Exploring Syntactic Structural Features for Sub-Tree Alignment using Bilingual Tree Kernels

Jun Sun[1,2]   **Min Zhang[1]**   **Chew Lim Tan[2]**

[1] Institute for Infocomm Research   [2]School of Computing, National University of Singapore

sunjun@comp.nus.edu.sg   mzhang@i2r.a-star.edu.sg   tancl@comp.nus.edu.sg

Abstract

We propose Bilingual Tree Kernels (BTKs) to capture the structural similarities across a pair of syntactic translational equivalences and apply BTKs to sub-tree alignment along with some plain features. Our study reveals that the structural features embedded in a bilingual parse tree pair are very effective for sub-tree alignment and the bilingual tree kernels can well capture such features. The experimental results show that our approach achieves a significant improvement on both gold standard tree bank and automatically parsed tree pairs against a heuristic similarity based method. We further apply the sub-tree alignment in machine translation with two methods. It is suggested that the sub-tree alignment benefits both phrase and syntax based systems by relaxing the constraint of the word alignment.

1   Introduction

Syntax based Statistical Machine Translation (SMT) systems allow the translation process to be more grammatically performed, which provides decent reordering capability. However, most of the syntax based systems construct the syntactic translation rules based on *word alignment*, which not only suffers from the pipeline errors, but also fails to effectively utilize the syntactic structural features. To address those deficiencies, Tinsley et al. (2007) attempt to directly capture the syntactic translational equivalences by automatically conducting sub-tree alignment, which can be defined as follows:

A sub-tree alignment process pairs up sub-tree pairs across bilingual parse trees whose contexts are semantically translational equivalent. According to Tinsley et al. (2007), a sub-tree aligned parse tree pair follows the following criteria:

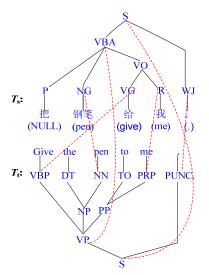(i)   a node can only be linked once;

Figure 1: Sub-tree alignment as referred to Node alignment

(ii)   descendants of a source linked node may only link to descendants of its target linked counterpart;

(iii)   ancestors of a source linked node may only link to ancestors of its target linked counterpart.

By sub-tree alignment, translational equivalent sub-tree pairs are coupled as aligned counterparts. Each pair consists of both the lexical constituents and their maximum tree structures generated over the lexical sequences in the original parse trees. Due to the 1-to-1 mapping between sub-trees and tree nodes, sub-tree alignment can also be considered as node alignment by conducting multiple links across the internal nodes as shown in Fig. 1.

Previous studies conduct sub-tree alignments by either using a rule based method or conducting some similarity measurement only based on lexical features. Groves et al. (2004) conduct sub-tree alignment by using some heuristic rules, lack of extensibility and generality. Tinsley et al. (2007)

and Imamura (2001) propose some score functions based on the lexical similarity and co-occurrence. These works fail to utilize the structural features, rendering the syntactic rich task of sub-tree alignment less convincing and attractive. This may be due to the fact that the syntactic structures in a parse tree pair are hard to describe using plain features. In addition, explicitly utilizing syntactic tree fragments results in exponentially high dimensional feature vectors, which is hard to compute. Alternatively, convolution parse tree kernels (Collins and Duffy, 2001), which implicitly explore the tree structure information, have been successfully applied in many NLP tasks, such as Semantic parsing (Moschitti, 2004) and Relation Extraction (Zhang et al. 2006). However, all those studies are carried out in monolingual tasks. In multilingual tasks such as machine translation, tree kernels are seldom applied.

In this paper, we propose Bilingual Tree Kernels (BTKs) to model the bilingual translational equivalences, in our case, to conduct sub-tree alignment. This is motivated by the decent effectiveness of tree kernels in expressing the similarity between tree structures. We propose two kinds of BTKs named dependent Bilingual Tree Kernel (dBTK), which takes the sub-tree pair as a whole and independent Bilingual Tree Kernel (iBTK), which individually models the source and the target sub-trees. Both kernels can be utilized within different feature spaces using various representations of the sub-structures.

Along with BTKs, various lexical and syntactic structural features are proposed to capture the correspondence between bilingual sub-trees using a polynomial kernel. We then attempt to combine the polynomial kernel and BTKs to construct a composite kernel. The sub-tree alignment task is considered as a binary classification problem. We employ a kernel based classifier with the composite kernel to classify each candidate of sub-tree pair as *aligned* or *unaligned*. Then a greedy search algorithm is performed according to the three criteria of sub-tree alignment within the space of candidates classified as *aligned*.

We evaluate the sub-tree alignment on both the gold standard tree bank and an automatically parsed corpus. Experimental results show that the proposed BTKs benefit sub-tree alignment on both corpora, along with the lexical features and the plain structural features. Further experiments in machine translation also suggest that the obtained sub-tree alignment can improve the performance of both phrase and syntax based SMT systems.

2 Bilingual Tree Kernels

In this section, we propose the two BTKs and study their capability and complexity in modeling the bilingual structural similarity. Before elaborating the concepts of BTKs, we first illustrate some notations to facilitate further understanding.

Each sub-tree pair $(S \cdot T)$ can be explicitly decomposed into multiple sub-structures which belong to the given sub-structure spaces. $\mathcal{S}_I = \{s_1, \dots, s_i, \dots, s_I\}$ refers to the *source* tree sub-structure space; while $\mathcal{T}_J = \{t_1, \dots, t_j, \dots, t_J\}$ refers to the *target* sub-structure space. A sub-structure pair (s_i, t_j) refers to an element in the set of the Cartesian product of the two sub-structure spaces: $(s_i, t_j) \in \mathcal{S}_I \times \mathcal{T}_J$.

2.1 Independent Bilingual Tree Kernel (iBTK)

Given the sub-structure spaces $\mathcal{S}_I$ and $\mathcal{T}_J$, we construct two vectors using the integer counts of the source and target sub-structures:

$$\phi(S) = \big(\#(s_1), \dots, \#(s_k), \dots, \#(s_{|\mathcal{S}_I|})\big)$$
$$\phi(T) = \big(\#(t_1), \dots, \#(t_k), \dots, \#(t_{|\mathcal{T}_J|})\big)$$

where $\#(s_k)$ and $\#(t_k)$ are the numbers of occurrences of the sub-structures s_k and t_k. In order to compute the dot product of the feature vectors in the exponentially high dimensional feature space, we introduce the tree kernel functions as follows:

$$\mathcal{K}_{iBTK}(S \cdot T, S' \cdot T') = \mathcal{K}(S, S') + \mathcal{K}(T, T')$$

The iBTK is defined as a composite kernel consisting of a source tree kernel and a target tree kernel which measures the source and the target structural similarity respectively. Therefore, the composite kernel can be computed using the ordinary monolingual tree kernels (Collins and Duffy, 2001).

$$\mathcal{K}(S, S') = < \phi(S), \phi(S') >$$
$$= \sum_{i=1}^{|\mathcal{S}_I|} \left(\sum_{n_s \in N_S} I_i(n_s) \cdot \sum_{n_s' \in N_S'} I_i(n_s') \right)$$
$$= \sum_{n_s \in N_S} \sum_{n_s' \in N_S'} \Delta(n_s, n_s')$$

where N_S and N_S' refer to the node sets of the source sub-tree S and S' respectvely. $I_i(n_s)$ is an indicator function which equals to 1 iff the sub-structure s_i is rooted at the node n_s and 0 otherwise. $\Delta(n_s, n_s') = \sum_{i=1}^{|\mathcal{S}_I|} \big(I_i(n_s) \cdot I_i(n_s')\big)$ is the number of identical sub-structures rooted at n_s and n_s'. Then we compute the $\Delta(n_s, n_s')$ function as follows:

(1) If the production rule at n_s and n_s' are different, $\Delta(n_s, n_s') = 0$;

(2) else if both n_s and n_s' are POS tags, $\Delta(n_s, n_s') = \lambda$;

(3) else, $\Delta(n_s, n_s') = \lambda \prod_{l=1}^{nc(n_s)} \left(1 + \Delta(c(n_s, l), c(n_s', l))\right)$.

where $nc(n_s)$ is the child number of n_s, $c(n_s, l)$ is the lth child of n_s, λ is the decay factor used to make the kernel value less variable with respect to the number of sub-structures.

Similarly, we can decompose the target kernel as $\mathcal{K}(T, T') = \sum_{n_t \in N_T} \sum_{n_t' \in N_T'} \Delta(n_t, n_t')$ and run the algorithm above as well.

The disadvantage of the iBTK is that it fails to capture the correspondence across the sub-structure pairs. However, the composite style of constructing the iBTK helps keep the computational complexity comparable to the monolingual tree kernel, which is $O(|N_S| \cdot |N_S'| + |N_T| \cdot |N_T'|)$.

2.2 Dependent Bilingual Tree Kernel (dBTK)

The iBTK explores the structural similarity of the source and the target sub-trees respectively. As an alternative, we further define a kernel to capture the relationship across the counterparts without increasing the computational complexity. As a result, we propose the dependent Bilingual Tree kernel (dBTK) to jointly evaluate the similarity across sub-tree pairs by enlarging the feature space to the Cartesian product of the two sub-structure sets.

A dBTK takes the source and the target sub-structure pair as a whole and recursively calculate over the joint sub-structures of the given sub-tree pair. We define the dBTK as follows:

Given the sub-structure space $\mathcal{S}_I \times \mathcal{T}_J$, we construct a vector using the integer counts of the sub-structure pairs to represent a sub-tree pair:

$$\phi(S \cdot T) = \begin{pmatrix} \#(s_1, t_1), \dots, \#(s_1, t_{|\mathcal{T}_J|}), \#(s_2, t_1), \\ \dots, \#(s_{|\mathcal{S}_I|}, t_1), \dots, \#(s_{|\mathcal{S}_I|}, t_{|\mathcal{T}_J|}) \end{pmatrix}$$

where $\#(s_i, t_j)$ is the number of occurrences of the sub-structure pair (s_i, t_j).

$$\mathcal{K}_{dBTK}(S \cdot T, S' \cdot T')$$

$$= <\phi(S \cdot T), \phi(S' \cdot T')>$$

$$= \sum_{k=1}^{|\mathcal{S}_I \times \mathcal{T}_J|} \begin{pmatrix} \sum_{n_s \in N_S} \sum_{n_t \in N_T} I_k(n_s, n_t) \cdot \\ \sum_{n_s' \in N_S'} \sum_{n_t' \in N_T'} I_k(n_s', n_t') \end{pmatrix}$$

$$= \sum_{n_s \in N_S} \sum_{n_t \in N_T} \sum_{n_s' \in N_S'} \sum_{n_t' \in N_T'} \Delta \begin{pmatrix} (n_s, n_t), \\ (n_s', n_t') \end{pmatrix} \quad (1)$$

$$= \sum_{n_s \in N_S} \sum_{n_t \in N_T} \sum_{n_s' \in N_S'} \sum_{n_t' \in N_T'} \begin{pmatrix} \Delta(n_s, n_s') \cdot \\ \Delta(n_t, n_t') \end{pmatrix} \quad (2)$$

$$= \sum_{n_s \in N_S} \sum_{n_s' \in N_S'} \Delta(n_s, n_s') \sum_{n_t \in N_T} \sum_{n_t' \in N_T'} \Delta(n_t, n_t')$$

$$= \mathcal{K}(S, S') \cdot \mathcal{K}(T, T')$$

It is infeasible to explicitly compute the kernel function by expressing the sub-trees as feature vectors. In order to achieve convenient computation, we deduce the kernel function as the above.

The deduction from (1) to (2) is derived according to the fact that the number of identical sub-structure pairs rooted in the node pairs (n_s, n_t) and (n_s', n_t') equals to the product of the respective counts. As a result, the dBTK can be evaluated as a product of two monolingual tree kernels. Here we verify the correctness of the kernel by directly constructing the feature space for the inner product. Alternatively, Cristianini and Shawe-Taylor (2000) prove the positive semi-definite characteristic of the tensor product of two kernels. The decomposition benefits the efficient computation to use the algorithm for the monolingual tree kernel in Section 2.1.

The computational complexity of the dBTK is still $O(|N_S| \cdot |N_S'| + |N_T| \cdot |N_T'|)$.

3 Sub-structure Spaces for BTKs

The syntactic translational equivalences under BTKs are evaluated with respective to the sub-structures factorized from the candidate sub-tree pairs. In this section, we propose different sub-structures to facilitate the measurement of syntactic similarity for sub-tree alignment. Since the proposed BTKs can be computed by individually evaluating the source and target monolingual tree kernels, the definition of the sub-structure can be simplified to base only on monolingual sub-trees.

3.1 Subset Tree

Motivated from Collins and Duffy (2002) in monolingual tree kernels, the Subset Tree (SST) can be employed as sub-structures. An SST is any subgraph, which includes more than one non-terminal node, with the constraint that the entire rule productions are included. Fig. 2 shows an example of the SSTs decomposed from the source sub-tree rooted at VP*.

3.2 Root directed Subset Tree

Monolingual Tree kernels achieve decent performance using the SSTs due to the rich exploration of syntactic information. However, the sub-tree alignment task requires strong capability of discriminating the sub-trees with their roots across adjacent generations, because those candidates share many identical SSTs. As illustrated in Fig 2, the source sub-tree rooted at VP*, which should be aligned to the target sub-tree rooted at NP*, may be likely aligned to the sub-tree rooted at PP*,
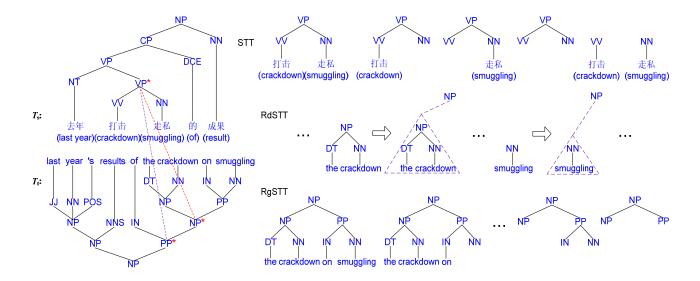
Figure 2: Illustration of SST, RdSST and RgSST

which shares quite a similar context with NP*. It is also easy to show that the latter shares all the SSTs that the former obtains. In consequence, the values of the SST based kernel function are quite similar between the candidate sub-tree pair rooted at (VP*,NP*) and (VP*,PP*).

In order to effectively differentiate the candidates like the above, we propose the Root directed Subset Tree (RdSST) by encapsulating each SST with the root of the given sub-tree. As shown in Fig 2, a sub-structure is considered identical to the given examples, when the SST is identical and the root tag of the given sub-tree is NP. As a result, the kernel function in Section 2.1 is re-defined as:

$$\mathcal{K}(S,S') = \sum_{n_s \in N_S} \sum_{n'_s \in N'_S} \Delta(n_s, n'_s) I(r_s, r'_s)$$
$$= I(r_s, r'_s) \sum_{n_s \in N_S} \sum_{n'_s \in N'_S} \Delta(n_s, n'_s)$$

where r_s and r'_s are the root nodes of the sub-tree S and S' respectively. The indicator function $I(r_s, r'_s)$ equals to 1 if r_s and r'_s are identical, and 0 otherwise. Although defined for individual SST, the indicator function can be evaluated outside the summation, without increasing the computational complexity of the kernel function.

3.3 Root generated Subset Tree

Some grammatical tags (NP/VP) may have identical tags as their parents or children which may make RdSST less effective. Consequently, we step further to propose the sub-structure of Root generated Subset Tree (RgSST). An RgSST requires the root node of the given sub-tree to be part of the sub-structure. In other words, all sub-structures should be generated from the root of the given sub-tree as presented in Fig. 2. Therefore the ker-

nel function can be simplified to only capture the sub-structure rooted at the root of the sub-tree.

$$\mathcal{K}(S,S') = \Delta(r_s, r'_s)$$

where r_s and r'_s are the root nodes of the sub-tree S and S' respectively. The time complexity is reduced to $O(|N_S| + |N'_S| + |N_T| + |N'_T|)$.

3.4 Root only

More aggressively, we can simplify the kernel to only measure the common root node without considering the complex tree structures. Therefore the kernel function is simplified to be a binary function with time complexity $O(1)$.

$$\mathcal{K}(S,S') = I(r_s, r'_s)$$

4 Plain features

Besides BTKs, we introduce various plain lexical features and structural features which can be expressed as feature functions. The lexical features with directions are defined as conditional feature functions based on the conditional lexical translation probabilities. The plain syntactic structural features can deal with the structural divergence of bilingual parse trees in a more general perspective.

4.1 Lexical and Word Alignment Features

In this section, we define seven lexical features to measure semantic similarity of a given sub-tree pair.

Internal Lexical Features: We define two lexical features with respective to the internal span of the sub-tree pair.

$$\phi_1(S|T) = \left(\prod_{v \in in(T)} \sum_{u \in in(S)} P(u|v)\right)^{\frac{1}{|in(T)|}}$$

$$\phi_2(T|S) = \left(\prod_{u \in in(S)} \sum_{v \in in(T)} P(v|u)\right)^{\frac{1}{|in(S)|}}$$

where $P(v|u)$ refers to the lexical translation probability from the source word u to the target word v within the sub-tree spans, while $P(u|v)$ refers to that from target to source; $in(S)$ refers to the word set for the internal span of the source sub-tree S, while $in(T)$ refers to that of the target sub-tree T.

Internal-External Lexical Features: These features are motivated by the fact that lexical translation probabilities within the translational equivalence tend to be high, and that of the non-equivalent counterparts tend to be low.

$$\phi_3(S|T) = \left(\prod_{v \in in(T)} \sum_{u \in out(S)} P(u|v)\right)^{\frac{1}{|in(T)|}}$$

$$\phi_4(T|S) = \left(\prod_{u \in in(S)} \sum_{v \in out(T)} P(v|u)\right)^{\frac{1}{|in(S)|}}$$

where $out(S)$ refers to the word set for the external span of the source sub-tree S, while $out(T)$ refers to that of the target sub-tree T.

Internal Word Alignment Features: The word alignment links account much for the co-occurrence of the aligned terms. We define the internal word alignment features as follows:

$$\phi_5(S,T) = \frac{\sum_{v \in in(T)} \sum_{u \in in(S)} \delta(u,v) \cdot \left(P(u|v) \cdot P(v|u)\right)^{\frac{1}{2}}}{(|in(S)| \cdot |in(T)|)^{\frac{1}{2}}}$$

where

$$\delta(u,v) = \begin{cases} 1 & \text{if } (u,v) \text{ is aligned} \\ 0 & \text{otherwise} \end{cases}$$

The binary function $\delta(u,v)$ is employed to trigger the computation only when a word aligned link exists for the two words (u,v) within the sub-tree span.

Internal-External Word Alignment Features: Similar to the lexical features, we also introduce the internal-external word alignment features as follows:

$$\phi_6(S,T) = \frac{\sum_{v \in in(T)} \sum_{u \in out(S)} \delta(u,v) \cdot \left(P(u|v) \cdot P(v|u)\right)^{\frac{1}{2}}}{(|out(S)| \cdot |in(T)|)^{\frac{1}{2}}}$$

$$\phi_7(S,T) = \frac{\sum_{v \in out(T)} \sum_{u \in in(S)} \delta(u,v) \cdot \left(P(u|v) \cdot P(v|u)\right)^{\frac{1}{2}}}{(|in(S)| \cdot |out(T)|)^{\frac{1}{2}}}$$

where

$$\delta(u,v) = \begin{cases} 1 & \text{if } (u,v) \text{ is aligned} \\ 0 & \text{otherwise} \end{cases}$$

4.2 Online Structural Features

In addition to the lexical correspondence, we also capture the structural divergence by introducing the following tree structural features.

Span difference: Translational equivalent sub-tree pairs tend to share similar length of spans.

Thus the model will penalize the candidate sub-tree pairs with largely different length of spans.

$$\varphi_1(S,T) = \left| \frac{|in(S)|}{|in(\mathbf{S})|} - \frac{|in(T)|}{|in(\mathbf{T})|} \right|$$

S and **T** refer to the entire source and target parse trees respectively. Therefore, $|in(\mathbf{S})|$ and $|in(\mathbf{T})|$ are the respective span length of the parse tree used for normalization.

Number of Descendants: Similarly, the number of the root's descendants of the aligned sub-trees should also correspond.

$$\varphi_2(S,T) = \left| \frac{|R(S)|}{|R(\mathbf{S})|} - \frac{|R(T)|}{|R(\mathbf{T})|} \right|$$

where $R(.)$ refers to the descendant set of the root to a sub-tree.

Tree Depth difference: Intuitively, translational equivalent sub-tree pairs tend to have similar depth from the root of the parse tree. We allow the model to penalize the candidate sub-tree pairs with quite different distance of path from the root of the parse tree to the root of the sub-tree.

$$\varphi_3(S,T) = \left| \frac{Depth(S)}{Height(\mathbf{S})} - \frac{Depth(T)}{Height(\mathbf{T})} \right|$$

5 Alignment Model

Given feature spaces defined in the last two sections, we propose a 2-phase sub-tree alignment model as follows:

In the 1st phase, a kernel based classifier, SVM in our study, is employed to classify each candidate sub-tree pair as *aligned* or *unaligned*. The feature vector of the classifier is computed using a composite kernel:

$$\mathcal{K}(S \cdot T, S' \cdot T') =$$

$$\theta_0 \widehat{\mathcal{K}}_p(S \cdot T, S' \cdot T') + \sum_{i=1}^{K} \theta_i \widehat{\mathcal{K}}_{BTK}^i(S \cdot T, S' \cdot T')$$

$\widehat{\mathcal{K}}_p(\cdot,\cdot)$ is the normalized form of the polynomial kennel $\mathcal{K}_p(\cdot,\cdot)$, which is a polynomial kernel with the degree of 2, utilizing the plain features. $\widehat{\mathcal{K}}_{BTK}^i(\cdot,\cdot)$ is the normalized form of the BTK $\mathcal{K}_{BTK}^i(\cdot,\cdot)$, exploring the corresponding sub-structure space. The composite kernel can be constructed using the polynomial kernel for plain features and various BTKs for tree structure by linear combination with coefficient θ_i, where $\sum_{i=0}^{K} \theta_i = 1$.

In the 2nd phase, we adopt a greedy search with respect to the alignment probabilities. Since SVM is a large margin based discriminative classifier rather than a probabilistic model, we introduce a sigmoid function to convert the distance against the hyperplane to a posterior alignment probability as follows:

$$P(a_+|S,T) = \frac{1}{1 + e^{-D_+}}$$

$$P(a_-|S,T) = \frac{1}{1 + e^{-D_-}}$$

where D_+ is the distance for the instances classified as *aligned* and D_- is that for the *unaligned*. We use $P(a_+|S,T)$ as the confidence to conduct the *sure* links for those classified as *aligned*. On this perspective, the alignment probability is suitable as a searching metric. The search space is reduced to that of the candidates classified as *aligned* after the 1st phase.

6 Experiments on Sub-Tree Alignments

In order to evaluate the effectiveness of the alignment model and its capability in the applications requiring syntactic translational equivalences, we employ two corpora to carry out the sub-tree alignment evaluation. The first is HIT gold standard English Chinese parallel tree bank referred as HIT corpus[1]. The other is the automatically parsed bilingual tree pairs selected from FBIS corpus (allowing minor parsing errors) with human annotated sub-tree alignment.

6.1 Data preparation

HIT corpus, which is collected from English learning text books in China as well as example sentences in dictionaries, is used for the gold standard corpus evaluation. The word segmentation, tokenization and parse-tree in the corpus are manually constructed or checked. The corpus is constructed with manually annotated sub-tree alignment. The annotation strictly reserves the semantic equivalence of the aligned sub-tree pair. Only *sure* links are conducted in the internal node level, without considering *possible* links adopted in word alignment. A different annotation criterion of the Chinese parse tree, designed by the annotator, is employed. Compared with the widely used Penn TreeBank annotation, the new criterion utilizes some different grammar tags and is able to effectively describe some rare language phenomena in Chinese. The annotator still uses Penn TreeBank annotation on the English side. The statistics of HIT corpus used in our experiment is shown in Table 1. We use 5000 sentences for experiment and divide them into three parts, with 3k for training, 1k for testing and 1k for tuning the parameters of kernels and thresholds of pruning the negative instances.

---

[1]HIT corpus is designed and constructed by HIT-MITLAB. http://mitlab.hit.edu.cn/index.php/resources.html .

	Chinese	English
# of Sentence pair	5000	
Avg. Sentence Length	12.93	12.92
Avg. # of sub-tree	21.40	23.58
Avg. # of alignment	11.60	

Table 1. Corpus Statistics for HIT corpus

Most linguistically motivated syntax based SMT systems require an automatic parser to perform the rule induction. Thus, it is important to evaluate the sub-tree alignment on the automatically parsed corpus with parsing errors. In addition, HIT corpus is not applicable for MT experiment due to the problems of domain divergence, annotation discrepancy (Chinese parse tree employs a different grammar from Penn Treebank annotations) and degree of tolerance for parsing errors.

Due to the above issues, we annotate a new data set to apply the sub-tree alignment in machine translation. We randomly select 300 bilingual sentence pairs from the Chinese-English FBIS corpus with the length ≤ 30 in both the source and target sides. The selected plain sentence pairs are further parsed by Stanford parser (Klein and Manning, 2003) on both the English and Chinese sides. We manually annotate the sub-tree alignment for the automatically parsed tree pairs according to the definition in Section 1. To be fully consistent with the definition, we strictly reserve the semantic equivalence for the aligned sub-trees to keep a high precision. In other words, we do not conduct any doubtful links. The corpus is further divided into 200 aligned tree pairs for training and 100 for testing as shown in Table 2.

	Chinese	English
# of Sentence pair	300	
Avg. Sentence Length	16.94	20.81
Avg. # of sub-tree	28.97	34.39
Avg. # of alignment	17.07	

Table 2. Statistics of FBIS selected Corpus

6.2 Baseline approach

We implement the work in Tinsley et al. (2007) as our baseline methodology.

Given a tree pair $< \mathbf{S}, \mathbf{T} >$, the baseline approach first takes all the links between the sub-tree pairs as alignment hypotheses, i.e., the Cartesian product of the two sub-tree sets:

$$\{S_1, ..., S_i, ..., S_I\} \times \{T_1, ..., T_j, ..., T_J\}$$

By using the lexical translation probabilities, each hypothesis is assigned an alignment score. All hypotheses with zero score are pruned out.

Feature Space	P	R	F
Lex	61.62	58.33	59.93
Lex +Online Str	70.08	69.02	69.54
Plain +dBTK-STT	80.36	78.08	79.20
Plain +dBTK-RdSTT	87.52	74.13	80.27
Plain +dBTK-RgSTT	88.54	70.18	78.30
Plain +dBTK-Root	81.05	84.38	**82.68**
Plain +iBTK-STT	81.57	73.51	77.33
Plain +iBTK-RdSTT	82.27	77.85	80.00
Plain +iBTK-RgSTT	82.92	78.77	**80.80**
Plain +iBTK-Root	76.37	76.81	76.59
Plain +dBTK-Root +iBTK-RgSTT	85.53	85.12	**85.32**
Baseline	64.14	66.99	65.53

Table 3. Structure feature contribution for HIT test set
*Plain= Lex +Online Str

Feature Space	P	R	F
Lex	73.48	71.66	72.56
Lex +Online Str	77.02	73.63	75.28
Plain +dBTK-STT	81.44	74.42	77.77
Plain +dBTK-RdSTT	81.40	69.29	74.86
Plain +dBTK-RgSTT	81.90	67.32	73.90
Plain +dBTK-Root	78.60	80.90	**79.73**
Plain +iBTK-STT	82.94	79.44	81.15
Plain +iBTK-RdSTT	83.14	80	**81.54**
Plain +iBTK-RgSTT	83.09	79.72	81.37
Plain +iBTK-Root	78.61	79.49	79.05
Plain +dBTK-Root +iBTK-RdSTT	82.70	82.70	**82.70**
Baseline	70.48	78.70	74.36

Table 4. Structure feature contribution for FBIS test set

Then the algorithm iteratively selects the link of the sub-tree pairs with the maximum score as a *sure* link, and blocks all hypotheses that contradict with this link and itself, until no non-blocked hypotheses remain.

The baseline system uses many heuristics in searching the optimal solutions with alternative score functions. Heuristic *skip1* skips the tied hypotheses with the same score, until it finds the highest-scoring hypothesis with no competitors of the same score. Heuristic *skip2* deals with the same problem. Initially, it skips over the tied hypotheses. When a hypothesis sub-tree pair (S_i, T_j) without any competitor of the same score is found, where neither S_i nor T_j has been skipped over, the hypothesis is chosen as a *sure* link. Heuristic *span1* postpones the selection of the hypotheses on the POS level. Since the highest-scoring hypotheses tend to appear on the leaf nodes, it may introduce ambiguity when conducting the alignment for a POS node whose child word appears twice in a sentence.

The baseline method proposes two score functions based on the lexical translation probability. They also compute the score function by splitting the tree into the internal and external components.

Tinsley et al. (2007) adopt the lexical translation probabilities dumped by GIZA++ (Och and Ney, 2003) to compute the span based scores for each pair of sub-trees. Although all of their heuristics combinations are re-implemented in our study, we only present the best result among them with the highest Recall and F-value as our baseline, denoted as *skip2_s1_span1*[2].

---

[2] s1 denotes score function 1 in Tinsley et al. (2007), skip2_s1_span1 denotes the utilization of heuristics skip2 and span1 while using score function 1

6.3 Experimental settings

We use SVM with binary classes as the classifier. In case of the implementation, we modify the Tree Kernel tool (Moschitti, 2004) and SVMLight (Joachims, 1999). The coefficient θ_i for the composite kernel are tuned with respect to F-measure (**F**) on the development set of HIT corpus. We empirically set C=2.4 for SVM and use $\alpha = 0.23$, the default parameter $\lambda = 0.4$ for BTKs.

Since the negative training instances largely overwhelm the positive instances, we prune the negative instances using the thresholds according to the lexical feature functions ($\phi_1, \phi_2, \phi_3, \phi_4$) and online structural feature functions ($\varphi_1, \varphi_2, \varphi_3$). Those thresholds are also tuned on the development set of HIT corpus with respect to F-measure.

To learn the lexical and word alignment features for both the proposed model and the baseline method, we train GIZA++ on the entire FBIS bilingual corpus (240k). The evaluation is conducted by means of Precision (**P**), Recall (**R**) and F-measure (**F**).

6.4 Experimental results

In Tables 3 and 4, we incrementally enlarge the feature spaces in certain order for both corpora and examine the feature contribution to the alignment results. In detail, the iBTKs and dBTKs are firstly combined with the polynomial kernel for plain features individually, then the best iBTK and dBTK are chosen to construct a more complex composite kernel along with the polynomial kernel for both corpora. The experimental results show that:

- All the settings with structural features of the proposed approach achieve better performance than the baseline method. This is because the

baseline only assesses semantic similarity using the lexical features. The improvement suggests that the proposed framework with syntactic structural features is more effective in modeling the bilingual syntactic correspondence.

- By introducing BTKs to construct a composite kernel, the performance in both corpora is significantly improved against only using the polynomial kernel for plain features. This suggests that the structural features captured by BTKs are quite useful for the sub-tree alignment task. We also try to use BTKs alone without the polynomial kernel for plain features; however, the performance is rather low. This suggests that the structure correspondence cannot be used to measure the semantically equivalent tree structures alone, since the same syntactic structure tends to be reused in the same parse tree and lose the ability of disambiguation to some extent. In other words, to capture the semantic similarity, structure features requires lexical features to cooperate.

- After comparing iBTKs with the corresponding dBTKs, we find that for *FBIS* corpus, iBTK greatly outperforms dBTK in any feature space except the Root space. However, when it comes the *HIT* corpus, the gaps between the corresponding iBTKs and dBTKs are much closer, while on the Root space, dBTK outperforms iBTK to a large amount. This finding can be explained by the relationship between the amount of training data and the high dimensional feature space. Since dBTKs are constructed in a joint manner which obtains a much larger high dimensional feature space than those of iBTKs, dBTKs require more training data to excel its capability, otherwise it will suffer from the data sparseness problem. The reason that dBTK outperforms iBTK in the feature space of Root in *FBIS* corpus is that although it is a joint feature space, the Root node pairs can be constructed from a close set of grammar tags and to form a relatively low dimensional space.

  As a result, when applying to *FBIS* corpus, which only contains limited amount of training data, dBTKs will suffer more from the data sparseness problem, and therefore, a relatively low performance. When enlarging the amount of training corpus to the *HIT* corpus, the ability of dBTKs excels and the benefit from data increasing of dBTKs is more significant than iBTKs.

- We also find that the introduction of BTKs gains more improvement in HIT gold standard corpus

than in FBIS corpus. Other than the factor of the amount of training data, this is also because the plain features in Table 3 are not as effective as those in Table 4, since they are trained on FBIS corpus which facilitates Table 4 more with respect to the domains. On the other hand, the grammatical tags and syntactic tree structures are more accurate in HIT corpus, which facilitates the performance of BTKs in Table 3.

- On the comparison across the different feature spaces of BTKs, we find that STT, RdSTT and TgSTT are rather selective, since **Recall**s of those feature spaces are relatively low, exp. for HIT corpus. However, the Root sub-structure obtains a satisfactory **Recall** for both corpora. That's why we attempt to construct a more complex composite kernel in adoption of the kernel of dBTK-Root as below.

- To gain an extra performance boosting, we further construct a composite kernel which includes the best iBTK and the best dBTK for each corpus along with the polynomial kernel for plain features. In the HIT corpus, we use dBTK in the Root space and iBTK in the RgSST space; while for FBIS corpus, we use dBTK in the Root space and iBTK in the RdSST space. The experimental results suggest that by combining iBTK and dBTK together, we can achieve more improvement.

7 Experiments on Machine Translation

In addition to the intrinsic alignment evaluation, we further conduct the extrinsic MT evaluation. We explore the effectiveness of sub-tree alignment for both phrase based and linguistically motivated syntax based SMT systems.

7.1 Experimental configuration

In the experiments, we train the translation model on FBIS corpus (7.2M (Chinese) + 9.2M (English) words in 240,000 sentence pairs) and train a 4-gram language model on the Xinhua portion of the English Gigaword corpus (181M words) using the SRILM Toolkits (Stolcke, 2002). We use these sentences with less than 50 characters from the NIST MT-2002 test set as the development set (to speed up tuning for syntax based system) and the NIST MT-2005 test set as our test set. We use the Stanford parser (Klein and Manning, 2003) to parse bilingual sentences on the training set and Chinese sentences on the development and test set. The evaluation metric is case-sensitive BLEU-4.

313

System	Model	BLEU
Moses	BP*	23.86
	DirC	23.98
	EWoS	24.48
Syntax STSG	STSG	24.71
	DirC	25.16
	EWoS	25.38
Syntax STSSG	STSSG	25.92
	DirC	25.95
	EWoS	26.45

Table 5. MT evaluation on various systems
*BP denotes bilingual phrases

For the phrase based system, we use Moses (Koehn et al., 2007) with its default settings. For the syntax based system, since sub-tree alignment can directly benefit Tree-2-Tree based systems, we apply the sub-tree alignment in a syntax system based on Synchronous Tree Substitution Grammar (STSG) (Zhang et al., 2007). The STSG based decoder uses a pair of *elementary tree*[3] as a basic translation unit. Recent research on tree based systems shows that relaxing the restriction from tree structure to tree sequence structure (Synchronous Tree Sequence Substitution Grammar: STSSG) significantly improves the translation performance (Zhang et al., 2008). We implement the STSG/STSSG based model in the Pisces decoder with the identical features and settings in Sun et al. (2009). In the Pisces decoder, the STSSG based decoder translates each span iteratively in a bottom up manner which guarantees that when translating a source span, any of its sub-spans is already translated. The STSG based decoding can be easily performed with the STSSG decoder by restricting the translation rule set to be elementary tree pairs only.

As for the alignment setting, we use the word alignment trained on the entire FBIS (240k) corpus by GIZA++ with heuristic grow-diag-final for both Moses and the syntax system. For sub-tree-alignment, we use the above word alignment to learn lexical/word alignment feature, and train with the FBIS training corpus (200) using the composite kernel of Plain+dBTK-Root+iBTK-RdSTT.

7.2 Experimental results

Compared with the adoption of word alignment, translational equivalences generated from structural alignment tend to be more grammatically aware and syntactically meaningful. However, utilizing syntactic translational equivalences alone for machine translation loses the capability of modeling non-syntactic phrases (Koehn et al., 2003). Consequently, instead of using phrases constraint by sub-tree alignment alone, we attempt to combine word alignment and sub-tree alignment and deploy the capability of both with two methods.

- *Directly Concatenate* (DirC) is operated by directly concatenating the rule set genereted from sub-tree alignment and the original rule set generated from word alignment (Tinsley et al., 2009). As shown in Table 5, we gain minor improvement in the Bleu score for all configurations.

- Alternatively, we proposed a new approach to generate the rule set from the scratch. We constrain the bilingual phrases to be consistent with *Either Word alignment or Sub-tree alignment* (EWoS) instead of being originally consistent with the word alignment only. The method helps tailoring the rule set decently without redundant counts for syntactic rules. The performance is further improved compared to DirC in all systems.

The findings suggest that with the modeling of non-syntactic phrases maintained, more emphasis on syntactic phrases can benefit both the phrase and syntax based SMT systems.

8 Conclusion

In this paper, we explore syntactic structure features by means of Bilingual Tree Kernels and apply them to bilingual sub-tree alignment along with various lexical and plain structural features. We use both gold standard tree bank and the automatically parsed corpus for the sub-tree alignment evaluation. Experimental results show that our model significantly outperforms the baseline method and the proposed Bilingual Tree Kernels are very effective in capturing the cross-lingual structural similarity. Further experiment shows that the obtained sub-tree alignment benefits both phrase and syntax based MT systems by delivering more weight on syntactic phrases.

---

[3] An elementary tree is a fragment whose leaf nodes can be either non-terminal symbols or terminal symbols.

[4] http://mitlab.hit.edu.cn/ .

314

References

David Burkett and Dan Klein. 2008. *Two languages are better than one* (*for syntactic parsing*). In Proceedings of EMNLP-08. 877-886.

Nello Cristianini and John Shawe-Taylor. 2000. *An introduction to support vector machines and other kernelbased learning methods*. Cambridge: Cambridge University Press.

Michael Collins and Nigel Duffy. 2001. *Convolution Kernels for Natural Language*. In Proceedings of NIPS-01.

Declan Groves, Mary Hearne and Andy Way. 2004. *Robust sub-sentential alignment of phrase-structure trees*. In Proceedings of COLING-04, pages 1072-1078.

Kenji Imamura. 2001. *Hierarchical Phrase Alignment Harmonized with Parsing*. In Proceedings of NLPRS-01, Tokyo. 377-384.

Thorsten Joachims. 1999. *Making large-scale SVM learning practical*. In B. SchÄolkopf, C. Burges, and A. Smola, editors, Advances in Kernel Methods - Support Vector Learning, MIT press.

Dan Klein and Christopher D. Manning. 2003. *Accurate Unlexicalized Parsing*. In Proceedings of ACL-03. 423-430.

Philipp Koehn, Franz Josef Och and Daniel Marcu. 2003. *Statistical phrase-based translation*. In Proceedings of HLT-NAACL-03. 48-54.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin and Evan Herbst. 2007. *Moses: Open Source Toolkit for Statistical Machine Translation*. In Proceedings of ACL-07. 177-180.

Franz Josef Och and Hermann Ney. 2003. *A systematic comparison of various statistical alignment models*. Computational Linguistics, 29(1):19-51, March.

Alessandro Moschitti. 2004. *A Study on Convolution Kernels for Shallow Semantic Parsing*. In Proceedings of ACL-04.

Andreas Stolcke. 2002. *SRILM - an extensible language modeling toolkit*. In Proceedings of ICSLP-02. 901-904.

Jun Sun, Min Zhang and Chew Lim Tan. 2009. *A non-contiguous Tree Sequence Alignment-based Model for Statistical Machine Translation*. In Proceedings of ACL-IJCNLP-09. 914-922.

John Tinsley, Ventsislav Zhechev, Mary Hearne and Andy Way. 2007. *Robust language pair-independent sub-tree alignment*. In Proceedings of MT Summit XI -07.

John Tinsley, Mary Hearne and Andy Way. 2009. *Parallel treebanks in phrase-based statistical machine translation*. In Proceedings of CICLING-09.

Min Zhang, Jie Zhang, Jian Su and Guodong Zhou. 2006. *A Composite Kernel to Extract Relations between Entities with both Flat and Structured Features*. In Proceedings of ACL-COLING-06. 825-832.

Min Zhang, Hongfei Jiang, AiTi Aw, Jun Sun, Sheng Li and Chew Lim Tan. 2007. *A tree-to-tree alignment-based model for statistical machine translation*. In Proceedings of MT Summit XI -07. 535-542.

Min Zhang, Hongfei Jiang, AiTi Aw, Haizhou Li, Chew Lim Tan and Sheng Li. 2008. *A tree sequence alignment-based tree-to-tree translation model*. In Proceedings of ACL-08. 559-567.