# Bilingually Induced Clause Parser for Tree-based Translation

**Wen Zhang**[1] **Qiuye Zhao**[1] **Wenbin Jiang**[1] **Qun Liu**[2,1]
[1]Institute of Computing Technology, Chinese Academy of Sciences
{zhangwen,zhaoqiuye,jiangwenbin,liuqun}@ict.ac.cn
[2]ADAPT Centre, School of Computing, Dublin City University

## Abstract

Tree-based machine translation models possess the property of long distance reordering by incorporating the syntactic annotations of parse trees from both or either side(s) of the bitext. However, with the increasing of sentence length, the parsing accuracy usually goes down, which will further drop the performance of tree-based machine translation. To alleviate it, we choose to translate clauses other than entire sentences, while the challenge is to split the source sentences appropriately. In this paper, we propose a novel approach to induce clause parser from word-aligned parallel corpora, and test its effectiveness on tree-to-string machine translation. Experiments on multi translation tasks show that our approach outperforms previous rule-based approaches which mainly depend on punctuations and pre-defined rules. More importantly, our approach works much better than rule-based method on text without punctuations.

## 1 Introduction

Tree-based statistical machine translation models take advantage of the syntactic structures of either the source language (Liu et al., 2006; Huang et al., 2006; Xie et al., 2011), the target language (Galley et al., 2006; Shen et al., 2010) or both (Liu et al., 2009a; Mi and Liu, 2010). However, the performance of parsing, by which we compute syntactic structures from raw sentences, decreases as the length of sentences increases. It is a natural idea to split a source sentence into clauses, translate each clause independently and linearly combine the translated clauses in the target language to form the final translation output of the source sentence. In previous work (Xiong et al., 2009), this initial idea has been approached by splitting source sentences with punctuations and pre-defined rules. Despite the limitation of such rule-based methods that employs source language information only, splitting sentences into clauses for MT has been shown to be a promising track to explore.

We propose a novel approach for splitting sentences into clauses with bilingual constraints, which are automatically extracted from parallel data, thus expected to serve machine translation better. In particular, we take tree-to-string machine translation as a case study. More specifically, we extract structural constraints from word aligned parallel data for both source and target language. For the source language, we train a parser to output structures that are flatter than normal syntactic trees but embody clauses information. For the target language, we train a language model from clauses data that contains ngrams from clauses only other than whole sentences. We extract translation rules from word-aligned clauses other than word-aligned sentences. For decoding, we first parse the source language with the clause parser, then translate each clause with clause models, and linearly combine the translation outputs as the final output of the source sentence.

As shown by experiments, by employing the bilingually induced clause parser for the source language and the language model trained from the target clauses, we observe improved translation performance on various datasets. And our method outperform the baseline that uses pre-defined rules and punctuations for sentence splitting by BLEU points (Papineni et al., 2002). Especially, when the source language does not present with punctuations, for example, the direct output from speech recognition, the baseline doesn't apply at all, but we can still induce clause parsers from aligned data without punctuations. Experiments show that our approach achieves much better translation performance on the direct output from speech recognition by BLEU.

In the remainder of this paper, we first formalize our approach of bilingually inducing clause boundaries recognition in Section 2. Then we describe how to build our clause parser for machine translation in Section 3. Next, we make a brief overview of related work in Section 4 and conduct experiments to validate the effectiveness of the proposed approach in Section 5. Finally, we conclude and provide directions for future work in Section 6.

## 2 Bilingually Inducing Clause Boundaries Recognition

We propose an approach to automatically learn boundary information from word-aligned parallel data, which are expected to maintain translation blocks. More specifically, we design a top-down algorithm that splits clauses recursively, based on the word weighted alignment matrix (Liu et al., 2009b).

Given a pair of aligned clauses$(F, E)$, which are initially the pair of two aligned sentences, we define a 'dividable degree' $d(x, y)$ to measure whether it is suitable to cut the aligned clauses further at some alignment point $(x, y)$, where the $x_{th}$ word in the source sentence is aligned with the $y_{th}$ word in the target sentence. We consider a cut is suitable only when the two new pairs of aligned clauses generated by this cut is consistent with the word alignment. Let $F_{\_}$ denote the left boundary of the source clause $F$, ${}_{\_}F$ the right boundary of the source clause $F_{\_}$, and respectively $E_{\_}$ and ${}_{\_}E$ for the boundaries of target clause $E$. The new pairs of clauses generated by a cut $(x, y)$ are denoted as $(F_{\_} : x, E_{\_} : y)$ and $(x : {}_{\_}F, y : {}_{\_}E)$. We compute $d(x, y)$ by the probability computed from the alignment matrix as follows,

$$d(x, y) = p(F_{\_} : x, E_{\_} : y) \times p(x : {}_{\_}F, y : {}_{\_}E).$$

where the probability of two clauses being well-aligned can be computed as follows,

$$p(F_c, E_c) = \frac{\sum_{i \in F_c, j \in E_c} A(i, j)}{\sum_{i \in F, j \in E_c} A(i, j)} \times \frac{\sum_{i \in F_c, j \in E_c} A(i, j)}{\sum_{i \in F_c, j \in E} A(i, j)}$$

where $A(i, j)$ is a probability which measures the confidence that the $i_{th}$ word in source sentence and the $j_{th}$ word in target sentence are aligned to each other (assume index $i$ and $j$ start from 1), here we treat the probability as a score of an alignment point $(i, j)$ in the weighted alignment matrix, $F_c$ and $E_c$ indicate successive clause segments in sentences $F$ and $E$. $p(F_c, E_c)$ can be treated as the consistency degree between the correspondence of $F_c$ and $E_c$ and the word alignment inside $F_c$ and $E_c$. A weighted alignment matrix gives the probability to align any two words in a pair of parallel sentences, and an excerpt of a weighted alignment matrix is depicted in Figure 1, where each element $M[i, j]$ in the matrix represents the probability of the $i_{th}$ word in source sentence and the $j_{th}$ word in target sentence are aligned to each other, if there is no alignment between the $i_{th}$ word in source sentence and the $j_{th}$ word in target sentence, $M[i, j]$ is equal to 0. Here $A(i, j)$ equals $M[i, j]$.

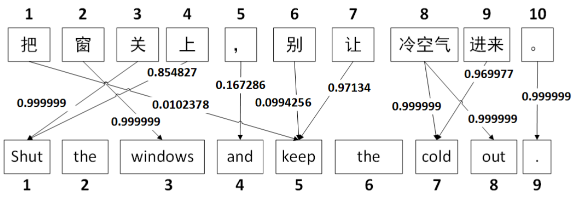

Figure 1: An excerpt of a weighted alignment matrix.

The 'dividable degree' of a 'cut' as defined above is a real number in the range of 0-1, designed to measure whether the two new pairs of clauses generated by a cut form well-aligned clauses. Starting from the pair of aligned sentences as the initial pair of aligned clauses, for each clause pair, we only make one cut whose dividable degree is the highest as computed relative to the current clause pair. And recursively, for each new pair of clauses we make further cuts when a cut's dividable degree is higher than some pre-defined threshold. We demonstrate such cuts by an example in Figure 2 and summarize the recursive algorithm in Algorithm 1. THRESHOLD denotes the pre-defined cut threshold and $R$ denotes a rules set which contains essential punctuations and words which often act as cut positions.

## 3 Building Clause Parser for MT

Given the bilingual clauses that are extracted from aligned parallel data as described above, as shown by step ① in Figure 3, we can train clause parsers for both the source and the target languages which

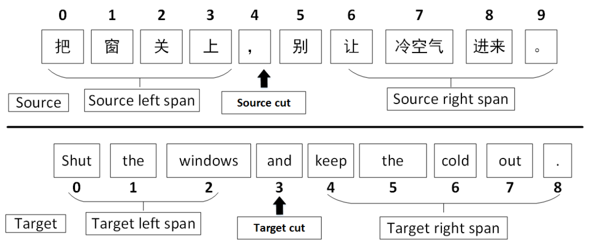

Figure 2: An example of cuts in clauses.

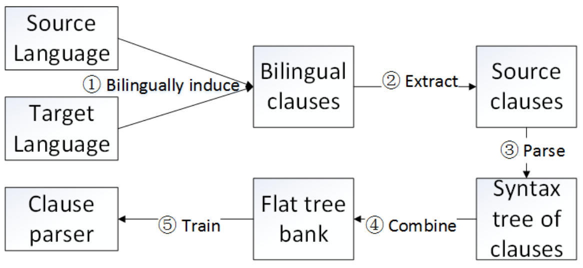

Figure 3: Training clause parser.

**Algorithm 1** The recursive algorithm to generate bilingually aligned clauses.

```
 1: procedure SENTCUTTER($Lf[b_1 : e_1]$, $Le[b_2 : e_2]$)
 2:     [$pos_i$,$pos_j$,$prob$] ← BestCut($Lf[b_1{:}e_1]$, $Le[b_2{:}e_2]$)
 3:     if $prob$ < THRESHOLD then
 4:         output bilingually aligned clauses
 5:     end if
 6:     $F_{lsp} \leftarrow Lf[b_1{:}pos_i]$
 7:     $F_{rsp} \leftarrow Lf[pos_i{+}1{:}e_1]$
 8:     $E_{lsp} \leftarrow Le[b_2{:}pos_j]$
 9:     $E_{rsp} \leftarrow Le[pos_j{+}1{:}e_2]$
10:     SentCutter($F_{lsp}$, $E_{lsp}$)
11:     SentCutter($F_{rsp}$, $E_{rsp}$)
12: end procedure
 1: procedure BESTCUT($Lf[b_1{:}e_1]$, $Le[b_2{:}e_2]$)
 2:     def $pos_i$, $pos_j$, $prob$
 3:     for $i \leftarrow b_1 \ldots e_1$ do
 4:         for $j \leftarrow b_2 \ldots e_2$ do
 5:             if $Lf[i] \not\subset R$ or $Le[j] \not\subset R$ then
 6:                 continue
 7:             end if
 8:             $F_{lsp} \leftarrow Lf[b_1{:}i]$
 9:             $F_{rsp} \leftarrow Lf[i{+}1{:}e_1]$
10:             $E_{lsp} \leftarrow Le[b_2{:}j]$
11:             $E_{rsp} \leftarrow Le[j{+}1{:}e_2]$
12:             $pOrd = p(F_{lsp},E_{lsp}) * p(F_{rsp},E_{rsp})$
13:             $pCrs = p(F_{lsp},E_{rsp}) * p(F_{rsp},E_{lsp})$
14:             if $pOrd > prob$ then
15:                 $pos_i \leftarrow i$, $pos_j \leftarrow j$, $prob \leftarrow pOrd$
16:             else
17:                 $pos_i \leftarrow i$, $pos_j \leftarrow j$, $prob \leftarrow pCrs$
18:             end if
19:         end for
20:     end for
21:     return [$pos_i$,$pos_j$,$prob$]
22: end procedure
```

embody clause information. However, we will only need the clause parser for the source language for tree-to-string translation. Steps ② - ⑤ in Figure 3 show the training process. Accordingly, we train a language model for target language from target clauses data.

So as to build training data to train a clause parser for the source language, we parse each source clause extracted from the aligned parallel data as described above, and linearly combine these subtrees as a whole tree of the original source sentence. We label each subtree of a corresponding clause with a new root label SUBROOT, so that the output of such kind of parser embodys clause information and it can be used to split source sentences in test data. Moreover, we delete all the punctuations from training data to mimic speech recognition data, so as to train a parser that is suitable to parse texts that have no punctuations presented.

Besides, we train a language model from clauses of the target language, instead of the standard language model that are computed from entire sentences. In other words, we extract ngrams from target clauses extracted from aligned parallel data, instead of ngrams in the entire sentences. Overall, our clause-based translation model is composed of a language model from target clauses data and a clause parser for the source language.

For test set, we split source sentences into clauses by the trained clause parser, then put them into tree-to-string machine translation system to decode. Getting clause translations, we combine them linearly according to clauses number of each original entire sentence in test set; Then, translation performance is measured by comparing combined translations with reference translations. We give a description of above process in Figure 4.

## 4 Related work

Since the performance of parsing decreases as the length of sentences increases, there are previous work devoted to sentence splitting for machine translation. (Doi and Sumita, 2004) explore a sentence splitting method to boost the speech translation quality of corpus-based Machine Translation systems. (Xiong et al., 2009) propose a method that dividing the source sentence by predefined rules on its syntax tree, which is helpful to improve the performance of machine translation. These work mainly use punctuations and predefined rules for cutting, thus they got their limita-

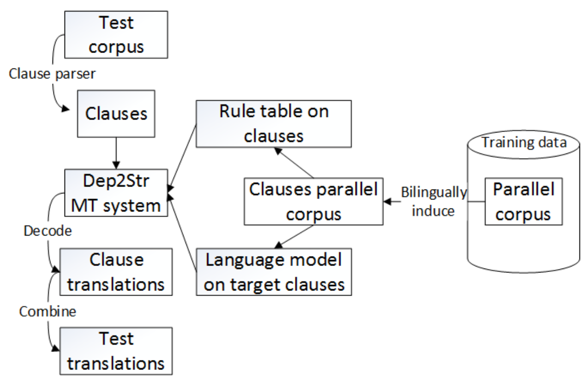

Figure 4: Clauses-based statistic machine translation.

tion, applying to data with specific markers only. On the contrast, we automatically acquire clause parsers from alignment matrix, thus adapts better to the task of machine translation and can be applied to mimicked speech recognition data where punctuations do not present.

Besides, there are emerging work (Zhang et al., 2013; Zhai et al., 2013; Durrett et al., 2012; Liu et al., 2012) that explore to induce syntactic constraints from bilingual data. Compared to parsers trained from monolingual treebank, bilingually induced parsers capture common structures shared by both the source and target languages. This work follows this promising track and contribute new experiments to show that bilingually induced parser adapts better to the machine translation task.

## 5 Experiment and Analysis

We conduct three experiments on different kinds of datasets. We train our clause parser based on the Berkeley Parser (Petrov et al., 2006) and GIZA++ (Och and Ney, 2003) for word alignment. Both the parser and GIZA++ are trained on bilingual whole sentences or bilingual clauses.

For the first experiment, our training corpus consists of 228K sentence pairs from FBIS, and we use NIST 2002 test set as development set, NIST 2003, NIST 2004, and NIST 2005 as test sets. We implement a dependency tree-based translation model (Xie et al., 2011) as our baseline system. For the standard data that include punctuations, we also implement two other baselines for comparison: CBC denotes that we split source sentences by commas only; and RBC refers to (Xiong et al., 2009) which splits clauses with predefined rules and punctuations. As shown in Table 1, our system outperforms the baseline models, in particular with +0.6 BLEU score averagely on test sets than the non-sentence-split baseline.

| Traing data | nist02 | nist03 | nist04 | nist05 |
|---|---|---|---|---|
| Fbis | 0.2526 | 0.2234 | 0.2719 | 0.2236 |
| CBC-Fbis | 0.258 | 0.2256 | 0.2723 | 0.2285 |
| RBC-Fbis | 0.2613 | 0.216 | 0.2756 | 0.2165 |
| BBC-Fbis | 0.2591 | 0.2257 | 0.2763 | 0.2299 |

*CBC: Comma-based sentence splitting
*RBC: Rule-based sentence splitting (LCA)
*BBC: Split by bilingually induced clause parser
*We use NIST 2002 test set as development set

Table 1: Comparing with non-sentence-splitting, by dep2str MT, BBC produce 0.5 bleu higher on NIST 2004, higher than rule-based splitting method

For the second experiment, our training corpus also consists of 228K sentence pairs from FBIS, and we use NIST 2002 test set as development set, NIST 2003, NIST 2004, and NIST 2005 as test sets which is same as the first experiment, we combine English sentence from Xinhua corpus, FBIS corpus and all NIST corpus into a huge English corpus on which we train a 5-gram language model by SRI tool. Rule extraction is done on FBIS, we translate the whole sentences in test set by using Moses tree-based machine translation system which is treated as baseline system; For translating clauses, then we split source sentences and target sentences from Xinhua corpus, FBIS corpus and all NIST corpus by Algorithm 1, then train language model on the clauses from English side. Rule extraction is done on clause pairs from FBIS. Development set is also splitted by Algorithm 1, a pre-trained clause parser is used to split source sentences in test set. Results are shown in Table 2.

For the third experiment, so as to mimic the direct output from speech recognition, we also delete all the punctuations in the standard dataset. On mimicked speech recognition data, we use the same development data and test data. However, all punctuations in the source sentence of test data are removed to mimic the output of speech recognition. As shown in Table 3, our clause parser could make sentence splitting on the output sentence of speech recognition which is appropriate for machine translation, and remarkably improve the performance of machine translation. Approaches of splitting sentences based on pre-defined rules can not have application in this situation well. However, our model shows better adaptability.

| Traing data | nist02 | nist03 | nist04 | nist05 |
|---|---|---|---|---|
| Fbis | 0.402563 | 0.3658 | 0.4106 | 0.3833 |
| BBC-Fbis | 0.431067 | 0.3738 | 0.4203 | 0.3912 |

*BBC: Split by bilingually induced clause parser
*On NIST 2004, BBC produce 1.0 bleu higher
*We use NIST 2002 test set as development set

Table 2: Comparing with non-sentence-splitting, by Moses tree-based MT, BBC produce 1.57 bleu higher on NIST 2006

| Traing data | nist02 | nist03 | nist04 | nist05 |
|---|---|---|---|---|
| Fbis | 0.242 | 0.2054 | 0.2436 | 0.2021 |
| BBC-Fbis | 0.2613 | 0.204 | 0.2449 | 0.2054 |

*BBC: Split by bilingually induced clause parser
*On text without punctuations, BBC produce 0.2 bleu higher
*We use NIST 2002 test set as development set

Table 3: Mimicked speech recognition data

## 6 Conclusion

In this work, we propose a novel approach to induce clause parser from parallel data, which can be used to improve tree-based machine translation. As shown by experiments on both standard NIST datasets and mimicked speech recognition datasets, the proposed approach improves the performance of tree-based translation. In the future, we would like to induce more complex parsers from bilingual constraints, so as to acquire alternative syntactic structures that are more suitable for the task of machine translation than monolingual parsers.

## References

Takao Doi and Eiichiro Sumita. 2004. Splitting input sentence for machine translation using language model with sentence similarity. In *Proceedings of the 20th International Conference on Computational Linguistics*, COLING '04, Stroudsburg, PA, USA. Association for Computational Linguistics.

Greg Durrett, Adam Pauls, and Dan Klein. 2012. Syntactic transfer using a bilingual lexicon. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1–11. 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 the 44th annual meeting of the Association for Computational Linguistics*, pages 961–968.

Liang Huang, Kevin Knight, and Aravind Joshi. 2006. Statistical syntax-directed translation with extended domain of locality. In *Proceedings of AMTA*, pages 66–73.

Yang 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 the 44th annual meeting of the Association for Computational Linguistics*, pages 609–616.

Yang Liu, Yajuan Lü, and Qun Liu. 2009a. Improving tree-to-tree translation with packed forests. In *ACL 2009, Proceedings of the 47th Annual Meeting of the Association for Computational Linguistics and the 4th International Joint Conference on Natural Language Processing of the AFNLP, 2-7 August 2009, Singapore*, pages 558–566.

Yang Liu, Tian Xia, Xinyan Xiao, and Qun Liu. 2009b. Weighted alignment matrices for statistical machine translation. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 1017–1026, Singapore, August. Association for Computational Linguistics.

Shujie Liu, Chi-Ho Li, Mu Li, and Ming Zhou. 2012. Re-training monolingual parser bilingually for syntactic smt. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 854–862. Association for Computational Linguistics.

Haitao Mi and Qun Liu. 2010. Constituency to dependency translation with forests. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1433–1442.

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

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics.

Slav Petrov, Leon Barrett, Romain Thibaux, and Dan Klein. 2006. Learning accurate, compact, and interpretable tree annotation. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics*, ACL-44, pages 433–440, Stroudsburg, PA, USA. Association for Computational Linguistics.

Libin Shen, Jinxi Xu, and Ralph M. Weischedel. 2010. String-to-dependency statistical machine translation. *Computational Linguistics*, 36(4):649–671.

Jun Xie, Haitao Mi, and Qun Liu. 2011. A novel dependency-to-string model for statistical machine translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 216–226.

Hao Xiong, Wenwen Xu, Haitao Mi, Yang Liu, and Qun Liu. 2009. Sub-sentence division for tree-based machine translation. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, pages 137–140, Suntec, Singapore, August. Association for Computational Linguistics.

Feifei Zhai, Jiajun Zhang, Yu Zhou, and Chengqing Zong. 2013. Unsupervised tree induction for tree-based translation. *Transactions of the Association for Computational Linguistics*, 1:243–254.

Jiajun Zhang, Feifei Zhai, and Chengqing Zong. 2013. Syntax-based translation with bilingually lexicalized synchronous tree substitution grammars. *Audio, Speech, and Language Processing, IEEE Transactions on*, 21(8):1586–1597.