.org/moses/
[14] https://github.com/neubig/pialign/tree/master/script
[15] https://github.com/moses-smt/mosesdecoder/tree/master/scripts/tokenizer
[16] http://lotus.kuee.kyoto-u.ac.jp/WAT/WAT2015/baseline/tools.html

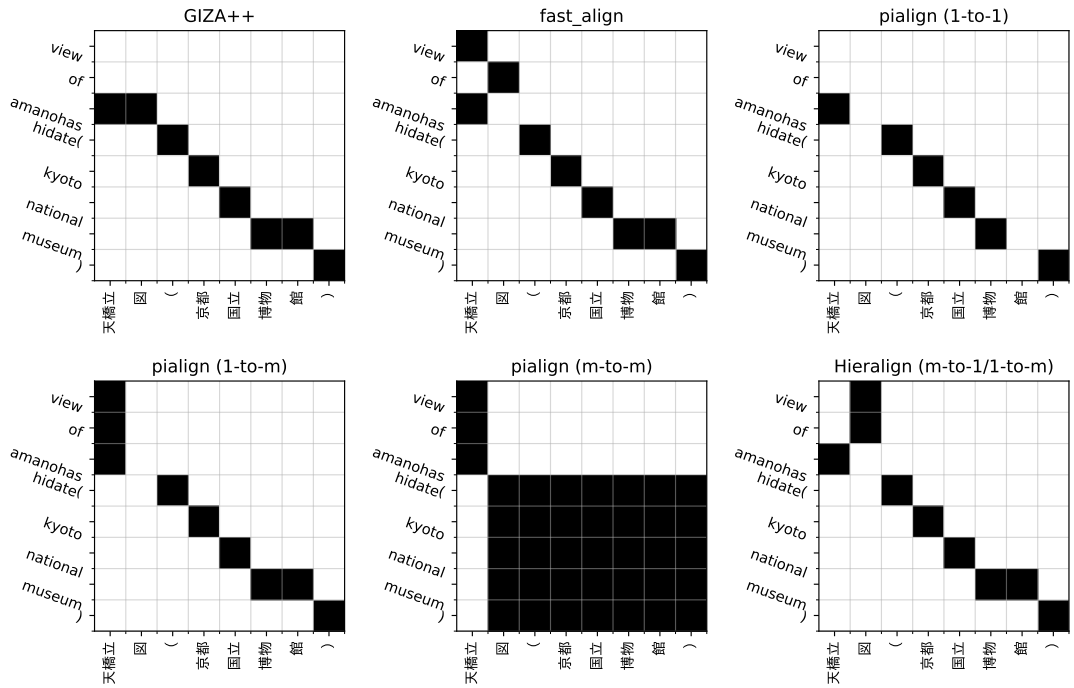

Fig. 6 Example of alignment matrices output by `GIZA++` (+GDFA), `fast_align` (+GDFA) , `pialign` and `Hieralign`. `pialign` only outputs asymmetric block alignments (1-to-m). `Hieralign` outputs block alignments which is symmetrical (1-to-m/m-to-1) .

extra information except the word-to-word translation probabilities, it can be regarded as an online word aligner.

Table 5 gives the performance of various alignment profiles on the basis of human annotated alignment data provided with the KFTT Corpus and Hansard Corpus. The first and second lines show the alignment difference using `GIZA++` + *grow-diag-final-and* and `fast_align` + *grow-diag-final-and*. From the table, we found both `GIZA++` and `fast_align` output less alignments than `Hieralign`. `Hieralign` tends to output more matches than `fast_align` but not as much as `GIZA++` from the point view of matching alignments and recall against the reference, which we can explain the reason may be: although `Hieralign` cannot explore some alignments because of the limitation of BTG constraints, when we force it to align each source and target word, it is prone to group the neighbouring alignments aggressively into the same block. This strategy makes it to output more alignments. AER and precision are behind `fast_align`, even more than `GIZA++`. However, Fraser and Marcu (2007), Ganchev, Graca, and Taskar (2008) question

Table 5 Evaluation of alignment results on Hansard Corpus and KFTT Corpus using various configuration of `GIZA++` (+GDFA), `Fast_align` (+GDFA), `pialign` and `Hieralign`.

	Hansard (French-English)					KFTT (Japanese-English)				
	Test	Prec	Rec	AER	Matches	Test	Prec	Rec	AER	Matches
`fast_align`	7,845	80.66	92.62	15.27	6,328	25,368	55.49	42.17	52.08	14,076
`GIZA++`	7,709	87.99	96.14	9.21	6,783	31,342	59.48	55.85	42.39	18,641
`pialign`										
1-to-1	4,341	**95.51**	80.01	**11.96**	4,146	16,705	**74.93**	37.50	50.01	12,517
1-to-many	6,648	84.42	85.14	15.31	5,612	30,386	55.36	50.40	**47.24**	16,821
many-to-many	16,295	51.14	**94.28**	40.29	**8,334**	57,053	33.99	**58.11**	57.11	**19,394**
`Hieralign: 1-to-many/many-to-1` ($\sigma_\theta = 1$)										
IBM1	8,979	65.85	87.10	27.56	5,913	42,317	33.46	42.42	62.59	14,160
VBIBM1	9,016	68.23	88.86	25.39	6,152	42,465	34.39	43.76	61.49	14,605
IBM1+VBH	9,037	68.58	87.74	25.50	6,198	43,150	37.15	**48.03**	**58.11**	**16,030**
VBIBM1+VBH	9,012	68.36	88.81	25.31	6,161	42,974	**37.20**	47.90	58.12	15,986
IBM2+VBH	9,008	70.32	89.57	23.72	6,334	44,123	36.01	47.60	59.00	15,887
IBM4+VBH	8,958	**72.91**	**89.97**	**21.79**	**6,531**	43,257	35.16	45.57	60.31	15,209
`Hieralign: 1-to-many/many-to-1` ($\sigma_\theta = 3, \sigma_\delta = 5$)										
IBM1	8,542	70.52	87.91	23.90	6,024	39,896	36.66	43.82	60.08	14,626
VBIBM1	8,540	71.02	87.72	23.62	6,008	39,107	37.22	43.82	59.75	14,627
IBM1+VBH	8,634	73.56	89.87	21.24	6,351	40,260	**42.43**	**51.18**	**53.60**	**17,082**
VBIBM1+VBH	8,638	74.09	90.56	20.66	6,400	40,214	42.10	50.72	53.99	16,929
IBM2+VBH	8,667	74.93	90.81	20.02	6,494	40,043	40.69	48.82	55.61	16,295
IBM4+VBH	8,668	**77.77**	**91.75**	**17.79**	**6,757**	40,543	40.31	48.96	55.78	16,343

the link between this word alignment quality metrics and translation results. There is no proof that improvements in alignment quality metrics lead to improvements in phrase-based machine translation performance. In other words, a lower AER does not imply a better translation accuracy, and we will show this in the following discussion. When sampling the alignment results, we found that the output of the proposed joint method usually generates more alignments against the reference. From this table, we can also draw the conclusion that adding the distortion feature slightly improves the alignment results. Figure 6 plots the different alignments output by different methods.

4.4 Translation evaluation

Since we proposed to solve the problem of discontiguity in phrase extraction because of discontiguous alignment. We force the word alignment at least 1-1, which should generate fewer entries in the translation tables. We also measured the sizes of the translation tables obtained. Figure 7 shows the comparison of phrase tables. We compare the translations produced by our

Table 6 BLEU and RIBES scores in translation experiments, † means significantly different with the `fast_align` baseline according to statistical significance tests ($p<0.05$).

	WMT08				KFTT				Europarl v7			
	en-fr		en-de		en-ja		ja-en		en-fr		es-pt	
	BLEU	RIBES	BLEU	RIBES	BLEU	RIBES	BLEU	RIBES	BLEU	RIBES	BLEU	RIBES
`fast_align`	26.59	76.65	19.61	70.02	21.32	68.10	17.67	65.71	54.10	91.14	49.50	90.79
`GIZA++`	**26.79**	**77.17**	**19.82**	**70.48**	22.57†	**68.79**	18.20	65.25	54.40	91.22	49.34	90.62
`pialign`												
1-to-1	26.39	76.79	19.10†	70.25	21.92	68.61	18.06	66.03	**54.71**	91.29	49.51	89.51
1-to-many	26.57	76.95	19.45	69.98	22.07	68.68	**18.40**	65.55	53.79	91.13	49.48	89.49
many-to-many	26.67	77.13	19.82	69.99	21.69	69.07	18.19	65.90	54.61	91.28	49.23	90.53
MOD	26.35	77.02	19.31	70.23	21.11	68.17	18.13	65.99	54.35	91.17	49.40	90.57
`Hieralign:` `1-to-many/many-to-1` ($\sigma_\theta = 1$)												
IBM1	26.05†	76.47	19.24	69.68	21.68	67.93	17.43	65.04	54.15	91.24	49.30	90.67
VBIBM1	26.23	76.71	19.58	70.12	22.10	68.36	17.42	64.67	54.24	91.25	49.59	90.79
IBM1+VBH	26.22	76.61	19.39	69.92	22.40†	68.56	17.70	64.93	53.83	91.14	49.15	90.65
VBIBM1+VBH	26.30	76.59	19.33	69.62	**22.69**†	67.94	17.60	66.06	53.86	91.26	49.35	91.21
IBM2+VBH	26.23	76.59	19.30	69.59	22.32†	68.11	17.44	**66.17**	53.52	91.06	49.54	90.76
IBM4+VBH	26.25	76.71	19.30	69.91	21.76	68.09	17.47	65.34	53.72	91.21	49.41	90.71
`Hieralign:` `1-to-many/many-to-1` ($\sigma_\theta = 3, \sigma_\delta = 5$)												
IBM1	26.13	76.70	19.18†	69.62	21.87	67.52	17.88	65.14	53.77	91.04	49.59	90.74
VBIBM1	26.30	76.97	19.43	69.92	21.31	67.35	17.58	64.83	54.32	91.24	49.61	90.75
IBM1+VBH	26.37	76.75	19.32	69.97	22.25	67.79	17.73	65.02	53.91	91.18	49.50	90.65
VBIBM1+VBH	26.55	76.86	19.55	69.90	22.57†	68.35	17.80	65.50	54.35	**91.31**	49.17	90.68
IBM2+VBH	26.25	76.54	19.56	70.00	22.44†	68.62	17.69	65.37	53.40	91.05	**49.80**	**90.80**
IBM4+VBH	26.22	76.58	19.65	70.03	22.34†	68.35	17.40	65.10	53.91	91.28	49.51	90.63

method to those produced by Vitebi-based aligner `GIZA++` (+GDFA), `fast_align` (+GDFA) and BTG-based aligner `pialign`. For the evaluation of machine translation accuracy, some standard automatic evaluation metrics have been used: BLEU (Papineni, Roukos, Ward, and Zhu 2002) and RIBES (Isozaki, Hirao, Duh, Sudoh, and Tsukada 2010). In order to compare the performance between MT systems, we also apply bootstrap re-sampling method as described in (Koehn 2004). The results in Table 5 show that the final translation scores are approximately the same. There is insignificant difference on the final results of machine translation when using the alignments output by the proposed method and those output by `GIZA++` or `fast_align`. It can be seen `GIZA++` and `fast_align` have comparable BLEU score but `pialign` (MOD) and our methods slightly behind the baseline on a very large size corpus. From Figure 7, we found that `pialign` (MOD) can significantly reduce the phrase table sizes while the traditional heuristic phrase extraction methods were not able to be observed. For our method `Hieralign`, we use (IBM 1+ VBH) to represent the remains, because tables obtained using `Hieralign` with differently initialized parameters have approximately the same size. The findings are not unexpected but are relevant, the size of the phrase table obtained from the alignments produced by `Hieralign`

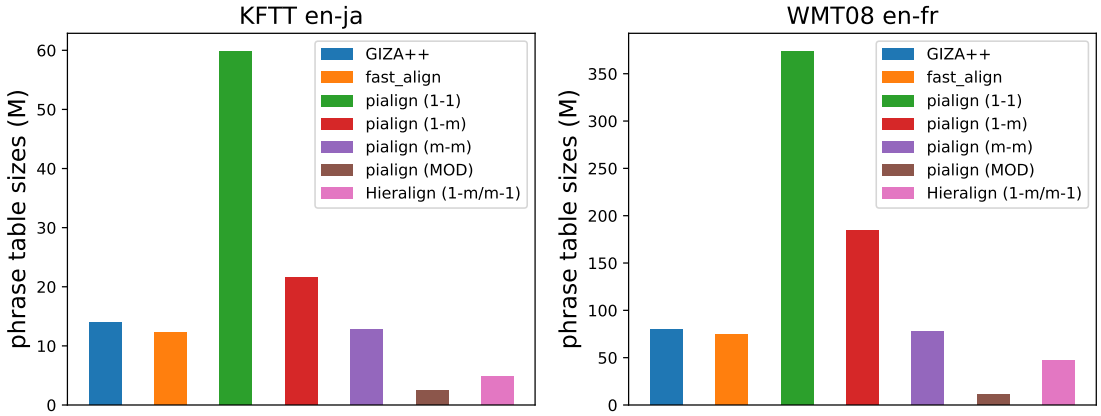

Fig. 7 Phrase-table sizes (# of entries) using `GIZA++` (+GDFA), `fast_align` (+GDFA), `pialign` and `Hieralign` with heuristic of phrase extraction (Koehn et al. 2003). For `pialign`, we also give the size of the table using model-based extraction (MOD) (Neubig et al. 2011).

is smaller by a third in comparison to those of the baseline in (en–fr, en–de, es–pt) (see Figure 7, right). In en–ja and ja–en, that is reduced by even two thirds (see Figure 7, left). `pialign` achieve a higher reduction ratio using model-based phrase extraction approach (MOD) than our method `pialign` but remember the training for `pialign` is really time-consuming (much more than 4 hours in our experiment while `Hieralign` is 2 minutes) and the translation results were not superior to our method.

5 Conclusion

In this paper, we proposed a novel joint hierarchical sub-sentential alignment method on the basis of IBM models. Our proposed method has several advantages: it is simple, fast and delivers small phrase tables. To summarize, when restricting the search space against all the possible permutations during aligning using BTG constraints, our BTG-forest-based alignment method can get better results than the previous best-1 hierarchical alignment method (Lardilleux et al. 2012). Experiments on word alignment showed that our proposed method align more words than other methods so as to largely reduced the phrase tables. Phrase-based machine translation systems using the smaller phrase tables produced by our method were able to achieve

comparable results, even better results on distant language pairs, like English-Japanese, with the traditional phrase extraction pipeline. The final experiment results show our method keeps the translation quality without significantly different while the speed of our methods was comparable with `fast_align`. As for future work, we will investigate better ways to build soft matrices using neural-based attention models, and we expect it deliver a better result. We will also investigate the possibility of using bilingual word-embedding vectors in our proposed framework.

Acknowledgement

This work is supported in part by China Scholarship Council (CSC) under the CSC Grant No. 201406890026. We also thank the anonymous reviewers for their insightful comments.

Reference

Brown, P., Cocke, J., Pietra, S. D., Pietra, V. D., Jelinek, F., Mercer, R., and Roossin, P. (1988). "A Statistical Approach to Language Translation." In *Proceedings of the 12th Conference on Computational Linguistics*, Vol. 1, pp. 71–76. Association for Computational Linguistics.

Brown, P. F., Pietra, V. J. D., Pietra, S. A. D., and Mercer, R. L. (1993). "The Mathematics of Statistical Machine Translation: Parameter Estimation." *Computational Linguistics*, **19** (2), pp. 263–311.

Cherry, C. and Lin, D. (2007). "Inversion transduction grammar for joint phrasal translation modeling." In *Proceedings of the NAACL-HLT 2007/AMTA Workshop on Syntax and Structure in Statistical Translation*, pp. 17–24. Association for Computational Linguistics.

Chiang, D. (2007). "Hierarchical Phrase-based Translation." *Computational Linguistics*, **33** (2), pp. 201–228.

Cohn, T. and Haffari, G. (2013). "An Infinite Hierarchical Bayesian Model of Phrasal Translation." In *ACL (1)*, pp. 780–790.

Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). "Maximum Likelihood from Incomplete Data via the EM Algorithm.". pp. 1–38.

Ding, C., Utiyama, M., and Sumita, E. (2015). "Improving fast_align by Reordering." In *Proceedings of the on Empirical Methods on Natural Language Processing*. Association for Computational Linguistics.

Dyer, C., Chahuneau, V., and Smith, N. A. (2013). "A Simple, Fast, and Effective Reparameterization of IBM Model 2." In *Proceedings for the NAACL*. Association for Computational

Linguistics.

Fraser, A. and Marcu, D. (2007). "Measuring Word Alignment Quality for Statistical Machine Translation." *Computational Linguistics*, **33** (3), pp. 293–303.

Ganchev, K., Graca, J. V., and Taskar, B. (2008). "Better Alignments= Better Translations?". p. 986. Association for Computational Linguistics.

Haghighi, A., Blitzer, J., DeNero, J., and Klein, D. (2009). "Better Word Alignments with Supervised ITG Models." In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2-Volume 2*, pp. 923–931. Association for Computational Linguistics.

Heafield, K. (2011). "KenLM: Faster and Smaller Language Model Queries." In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pp. 187–197. Association for Computational Linguistics.

Isozaki, H., Hirao, T., Duh, K., Sudoh, K., and Tsukada, H. (2010). "Automatic Evaluation of Translation Quality for Distant Language Pairs." In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pp. 944–952. Association for Computational Linguistics.

Kamigaito, H., Tamura, A., Takamura, H., Okumura, M., and Sumita, E. (2016). "Unsupervised Word Alignment by Agreement Under ITG Constraint." In *EMNLP*, pp. 1998–2004.

Koehn, P. (2004). "Statistical Significance Tests for Machine Translation Evaluation." In *EMNLP*, pp. 388–395. Citeseer.

Koehn, P., Axelrod, A., Birch, A., Callison-Burch, C., Osborne, M., Talbot, D., and White, M. (2005). "Edinburgh system description for the 2005 IWSLT speech translation evaluation." In *IWSLT*, pp. 68–75.

Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., et al. (2007). "Moses: Open Source Toolkit for Statistical Machine Translation." In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*, pp. 177–180. Association for Computational Linguistics.

Koehn, P., Och, F. J., and Marcu, D. (2003). "Statistical Phrase-based Translation." In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, Vol. 1, pp. 48–54. Association for Computational Linguistics.

Lardilleux, A., Yvon, F., and Lepage, Y. (2012). "Hierarchical Sub-sentential Alignment with Anymalign." In *Proceedings of the 16th Annual Conference of the European Association for Machine Translation*, pp. 279–286.

Li, P., Yang, L., and Sun, M. (2012). "A Beam Search Algorithm for ITG Word Alignment." In *Proceedings of the 24th International Conference on Computational Linguistics*, p. 673.

Liang, P., Taskar, B., and Klein, D. (2006). "Alignment by Agreement." In *Proceedings of the Main Conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, pp. 104–111. Association for Computational Linguistics.

Liu, Y., Liu, Q., and Lin, S. (2006). "Tree-to-string Alignment Template for Statistical Machine Translation." In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics*, pp. 609–616. Association for Computational Linguistics.

Liu, Y., Liu, Q., and Lin, S. (2010). "Discriminative Word Alignment by Linear Modeling." *Computational Linguistics*, **36** (3), pp. 303–339.

Liu, Y., Xia, T., Xiao, X., and Liu, Q. (2009). "Weighted Alignment Matrices for Statistical Machine Translation." In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, Vol. 2, pp. 1017–1026. Association for Computational Linguistics.

Matusov, E., Zens, R., and Ney, H. (2004). "Symmetric Word Alignments for Statistical Machine Translation." In *Proceedings of the 20th International Conference on Computational Linguistics*, p. 219. Association for Computational Linguistics.

Mermer, C. and Saraclar, M. (2011). "Bayesian Word Alignment for Statistical Machine Translation." In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers*, Vol. 2, pp. 182–187. Association for Computational Linguistics.

Mihalcea, R. and Pedersen, T. (2003). "An Evaluation Exercise for Word Alignment." In *Proceedings of the HLT-NAACL 2003 Workshop on Building and using Parallel Texts: Data Driven Machine Translation and Beyond*, Vol. 3, pp. 1–10. Association for Computational Linguistics.

Moore, R. C. (2005). "Association-based Bilingual Word Alignment." In *Proceedings of the ACL Workshop on Building and Using Parallel Texts*, pp. 1–8. Association for Computational Linguistics.

Neubig, G., Watanabe, T., Mori, S., and Kawahara, T. (2012). "Machine Translation without Words Through Substring Alignment." In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, pp. 165–174. Association for Computational Linguistics.

Neubig, G., Watanabe, T., Sumita, E., Mori, S., and Kawahara, T. (2011). "An Unsupervised

Model for Joint Phrase Alignment and Extraction." In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pp. 632–641, Portland, Oregon, USA. Association for Computational Linguistics.

Och, F. J. (2003). "Minimum Error Rate Training in Statistical Machine Translation." In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics*, Vol. 1, pp. 160–167. Association for Computational Linguistics.

Och, F. J. and Ney, H. (2003). "A Systematic Comparison of Various Statistical Alignment Models." *Computational Linguistics*, **29** (1), pp. 19–51.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). "BLEU: a Method for Automatic Evaluation of Machine Translation." In *Proceedings of the 40th ACL*, pp. 311–318. Association for Computational Linguistics.

Riesa, J. and Marcu, D. (2010). "Hierarchical Search for Word Alignment." In *Proceedings of the 48th annual meeting of the association for computational linguistics*, pp. 157–166. Association for Computational Linguistics.

Riley, D. and Gildea, D. (2012). "Improving the IBM Alignment Models using Variational Bayes." In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers*, Vol. 2, pp. 306–310. Association for Computational Linguistics.

Shi, J. and Malik, J. (2000). "Normalized cuts and Image Segmentation." *IEEE Transactions on Pattern Analysis and Machine intelligence*, **22** (8), pp. 888–905.

Vogel, S., Ney, H., and Tillmann, C. (1996). "HMM-based Word Alignment in Statistical Translation." In *Proceedings of the 16th Conference on Computational Linguistics*, Vol. 2, pp. 836–841. Association for Computational Linguistics.

Wang, H. and Lepage, Y. (2016a). "Combining fast_align with Hierarchical Sub-sentential Alignment for Better Word Alignments." In *Proceedings of the Sixth Workshop on Hybrid Approaches to Translation, COLING 2016*, pp. 1–7.

Wang, H. and Lepage, Y. (2016b). "Yet Another Symmetrical and Real-time Word Alignment Method: Hierarchical Sub-sentential Alignment using F-measure." In *The 30th Pacific Asia Conference on Language, Information and Computation*, pp. 143–152. PALCIC.

Wang, W., Knight, K., and Marcu, D. (2007). "Binarizing Syntax Trees to Improve Syntax-Based Machine Translation Accuracy." In *EMNLP-CoNLL*, pp. 746–754. Citeseer.

Wu, D. (1995). "Stochastic Inversion Transduction Grammars, with Application to Segmentation, Bracketing, and Alignment of Parallel Corpora." In *Proceedings of the 14th International Joint Conference on Artificial Intelligence*, Vol. 95, pp. 1328–1335.

Wu, D. (1997). "Stochastic Inversion Transduction Grammars and Bilingual Parsing of Parallel

Corpora." *Computational Linguistics*, **23** (3), pp. 377–403.

Wu, H. and Wang, H. (2007). "Comparative Study of Word Alignment Heuristics and Phrase-based SMT." In *Proceedings of the MT Summit XI.*

Xiong, D., Zhang, M., and Li, H. (2010). "Learning Translation Boundaries for Phrase-based Decoding." In *Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 136–144. Association for Computational Linguistics.

Zens, R., Ney, H., Watanabe, T., and Sumita, E. (2004). "Reordering Constraints for Phrase-based Statistical Machine Translation." In *Proceedings of the 20th International Conference on Computational Linguistics*, p. 205. Association for Computational Linguistics.

Zha, H., He, X., Ding, C., Simon, H., and Gu, M. (2001). "Bipartite Graph Partitioning and Data Clustering." In *Proceedings of the 10th International Conference on Information and knowledge Management*, pp. 25–32. ACM.

Zhang, H. and Gildea, D. (2005). "Stochastic Lexicalized Inversion Transduction Grammar for Alignment." In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pp. 475–482. Association for Computational Linguistics.

Zhang, M., Jiang, H., Aw, A., Sun, J., Li, S., and Tan, C. L. (2007). "A tree-to-tree Alignment-based Model for Statistical Machine Translation.". pp. 535–542.

Hao Wang : Hao Wang received his M.E. from Waseda University and M.Sc from Shanghai University in 2014. He is currently a Ph.D. candidate at Graduate School of Information, Production and Systems, Waseda University, sponsored by Oversea Graduate Student Project of the China Scholarship Council. His research interests include natural language processing, especially machine translation.

Yves Lepage : Yves Lepage received his Ph.D. degree from GETA, Grenoble university, France. He worked for ATR labs, Japan, as an invited researcher and a senior researcher until 2006. He joined Waseda University, Graduate School of Information, Production and Systems in April 2010. He is a member of the Information Processing Society of Japan, the Japanese Natural Language Processing Association, and the French Natural Language Processing Association, ATALA. He was editor-in-chief of the French journal on Natural Language Processing, TAL, from 2008 to 2016.