Cross-language Projection of Dependency Trees for Tree-to-tree Machine Translation

Yu Shen[1], **Chenhui Chu**[2], **Fabien Cromieres**[2], **Sadao Kurohashi**[1]
[1]Graduate School of Informatics, Kyoto University
[2]Japan Science and Technology Agency
shen-yu@nlp.ist.i.kyoto-u.ac.jp, (chu,fabien)@pa.jst.jp, kuro@i.kyoto-u.ac.jp

Abstract

Syntax-based machine translation (MT) is an attractive approach for introducing additional linguistic knowledge in corpus-based MT. Previous studies have shown that tree-to-string and string-to-tree translation models perform better than tree-to-tree translation models since tree-to-tree models require *two* high quality parsers on the source as well as the target language side. In practice, high quality parsers for both languages are difficult to obtain and thus limit the translation quality. In this paper, we explore a method to transfer parse trees from the language side which has a high quality parser to the side which has a low quality parser to obtain transferred parse trees. We then combine the transferred parse trees with the original low quality parse trees. In our tree-to-tree MT experiments we have observed that the new combined trees lead to better performance in terms of BLEU score compared to when the original low quality trees and the transferred trees are used separately.

1 Introduction

Depending on whether or not monolingual parsing is utilized, there are about 4 types of machine translation (MT) methods. string-to-string (Koehn et al., 2007; Chiang, 2005), string-to-tree (Galley et al., 2006; Shen et al., 2008), tree-to-string (Liu et al., 2006; Quirk et al., 2005; Mi and Huang, 2008), and tree-to-tree (Zhang et al., 2008; Richardson et al., 2014).

Though the tree-to-tree system that employs syntactic analysis for both source and target sides seems to be the best intuitively, in practice, two good quality parsers are difficult to acquire which affects the translation quality which is sensitive to the differences in syntax annotation. In many cases, one parser is of a much higher quality than the other since one of the languages is easier to parse and has a well annotated treebank. In case of Japanese to Chinese translation, Japanese is easier to parse than Chinese and Japanese parsers typically make fewer mistakes compared to Chinese parsers.

In this paper, we explore a method which relies on using parallel text for transferring syntactic knowledge from a high quality (HQ) parser to a low quality (LQ) parser using alignment information(Ganchev et al., 2009; Hwa et al., 2005). Henceforth we shall refer to Japanese as HQ or HQ side, indicating that it is the language which has a high quality parser. Conversely Chinese will be referred to as the LQ or LQ side since the Chinese parser is of a relatively lower quality and makes a number of parsing mistakes. One advantage is that the transferred parse information will possibly be more similar to the other side's parse. This will also reduce the parsing error on the LQ side and unify the syntactic annotation on both sides.

This idea has been proposed before, but not much has been done in the case of dependency-based tree-to-tree SMT system, which is the setting of this paper. Furthermore, this method results in two types of trees on the LQ side: The original LQ tree and the tree transferred from the HQ side. These two trees have their individual strengths: The transferred tree could be more precise compared to the original LQ one in theory, but it is much more sensitive to

29th Pacific Asia Conference on Language, Information and Computation: Posters, pages 80 - 88
Shanghai, China, October 30 - November 1, 2015

alignment errors or bad parallel sentences (not direct translation). To address these problems, previous studies simply apply language dependent rules to the transferred trees, for example in English *have* and *be* must have an object modifier. In this paper we consider combining these two trees and get improved results. We show in our experiments that combining the LQ-parsed trees with the transferred trees yield better translation results rather than only using them individually.

2 Related Work

2.1 Syntax Transfer for Non-MT Task

There are many previous works describing methods to improve the performance of NLP tasks for a resource poor language by using a related resource rich language (mainly English). Amongst these the ones which employ methods which transfer information perform better than unsupervised methods. (Das and Petrov, 2011) describe an approach for inducing unsupervised part-of-speech tags for languages that have no labeled training data. (Jiang et al., 2010) show a transfer strategy to construct a constituency parser. (Ganchev et al., 2009) present a partial, approximate transfer through linear expectation constraints to project only parts of the parse trees to the low resource language side.

However, improving monolingual parsing accuracy does not directly lead to higher MT performance, as it does not address the annotation criteria difference problem.

2.2 Syntax Transfer for MT Task

For MT tasks, most transfer based works assume that the source side has a poor tree-bank, a bad quality parser and/or little training data (parallel corpus to be precise). For phrase-based models, (Goto et al., 2015) proposed a cross transfer pre-ordering model which employs a target-language syntactic parser without requiring a source language parser. For tree based models, most works focus on the fact that they have no source side parse tree and create a parse tree with transfer. (Jiang et al., 2010) showed that a transferred constituent tree parser leads to results that are comparable with those obtained using a supervised tree parser. (Hwa et al., 2005) worked on transferring the results of an English parser to a resource poor language and applied post-transfer transformations like *An aspectual marker should modify the verb to its left*.

These previous works typically assume that only one out of the two languages has a parser. In practice however, the resource poor language has a parser, whose quality is worse than that of a resource rich language's parser. In this work, we consider such a setting. If we combine the transferred parse tree with the original parse tree, the performance should improve.

3 The Difficulties of Tree-to-Tree Approaches

A tree-to-tree model is the most natural in the MT scenario because it respects syntax. However for tree-to-tree translation models, we need both a good target-side parser and a good source-side parser. Even if the parsers are of high quality, we may have problems due to different syntax annotation criteria. Our main objective is thar we want to improve the translation quality a dependency-based tree-to-tree system such as kyotoEBMT (Richardson et al., 2014). The differences between the tree-to-string model and tree-to-tree models are shown in Fig 1 and Fig 2.

In Fig 2, it can be seen that the target side parsing error affects tree-to-tree system's search space. It shows that a relatively low quality parser limits the system. Unlike the tree-to-string approach, the tree to tree approach is affected by the parsing error on both sides. Another problem is shown in Fig 3, where the coordination relation 'wo he ta (my and his)' in the Chinese parse tree (bottom side) is annotated as 'siblings', but in Japanese side (top side) these three words 'watashi oyobi kareno (my and his) have a parent-child relation. Even if both of them are correct in their own tree-banks, tree-to-tree decoder won't match a sibling relation pair with a parent-child relation pair as is explained in Fig 2.

A transfer based method should solve these problems, because it makes LQ trees more similar to the HQ trees. Transferring HQ side syntax relation not only fixes the LQ side parsing error, but also unifies the syntax annotation criterion.

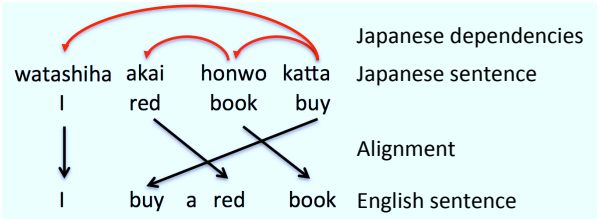

Figure 1: For tree-to-string system, it searches the case *watashi katta* (I buy) *akai honwo* (red book) because there are dependency pairs in source parse tree and does not search the case *watashi akai* (I red) because it is not a pair in source parse tree

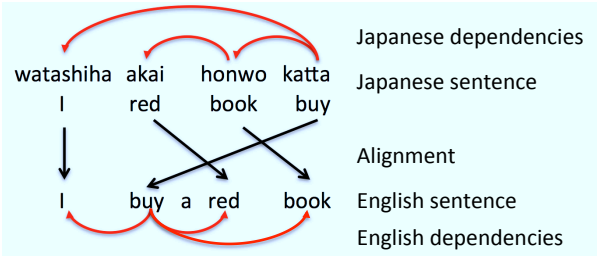

Figure 2: For tree-to-tree system, it searches the case *watashi katta* (I buy) because it is a dependency pair in source parse tree and target parse tree. It does not search the case *watashi akai* (I red) because it is not a dependency pair in source parse tree neither in target parse tree. Unlike Fig.1, it does not search *akai honwo* (red book) because although it is a dependency pair in source side, it is disconnected in target side

4 Transfer of Syntactic Dependencies

4.1 Overview and Notation

This section gives an overview of our approach. In the description below we use dependency tree structure. To represent dependency we will use the following notation $tree = \{(i,j), \cdots\}$. It means that the word in position j is the parent of the word in position i; we use $(i,-1)$ to represent that i is the root. To represent alignment we will use the following notation $a = \{i\text{-}j, \cdots\}$. It means that the word in the source side position i is aligned to the word in the target side position j.

By this we mean that, we first transfer the entire high quality (HQ) dependency tree $Tree_S$ to low quality (LQ) side which replaces the original LQ tree $Tree_T^{old}$ to a transferred tree $Tree_T^{new}$ (Section 4.2). For each word w_i in LQ side sentence, parent of w_i is denoted by $w_{i_{newp}}$ after transfer and $w_{i_{oldp}}$ before transfer. This direct mapping method always transfers source side syntax structure to the target side by alignment regardless of the possibility of alignment error or different expression. In practice, if we test $Tree_T^{new}$ in our tree-to-tree system, we get about a 2 point reduction in BLEU. As a result, the second step contains a backstep which makes some $w_{i_{newp}}$ back to $w_{i_{oldp}}$ for keeping the sentence projective. Notice that the original monolingual dependency tree is always projective (Most parsers give projective results). The worst case is to let all $w_{i_{newp}}$ point back to $w_{i_{oldp}}$, i.e. to keep the dependency tree structure unmodified (Section 5.2). In HQ-LQ (input-side has high-quality parser) MT task, for each training parallel sentence, creating a combined tree for LQ side is enough. However in the LQ-HQ (the input-side has Low-Quality parser) MT task, transferring training data is not enough. For the input LQ side sentence, it makes no sense to use the original monolingual LQ parse tree. Thus, for the third step, we re-train a new LQ side parser using the combined data (Section 6).

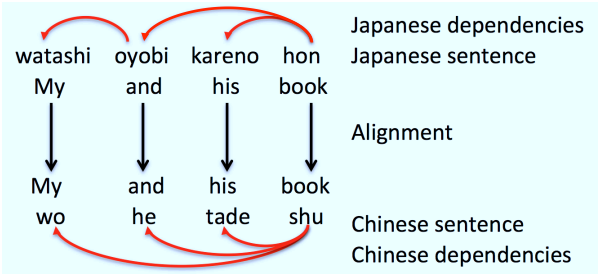

Figure 3: An example of different syntax annotation

4.2 Transfer Dependencies

Intuitively, mapping a high accuracy syntax parser to a low accuracy syntax parser will lead to better performance and the success of this approach depends on the quality of word alignment on a parallel corpus. This Direct Mapping (DM) can be formalized as below:

Given a sentence pair (S,T) where $S = s_1 s_2 ... s_n$ is a sentence of HQ parse side and $T = t_1 t_2 ... t_n$ is a sentence of LQ parse side, a dependency tree for S denoted as $Tree_S = \{(s_i, s_j)...\}$ which has been mentioned before. The new LQ parse tree $Tree_T^{new}$ is transferred from HQ parse tree $Tree_S$ as follows.

- one to one case: If s_i aligns to a unique t_j, s_x aligns to a unique t_y, and $(s_i, s_x) \in tree_S$, push (t_j, t_y) into $tree_T^{new}$.
- one to many case : If s_i aligns to $t_x..t_y$, then

take one of them as representative, a tree based alignment should let $t_x..t_y$ be a treelet. We take the root of $t_x..t_y$ as representative and then perform the same steps as in the one-to-one case. For the node t_z other than representative t_r, we simply push (t_z, t_r).

- many to one case : If $s_i..s_j$ aligns to t_x, like the one to many case, take one of them as a representative. A tree based alignment should let $s_i..s_j$ be a treelet, take the root of $s_i..s_j$ as the representative and then perform the same steps as in the one-to-one case.

- many to many case: Reduce this to one-to-many and many-to-one cases, i.e. both side select a representative and then perform the same steps as in the one-to-one case.

- unaligned case (HQ side): If s_i is an unaligned word, just treat it as non-existent and link two sides of s_i. More specifically, if s_i is not aligned, $(s_i, s_j) \in Tree_S$. and $(s_k, s_i) \in Tree_S$, push (s_k, s_j) into $Tree_S$.

- unaligned case (LQ side): If t_i is an unaligned word, just push $(t_i, t_i + 1)$ or $(t_i, t_i - 1)$ into $Tree_T^{new}$.

Direct Mapping (DM) gives us a simple way of obtaining dependency tree parsing and there are many works that investigate these kinds of mapping and show that they work well. We, however, still want to test the efficacy of direct mapping. We train a new parser based on transferred data and show that it leads to lower parsing score (Section 7.1). [1] This shows that the DM approach won't directly improve the parsing accuracy.

5 Post Transfer Transformations

5.1 Error Analysis

We check the difference between the correct anotation and transferred trees. There are three differences.

- different annotation criterion on both sides (false error): Because the labels were designed for a monolingual scenario, there is always a difference in the annotation criteria. For example: *Shanghai Industrial Technology school* on Ja and Zh. The gold standard data looks like: $Tree_S = \{(0,1)(1,2)(2,3)(3,-1)\}$ $Tree_T^{old} = \{(0,3)(1,3)(2,3)(3,-1)\}$, assume alignment is $a = \{0\text{-}0, 1\text{-}1, 2\text{-}2, 3\text{-}3\}$. By Direct Mapping, $Tree_T^{new} = \{(0,1)(1,2)(2,3)(3,-1)\}$ is the same as the one on the Ja side but completely wrong when comparing it to the gold standard tree. The reason is Ja side tends to let words of compound nouns be child-parent pair and Zh tends to let words of compound nouns be sibling. We call it false error, though the dependency score will decrease by comparing it with gold standard data, it actually helps tree-based translation system retrieve this Chinese compound noun better than before because its structure now is much more similar to the Japanese side.

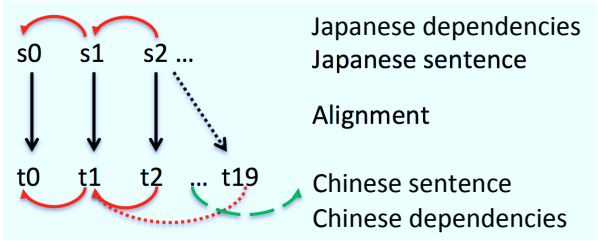

Figure 4: This is an example of alignment error where the solid line is the correct alignment and the dashed line is the wrong alignment. It shows that incorrect alignments lead to parsing error by DM

- alignment error (true error): The direct mapping method is highly dependent on alignment. If alignment is incorrect, direct mapping which uses this erroneous alignment gives a wrong dependency result. See Fig.4 for an example. This is quite critical; originally a tree-to-tree MT system might reject this wrong alignment by the distance on tree with feature or criterion like a child-parent pair in source side should translate to a child-parent pair in target side. We now force a child-parent pair to align to a child-parent pair using DM which will prevent the tree distance criterion from influencing the translation accuracy.

- different expression (true error): Sometimes

[1] The transferred tree which has been tested in Section 7.1 is the combined tree after the method in Section 5.2 is applied

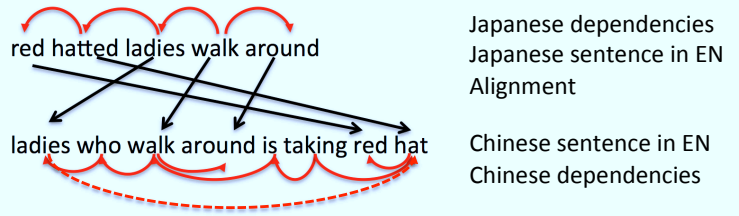

Figure 5: This is an example of a different expression where again the solid line is the correct parse and the dashed line is the wrong parse caused by DM.

the translation is not a direct translation, for example a Ja sentence with the meaning 'ladies wearing red hats walk around' will sometimes be translated to Zh with meaning 'ladies who walk around are taking red hats '. Even though the alignment is perfect, the direct mapping method makes the Zh words for 'ladies', 'red' and 'hat' take on child-parent relations which, ofcourse, is a parsing error as shown in Fig.5.

5.2 Keep the Tree Projective

Error analysis for true and false errors allowed us to revise our approach to incorporate a criterion *projectivity* that can distinguish a good mapping between the true and false errors discussed above. *projectivity* is a property of parse tree which means there shouldn't be any crosses in the tree structure. For example: $Tree_T^{new} = \{(0,2)(1,3)(2,3)(3,-1)\}$ is not projective. To be more precise, for two child-parent pairs $(a,b),(c,d)$, we denote the interval $[a,b](if(a > b), swap(a,b))$ as $span(a,b)$. Here statement ($span(a,b)$ cross $span(c,d)$) $equal(c \in [a,b])$ and $(d \notin [a,b])$. Notice that the root node is denoted as $(c,-1)$, $(d \notin [a,b])$ is always true. Many alignment errors cases can be detected by the property of projectivity. For example, in Fig.4, a parent-child pair $(19,2)$ created by erroneous alignment looks very strange and has a high probability of having crosses with other dependencies like the green dependency in the figure.

Many different expression cases can also be detected by the property of projectivity. Review the example Fig.5 in the previous section. Here child-parent pair (7,0)*hat,ladies* is a long distance dependency relation which crosses with other dependency pairs like (5,-1)*talking root*.

Although annotation criterion are different, both of them are reasonable for showing dependency relations. Thus a mapping one to another still keeps the tree projective. Consider the example mentioned above, *Shanghai Industrial Technology school* on JP and ZH. Original monolingual Chinese dependency tree $Tree_T^{old} = \{(0,3),(1,3),(2,3)(3,-1)\}$ and transferred tree $Tree_T^{new} = \{(0,1),(1,2),(2,3),(3,-1)\}$ are BOTH projective.

Thus keeping the tree projective prevents many projection errors. Notice that $Tree_T^{old}$ is always projective. We introduce a back search method which makes some words relations in $Tree_T^{new}$ back to relations in $Tree_T^{old}$. The pseudo-code shown below:

Algorithm 1 Back Searching

```
len ← Tree_T^new.length
for i = 1..len do
    for j = 1..len do
        find x s.t.  (t_i, t_x) ∈ Tree_T^new
        find y s.t.  (t_j, t_y) ∈ Tree_T^new
        if span(i,x) cross with span(j,y) then
            find z s.t.(t_i, t_z) ∈ Tree_T^old
            (t_i, t_x) ← (t_i, t_z)
            find z s.t.(t_j, t_z) ∈ Tree_T^old
            (t_j, t_y) ← (t_j, t_z)
        if a loop is created then undo
```

For a non-projective part, which means it creates cross in the dependency tree, We trust the original monolingual dependency tree and for projective part we retain the direct mapping result. More precisely, with $span(i,x)$ cross with $span(j,y)$, we find z that $(t_i,t_z) \in Tree_T^{old}$ and substitute (t_i,t_x) with (t_i,v_z). This operation actually moves t_i's parent back to the state before mapping. Doing the same thing to t_j, we find z that $(t_j,t_z) \in Tree_T^{old}$ and substitute (t_j,t_y) with (t_j,t_z).

For non-aligned word t_i, our strategy is a little different. In the direct mapping process, we did not change the non-aligned word's dependency because

we didn't have any information to decide its parent. When we encounter non-projecivity on t_i, we simply change t_i's parent to t_j which ensures projectivity. Changing t_i's parent to t_j solves the cross between $span(i, x)$ and $span(j, y)$. In any case we try to retain its original parent as much as possible.

Sometimes back searching like above leads to loops in the tree. This happens when other non-projective parts should be solved before this part. We check whether a loop exists in the tree after each back operation to t_i,t_j and undo the modification if so. We run several iterations of the procedure mentioned above till it reaches the worst case which means reverse $Tree_T^{new}$ is the same as $Tree_T^{old}$.

It is not so simple to test the effectiveness of *projectivity* on tasks other than MT. We manually check the percentage of errors our method has solved by manually evaluating 50 sentences (Section 7.2).

6 Re-train a New LQ Side Parser

Using the word alignments and original monolingual dependency trees, we successfully create combined trees using the parallel training corpus. As we have mentioned before, this is enough for the HQ-LQ MT task but still a bit not enough for LQ-HQ MT task. For a LQ side sentence as an input, it makes no sense to use the original monolingual LQ side parser which now has a different annotation criterion since it uses the combined trees in training corpus. Considering we have abandoned the original dependency tree, we now regard the combined trees as 'golden data' and train a new model with these 'golden data' using a LQ side parser. After that, for an input sentence, we utilize the new parser rather than the original one.

7 Experiment

7.1 Parsing Accuracy

We conducted a Chinese parsing experiment on scientific domain. The Chinese parser used in our experiment is the SKP parser (Shen et al., 2012).[2] As the baseline parser, we trained SKP with the Penn Chinese treebank version 5 (CTB5) containing 18k sentences on news domain, and a in-house treebank which contains about 10k sentences in scientific domain, with default parameters. The new combined parser that we proposed used the training data obtained from the ASPEC Ja-Zh parallel corpus,[3] containing 670k sentences. We used a Japanese parser KNP (Kawahara and Kurohashi, 2006)[4] and the baseline SKP to automatically parse these sentences. We then created combined Chinese dependency trees.[5] Finally, we trained a new parser using these combined Chinese dependency trees with the same parameters.

As test data we used an additional 1k sentences from our in-house treebank. Table 1 shows the results of these two parsers.

Parser	UAS	Root-Accuracy
Baseline	0.7433	0.6950
Combined	0.5890	0.6140

Table 1: Parsing accuracy

Because we deliberately ignored the annotation criterion problem and labelled the dependencies of non-aligned words quite freely, the decrease of accuracy is not surprising (according to us).

7.2 Projectivity for Solving True Errors

In addition, we evaluated our new parse trees in another way. There are two true errors for the DM approach which have been discussed in Section 5.1, some of them should be rejected by the property of *projectivity*, some of them could not. We randomly selected 50 sentences to check how many errors of these two error types have been solved.

Case	Solve rate
Alignment error	90%
Different expression	55%

Table 2: The percentage of alignment error and different expression which has been solved by projectivity

The results are shown in Table 2. We can see that projectivity is an effective way of addressing the alignment error problem, but the *different expression*

[2]https://bitbucket.org/msmoshen/skp-beta

[3]http://lotus.kuee.kyoto-u.ac.jp/ASPEC/

[4]http://nlp.ist.i.kyoto-u.ac.jp/EN/index.php?KNP

[5]We only obtained 419k combined trees, as the rest were unchanged sentences.

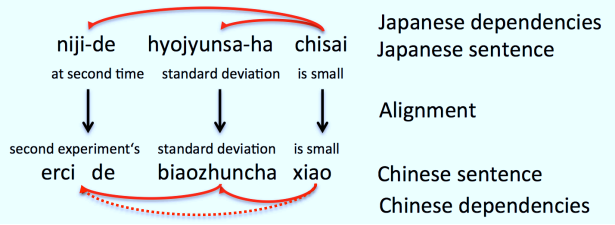

Figure 6: An example of different expression can't be detected by projectivity, solid line is the correct parse and the dashed line is the wrong parse caused by DM.

problem is much harder to detect. Fig 6 shows an example of the different expression problem. A possible solution is make some language based rules like *the word before possessive's parent is the word after it*.

7.3 Translation

We conducted experiments for Japanese-to-Chinese (Ja-Zh) and Chinese-to-Japanese (Zh-Ja) translation. For both tasks, we used the ASPEC Ja-Zh parallel corpus as training data. We used 2,090 and 2,107 additional sentence pairs for tuning and testing, respectively. In our experiments, we compared the MT performance of our proposed projection method with the baseline parser. We used a tree-to-tree system KyotoEBMT for our experiments (Richardson et al., 2014).[6] To parse the Chinese and Japanese sentences, we again used SKP and KNP, respectively.

In order to test our combined tree approach, we substituted the original SKP parsing results to combined parse trees. We also tested the direct mapping method which simply transfers the Japanese parse trees to the Chinese side. In Ja-Zh task, we trained a new Zh parser by using the combined parse trees.

System	Ja-Zh	Zh-Ja
Moses	27.25	33.94
KyotoEBMT	29.08	35.10
Direct map	27.28	33.23
combined map	29.89*	35.59*

Table 3: BLEU scores for ASPEC JA-ZH and ZH-JA. (* denotes that the result is significantly better than 'KyotoEBMT' at $p < 0.05$)

Table 3 shows the results. For reference, we also show the MT performance of the phrase based system Moses, which is based on the open-source GIZA++/Moses pipeline (Koehn et al., 2007). We also conduct significance tests which were performed using the bootstrap resampling method proposed by Koehn (2004). The Direct mapping method which does not use the Chinese parser decreases the MT performance. The combined tree method works pretty well. In the Ja-Zh direction, it gets a 0.8 BLEU score improvement and 0.5 in the Zh-Ja direction. The biggest problem for the Zh-Ja direction is that we have to feed automatic data to SKP for training a new parser. SKP is designed to work with manually annotated training data (gold data set) but not automatically generated training data. The best evidence is that, if we parse a training sentence with this parser, the result is quite different with the original training data. For Ja-Zh direction, it is quite straightforward, we just combine SKP and KNP parsing results to create a Ja like Zh dependency tree for each training sentence. Combined tree increased the BLEU score but decreased parsing accuracy since parsing accuracy is tested on monolingual test data. Lingusitic parse tree structures are not the most appropriate for a tree-to-tree MT system.

8 Conclusion and Future Work

In this paper, we have proposed a method to use both source side and target side parse trees to create a combined parse tree which improves the BLEU score on a tree-to-tree MT system. Transferring parsing information from a relatively high accuracy parser on the source language side to the target language side with a relatively low accuracy parser constrained by *projectivity* performs well. It not only fixes the parsing error of the low accuracy side, but also addresses the problem of different annotation criteria on both sides.

[6] http://nlp.ist.i.kyoto-u.ac.jp/EN/index.php?KyotoEBMT)

We used automatically created, combined parse tree to train a new parser by using an existing parser. It is not very appropriate because parsers are designed to train on gold standard data and not automatic data. Though it leads to some positive results, we think that the quality could be further improved. Instead of re-parsing the parser, we could also save the different parts in the combined parse trees, and apply them to the input sentences. This could be much more efficient and logical than re-parsing but a bit difficult when considering issues such as how to save and how to apply them to the input sentences.

References

David Chiang. 2005. A hierarchical phrase-based model for statistical machine translation. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 263–270, Ann Arbor, Michigan, June. Association for Computational Linguistics.

Dipanjan Das and Slav Petrov. 2011. Unsupervised part-of-speech tagging with bilingual graph-based projections. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 600–609, Portland, Oregon, USA, June. Association for Computational Linguistics.

Michel Galley, Jonathan Graehl, Kevin Knight, Daniel Marcu, Steve DeNeefe, Wei Wang, and Ignacio Thayer. 2006. Scalable inference and training of context-rich syntactic translation models. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 961–968, Sydney, Australia, July. Association for Computational Linguistics.

Kuzman Ganchev, Jennifer Gillenwater, and Ben Taskar. 2009. Dependency grammar induction via bitext projection constraints. 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*, pages 369–377, Suntec, Singapore, August. Association for Computational Linguistics.

Isao Goto, Masao Utiyama, Eiichiro Sumita, and Sadao Kurohashi. 2015. Preordering using a target-language parser via cross-language syntactic projection for statistical machine translation. *ACM Trans. Asian Low-Resour. Lang. Inf. Process.*, 14(3):13:1–13:23, June.

Rebecca Hwa, Philip Resnik, Amy Weinberg, Clara Cabezas, and Okan Kolak. 2005. Bootstrapping parsers via syntactic projection across parallel texts. *Natural language engineering*, 11(03):311–325.

Wenbin Jiang, Yajuan Lv, Yang Liu, and Qun Liu. 2010. Effective constituent projection across languages. In *Coling 2010: Posters*, pages 516–524, Beijing, China, August. Coling 2010 Organizing Committee.

Daisuke Kawahara and Sadao Kurohashi. 2006. A fully-lexicalized probabilistic model for japanese syntactic and case structure analysis. In *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*, pages 176–183, New York City, USA, June. Association for Computational Linguistics.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 177–180, Prague, Czech Republic, June. Association for Computational Linguistics.

Yang (1) Liu, Qun Liu, and Shouxun Lin. 2006. Tree-to-string alignment template for statistical machine translation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 609–616, Sydney, Australia, July. Association for Computational Linguistics.

Haitao Mi and Liang Huang. 2008. Forest-based translation rule extraction. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 206–214, Honolulu, Hawaii, October. Association for Computational Linguistics.

Chris Quirk, Arul Menezes, and Colin Cherry. 2005. Dependency treelet translation: Syntactically informed phrasal SMT. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 271–279, Ann Arbor, Michigan, June. Association for Computational Linguistics.

John Richardson, Fabien Cromières, Toshiaki Nakazawa, and Sadao Kurohashi. 2014. Kyotoebmt: An example-based dependency-to-dependency translation framework. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 79–84, Baltimore, Maryland, June. Association for Computational Linguistics.

Libin Shen, Jinxi Xu, and Ralph Weischedel. 2008. A new string-to-dependency machine translation algorithm with a target dependency language model. In *Proceedings of ACL-08: HLT*, pages 577–585, Colum-

bus, Ohio, June. Association for Computational Linguistics.

Mo Shen, Daisuke Kawahara, and Sadao Kurohashi. 2012. A reranking approach for dependency parsing with variable-sized subtree features. In *Proceedings of the 26th Pacific Asia Conference on Language, Information, and Computation*, pages 308–317, Bali,Indonesia, November. Faculty of Computer Science, Universitas Indonesia.

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: HLT*, pages 559–567, Columbus, Ohio, June. Association for Computational Linguistics.