Automatic Learning of Parallel Dependency Treelet Pairs

Yuan Ding and **Martha Palmer**
Department of Computer and Information Science
University of Pennsylvania
Philadelphia, PA 19104
{yding, mpalmer}@linc.cis.upenn.edu

Abstract

Induction of synchronous grammars from empirical data has long been a problem unsolved; despite that generative synchronous grammars theoretically suit the machine translation task very well. This fact is mainly due to pervasive structural divergences between languages. This paper presents a statistical approach to learn dependency structure mappings from parallel corpora. The algorithm introduced in this paper extends the dependency tree word alignment algorithm in (Ding, 2003). The new algorithm automatically learns parallel dependency treelet pairs from loosely matched non-isomorphic dependency trees while keeping computational complexity polynomial in the length of the sentences. A set of heuristics are introduced and specifically optimized for the parallel treelet learning purpose using Minimum Error Rate training. As learning parallel syntactic structures is the key step in the automatic learning of a synchronous grammar, the learnt parallel dependency treelet pairs by our approach serve as an important first step of any lexicalized synchronous grammar induction.

1 Introduction

Statistical approach to machine translation, pioneered by (Brown et al., 1990, 1993), perform competitively with more traditional interlingua based approaches, mainly due the ability to leverage large amounts of parallel corpora. However, a major criticism to "pure" statistical MT approaches is that they lack any internal representation of syntax or semantics and simply estimate word to word translation probabilities and sentence reordering probabilities directly from a large corpus of parallel sentences.

In recent years, hybrid approaches, also known as "syntax based statistical machine translation", which aim at applying statistical models to structural data, have begun to emerge. Although it is generally agreed that human language is more than a set of linear strings, and the idea of combining natural language syntax and machine learning methods for MT seems to be plausible, up to today, according to our knowledge, the performance of syntax based statistical MT systems are still outperformed by the state of the art template based statistical MT systems (Och, 2003). We believe that this is due to at least two reasons. (1) The problem of pervasive structural divergences between languages, due to both systematic differences between languages (Dorr, 1994) and the vagaries of loose translations in real corpora. (2) The lack of well-defined formal syntax systems that can be learnt efficiently from empirical data.

(Wu, 1997) introduced a polynomial-time solution for the alignment problem based on synchronous binary trees. (Alshawi et al., 2000) extended the tree-based approach by representing each production in parallel dependency trees as a finite-state transducer. Both these approaches learn the tree representations directly from parallel sentences, and do not make allowance for non-isomorphic structures. (Yamada and Knight, 2001, 2002) modeled translation as a sequence of tree operations transforming a syntactic tree in one language into a string of the second language.

When researchers tried to use syntactic trees in both languages, which are usually derived by automatic parsers trained from treebanks, the problem of non-isomorphism grows when trees in both languages are required to match. (Hajic et al., 2002) allowed limited non-isomorphism in that n-to-m matching of nodes in the two trees is permitted. However, even after extending this model by allowing cloning operations on subtrees, (Gildea,

2003) found that parallel trees over-constrained the alignment problem, and achieved better results with a tree-to-string model using one input tree than with a tree-to-tree model using two.

Most of the above approaches do tree to tree transductions by defining a set of tree operations rather than defining a synchronous grammar formalism and explicitly inducing such a grammar from parallel corpora.

At the same time, grammar theoreticians have proposed various generative synchronous grammar formalisms for MT, such as Synchronous Context Free Grammars (S-CFG) or Synchronous Tree Adjoining Grammars (S-TAG) (Shieber and Schabes, 1990). Mathematically, generative synchronous grammars share many nice properties similar to their monolingual counterparts such as CFG or TAG. If such a synchronous grammar could be learnt from parallel corpora, the MT task would become a mathematically clean generative process. However, up to today, according to our knowledge, the problem of how to induce a synchronous grammar from parallel corpora is still unsolved, mainly due to the inability to handle automatic learning of syntactic structures from large amount of non-isomorphic tree pairs.

This motivates us to look for an algorithm that will automatically learn a certain type of parallel syntactic structures from parallel corpora, without the unrealistic tree isomorphism assumption.

This paper presents a statistical approach to learn dependency structure mappings from parallel corpora. We first define the synchronous tree partitioning operation on parallel dependence trees. Then we introduce the algorithm, which breaks down the sentence-level parallel dependency trees into phrase-level parallel dependency treelets by a series of synchronous tree partitioning operations. The major framework of the algorithm is an extension to the dependency tree word alignment algorithm in (Ding, 2003). The algorithm is executed in an iterative fashion, first assumes free word mapping without any constraints and then gradually adds constraints to word level alignments by breaking down the parallel dependency structures into smaller pieces. In each iteration the algorithm generates a set of more fine-grained treelet pairs.

The major advance of the new algorithm in this paper from that of (Ding, 2003) is the use of syntactic categories in dependency trees and Minimum Error Rate Training to combine various heuristics with parameters specifically optimized for the treelet learning purpose. This allowed us to learn dependency treelet pairs of higher quality.

We first look at the scenario for the dependency structure learning problem and define the synchronous tree partitioning operation in section 2. Then we introduce the iterative framework of the algorithm in section 3. We specify the syntactic constraints for the algorithm in section 4, and analyze different heuristic functions for the algorithm in section 5. Then we introduce the minimal error training method in section 6. The result of the algorithm is provided in section 7.

2 Objective

2.1 Why Dependency Structures?

According to (Fox, 2002), the dependency representation has the best phrasal cohesion properties across languages. The percentage for head crossings per chance is 12.62% and that of modifier crossings per chance is 9.22%.

As stated before, the major purpose for us to learn parallel syntactic structures is future synchronous grammar induction. A grammar based on dependency structures, for this purpose, has the advantage of being simple in formalism yet having the formal generative power equivalent to that of CFG. A higher level look of three different grammar formalisms is given in the following table:

Formalism	CFG	TAG	DG
Node#	*2n*	*2n*	*n*
Lexicalized?	NO	YES	YES
Node types	2	2	1
Operation types	1	2	1

(A comparison of Context Free Grammar, Tree Adjoining Grammar, and Dependency Grammar)

Figure 1.

The simplicity of dependency structures is the key that allowed our algorithm to work. At the same time, dependency structures are inherently lexicalized as each node is one word. In comparison, phrasal structures (Treebank style trees) has two types of nodes, where the lexical items go to the terminals and the word order and phrase scope information goes to the non-terminals.

What is more, statistical parsers (Collins 1999) showed significant performance improvement by using bilexical probabilities, i.e. probabilities of

word pair occurrences. This is what dependency grammars model explicitly.

2.2 Synchronous Partitioning Operation

We introduce the synchronous partitioning operation by first looking at a pseudo-translation example:

- [*Source*] *The girl kissed her kitty cat.*
- [*Target*] *The girl gave a kiss to her cat.*

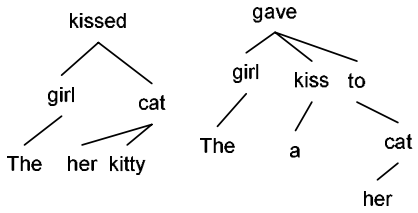


Figure 2

If we examine the node mappings in the above two sentences, we will find that almost any tree-transduction operations defined on a single node will fail to generate the target sentence from the source sentence.

However, suppose that we find the two node pairs lexicalized as "*girl*" and "*cat*" on both sides should be aligned, and hence fix the two alignments. We then partition the two dependency trees by splitting the trees at the fixed alignments. This operation is defined as the synchronous partitioning operation, which generates the following dependency graphs: ((e) stands for an empty node)

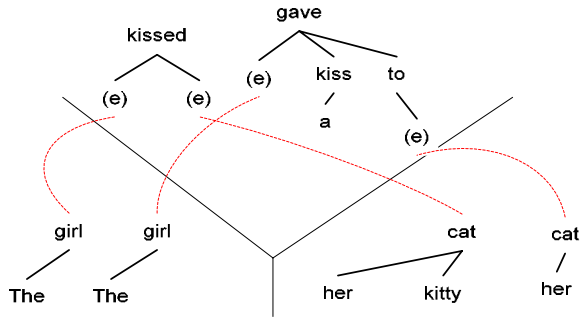


Figure 3

We refer to the three resultant dependency substructures as "treelets". This is to avoid confusion with "subtrees" since treelets don't necessarily go down to every leaf.

Surprisingly, if we view each treelet as a single non-devisable unit for any tree-based transduction, (or an "elementary tree" in a tree based grammar sense), the two sentences showed isomorphism between treelet derivations.

Ideally, our algorithm should generate such treelet pairs as fine-grained as possible, which will finally allow us to generate a synchronous grammar which captures structure variations between languages. We call such a grammar "Synchronous Dependency Insertion Grammar", the detailed specifications for which is not given in this paper due to space limitations.

3 The Framework

The framework of this algorithm is an extension to the word alignment algorithm on parallel dependency structures in (Ding, 2003).

The iterative framework of the algorithm bootstraps between statistical modelling and the parallel tree structures.

Step 0. For each sentence pair e and f, initialize the two trees for e and f as a treelet pair.

Step 1. View each tree as a "bag of words" and train a statistical translation model on all the tree pairs to acquire word-to-word translation probabilities. In our implementation, IBM Model 1 (Brown et al., 1993) is used.

Step 2. For each treelet pair, compute probabilities of the word to word mappings between the two trees for all NON-ROOT nodes. Then use a heuristic function to select the BEST word pair (e_{i^*}, f_{j^*}) for the tentative synchronous partitioning operation.

Step 3. If the word mapping (e_{i^*}, f_{j^*}) selected in Step 2 satisfies certain syntactic constraints, execute synchronous tree partitioning operation on the treelet pair at the word pair (e_{i^*}, f_{j^*}). Hence two new treelet pairs are created, and the old treelet pair is replaced.

Step 4. Go back to Step 1.

As the algorithm walks through the iterations, we have an increasing number of treelet pairs. With certain syntactic constraints in Step 3, the treelet pairs will finally converge to a maximal set.

In IBM Model 1, the translation probability $P(f \mid e)$ has a unique local maximum, so the values in the t table are independent of initialization. In Step 1, the values in t table converge to a unique set of values. So theoretically, the result of this algorithm is only dependent on the choice of

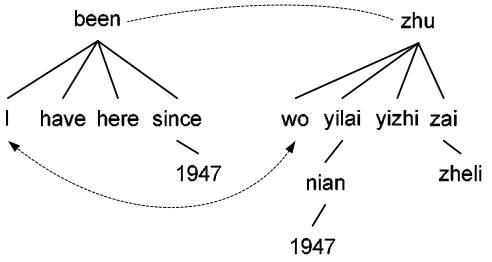
the heuristic function h. The algorithm calls the heuristic function h square to the size of the treelet pair. So the time complexity of the algorithm is $O(n \times m \times T(h))$, where $T(h)$ is the time complexity for the heuristic function and n and m are the sizes of the two treelets.

In our implementation, we only pick tentative word pairs that maximize the word translation probability. In other words, for each $f_j \in f$, we find one tentative word pair (f_j, e_{a_j}) such that $e_{a_j} = \arg\max_{e_i \in e} t(f_j, e_i)$. Thus we reduce the algorithm time complexity to $O(m \times T(h))$, which is linear to the size of the treelet pair.

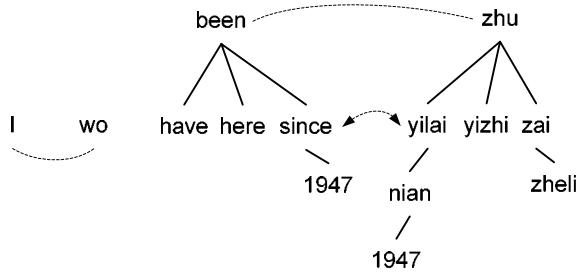
An example of a series of synchronous partitioning of the treelets for three iterations is given below to illustrate the algorithm. The Chinese is given in romanised form.

- [*English*] *I have been here since 1947.*
- [*Chinese*] *Wo 1947 nian yilai yizhi zhu zai zheli.*

[ITERATION 1] Partition at word pair "*I*" and "*wo*"



[ITERATION 2] Partition at word pair "*since*" and "*yilai*"



[ITERATION 3] Partition at word pair "*here*" and "*zheli*"

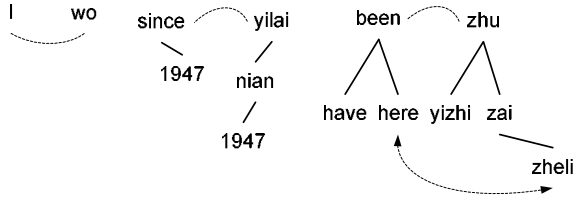


Figure 4. An Example

Partition Assumption:

The correctness of the algorithm relies on the assumption that when a synchronous partitioning operation happens to a pair of nodes, meaning the two nodes are aligned, all their descendents shall align to their descendents. This is however, much weaker than the word level isomorphism assumption. We call this the "partition assumption".

Although violations of such assumption do occur in data, the hope is, since IBM Model 1 is a bag of words model, if we only have operations above and below such violations, the probability training results will be the same.

4 Syntactic Constraints

4.1 Content Words

For each language, we classify the words into two categories: content words and functional words. Linguistically, content words have their own semantics while functional words are mainly used for syntactic purposes. During translation, the functional words have a good chance not being aligned to any word in the other language. Hence we want to constrain all the tentative word pairs used for the synchronous partitioning operation to be a pair of content words.

The content word categories for English and Chinese we used are given below (POS tags in accordance with Penn English Treebank and Penn Chinese Treebank):

- [*English*] JJ JJR JJS NN NNS NNP NNPS PRP PRP$ VB VBD VBG VBN VBP VBZ WP WP$ CD MD
- [*Chinese*] VV VA VC VE NR NT NN PN JJ CD OD

Experiments showed that such syntactic constraints significantly improved the accuracy of the synchronous partitioning operations and hence achieved higher quality resultant treelet pairs.

4.2 Fertility Threshold

Let us suppose we want to create two treelet pairs by fixing the node pair (f_j, e_{a_j}). We want to take the size and topology of the two resulting treelet pairs into consideration. We don't want to have a treelet pair with one node on one side and 9 nodes on the other. Although some of the heuristic func-

tions we used considered this issue, we nevertheless set a fertility threshold to make sure that the resultant treelet pairs have reasonable sizes on both sides. Currently this fertility threshold is set to be 2.0. If any treelet pair resulting from a tentative synchronous partitioning operation has the numbers of the nodes on two sides differing with a ratio larger than the threshold, the tentative synchronous partitioning operation is invalid.

5 Heuristic Functions

The heuristic function in Step 2 takes a tentative node pair (f_j, e_{a_j}) from the two treelets and outputs a certain value which corresponds to confidence in this labelling. Here we introduce five different heuristic functions.

5.1 Inside-Outside Probability

Suppose we have two trees initially, $T(e)$ and $T(f)$. For each synchronous partitioning operation on word pair (f_j, e_{a_j}), both trees will be partitioned into two treelet pairs. The treelets rooted at the original roots is called the "outside treelets" and the treelets rooted at the node pair used for the synchronous partitioning operation is called the "inside treelets", which is shown below:

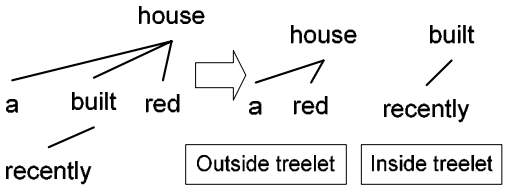


Figure 5. Inside and Outside Treelets
Partition "*a recently built red house*" at node "*built*"

Here we borrow the idea from PCFG parsing, and call the treelets rooted with e_{a_j} and f_j inside treelets $T_{in}(e, e_{a_j})$ and $T_{in}(f, f_j)$. And the other two treelets are called outside treelets $T_{out}(e, e_{a_j})$ and $T_{out}(f, f_j)$. So, we have:

$$P(T(f),\ (f_j, e_{a_j}) | T(e))$$
$$= P(T_{out}(f, f_j) | T_{out}(e, e_{a_j})) \times P(T_{in}(f, f_j) | T_{in}(e, e_{a_j}))$$

The intuition behind this heuristic is that the inside-out side probability is a measure of how well the new treelet pairs created by the tentative word pair (f_j, e_{a_j}) preserve the dependencies. In other words, if the inside-outside probability is large, we expect the nodes in one outside treelet to be aligned to the nodes in the other outside treelet and the same for the nodes in the inside treelets.

We can further decompose this probability. Define the size function to be the number of nodes in a treelet. And let

$$l_{out} = size(T_{out}(e, e_{a_j})),\ m_{out} = size(T_{out}(f, f_j))$$
$$l_{in} = size(T_{in}(e, e_{a_j})),\ m_{in} = size(T_{in}(f, f_j))$$

we have:

$$P(T_{out}(f, f_j) | T_{out}(e, e_{a_j}))$$
$$= \frac{1}{(l_{out}+1)^{m_{out}}} \prod_{f_k \in T_{out}(f, f_j)} \sum_{e_{a_k} \in T_{out}(e, e_{a_j}) \cup \{0\}} t(f_k | e_{a_k})$$
$$P(T_{in}(f, f_j) | T_{in}(e, e_{a_j}))$$
$$= \frac{1}{(l_{in}+1)^{m_{in}}} \prod_{f_k \in T_{in}(f, f_j)} \sum_{e_{a_k} \in T_{in}(e, e_{a_j}) \cup \{0\}} t(f_k | e_{a_k})$$

The above two probabilities are derived using ideas from IBM Model 1 (Brown, 1993). Define the first heuristic function as:

$$h_1(f_j, e_{a_j}) = \log P(T(f),\ (f_j, e_{a_j}) | T(e))$$

5.2 Inside-Outside Penalty

Having defined the inside-outside probability, we now use its opposite, which is the probability of the inside treelet generated by the outside treelet times that of the outside treelet generated by the inside treelet:

$$Inside\_outside\_penalty$$
$$= P(T_{out}(f, f_j) | T_{in}(e, e_{a_j})) \times P(T_{in}(f, f_j) | T_{out}(e, e_{a_j}))$$

While we want to increase the inside outside probability, we want this penalty to be as small as possible. So we can define the second heuristic:

$$h_2(f_j, e_{a_j}) = -1 \times (\log P(T_{out}(f, f_j) | T_{in}(e, e_{a_j})) + \log P(T_{in}(f, f_j) | T_{out}(e, e_{a_j})))$$

5.3 Entropy

Since each tentative word pair (f_j, e_{a_j}) satisfies $\arg\max_{e_i \in e} t(f_j | e_i)$, where e is the set nodes in the English treelet, and $t(f_j | e_i)$ is the probability of e_i being translated to f_j, we have

$t(e_i \mid f_j) = \frac{t(f_j \mid e_i)P(e_i)}{P(f_j)}$. A reliable set of translation probabilities will provide a distribution with a high concentration of probabilities on the chosen word pairs. Intuitively, the conditional entropy of the translation probability distribution will serve as a good estimate of the confidence in the chosen word pair. Let $S = \sum_{e_i \in e} t(e_i \mid f_j)$ and define $\hat{e} \in e$ as a random variable.

$$H(\hat{e} \mid f_j) = \sum_{e_i \in e} -\frac{t(e_i \mid f_j)}{S} \log(\frac{t(e_i \mid f_j)}{S})$$

$$= \frac{\sum_{e_i \in e} -t(e_i \mid f_j)\log(t(e_i \mid f_j))}{S} + \log S$$

Since we need to compute conditional entropy given f_j, here the translation probabilities are normalized. The third heuristic function is defined as: $h_3(f_j, e_i) = -1 \times H(\hat{e} \mid f_j)$

5.4 Word Pair Probability

This heuristic is simply defined as the word pair translation probability: $h_4 = t(f_j \mid e_{a_j})$

5.5 Syntactic Category Templates

The dependency trees acquired from automatic parsers already provide us with the syntactic category of each node. Observing this, we collected syntactic category mappings from the automatically aligned results and did a cut off. By doing this we generated a set of likely syntactic category mappings, as we expect such mappings would be observed more often in the parallel corpora.

Chinese	*English*
VA	JJ / RB / NN / IN
VE	VBZ
VC	VBZ / VBD / VBP
VV	VB / NN / VBD / VBN / JJ / VBG / MD / VBP / VBZ / NNS / PRP
CD	CD / JJ / NN / NNS
NN	NN / NNS / JJ / NNP / VB / VBN / VBD / VBG
NR	NNP / NN / JJ
NT	NN
JJ	JJ / NN
PN	PRP / PRP$ / NN / WP VBP

Figure 6

Hence we can define the fifth heuristic as:

$$h_5 = \begin{cases} 1 & template\_match \\ 0 & no\_template\_match \end{cases}$$

6 Minimum Error Rate Training

We define the total score of the heuristic as a linear combination of all the heuristics introduced in Section 5.

$$h(f_j, e_{a_j}) = \sum_k \lambda_k h_k(f_j, e_{a_j})$$

In order to determine the λ_k for each heuristic function, we use minimum error rate training. The error function is defined as following:

Let (f_j, e_{a_j}) denote the word pair used for a tentative synchronous partitioning operation, which will create two treelet pairs:
$\left(T_{in}(f, f_j), T_{in}(e, e_{a_j})\right)$ and $\left(T_{out}(f, f_j), T_{out}(e, e_{a_j})\right)$.

Let P be the set of all the possible mappings of the two treelet pairs ($\times$ stands for cross-product):

$$P = Nodes\{T_{in}(f, f_j)\} \times Nodes\{T_{in}(e, e_{a_j})\} \cup$$
$$Nodes\{T_{out}(f, f_j)\} \times Nodes\{T_{out}(e, e_{a_j})\}$$

Let G be the set of word alignment pairs in the gold file, define:

$$G - P = \{(\hat{e}, \hat{f}) \mid (\hat{e}, \hat{f}) \in G, (\hat{e}, \hat{f}) \notin P\}$$

So we have: $Err(f_j, e_{a_j}) = \frac{|G - P|}{|G|}$

If we recall the definition of the "partition assumption" introduced at the end of Section 3, this error measure is exactly the percentage of gold file alignments that violates the partition assumption.

Let the best parameter set $\vec{\lambda}^*$ be:

$$\vec{\lambda}^* = \arg\min_{\hat{\lambda}} Err(f', e'),$$

$$\text{where } (f', e') = \arg\max_{(f_j, e_{a_j})} h(f_j, e_{a_j})$$

We used hill climbing with random restart to search for the best parameter vector in the parameter space. Although we would like to use more efficient methods to find the best parameter setting, we cannot do so due to the following two reasons:

1. The error function does not have an analytical form; hence we cannot compute the gradient.
2. The parameter vector $\vec{\lambda}$ interacts with the values of different heuristic functions, i.e. dif-

ferent $\vec{\lambda}$ would change the probabilities derived in the training process.

7 Results

We use an automatic syntactic parser (Bikel, 2002) to produce the parallel unaligned syntactic structures. The parser was trained using Penn English Treebank and Penn Chinese Treebank. We then used the algorithm in (Xia 2001) to convert the phrasal structure trees into dependency trees.

The following table shows the statistics of the datasets we used. (Genre, number of sentence pairs, number of Chinese words, number of English words, type: unaligned or manually word-level aligned, and usage).

Dataset	Xinhua	FBIS	Microsoft	IBM
Genre	News	News	Stories	News
Sent#	56263	21003	500	326
Chn W#	1456495	522953	5239	4718
Eng W#	1490498	658478	5476	7184
Type	unaligned	unaligned	aligned	aligned
Usage	training	training	testing	testing

Figure 7

The training sets consists Xinhua newswire data from LDC and FBIS data under the DARPA TIDES project. We filtered both datasets to ensure parallel sentence pair quality.

The testing sentences are provided by the courtesy of Microsoft Research, Asia and IBM Research. The sentence pairs are manually aligned at word level.

We used a part of the testing data as dev test to train the best $\vec{\lambda}*$. The results are as following:

h	1	2	3	4	5
λ	0.1459	0.0343	0.9129	0.4175	0.6546

Note that the trained parameter values in the above table do not reflect the relative weights of different heuristics since different heuristics are not normalized.

The following figure shows the number of parallel dependency treelets learnt from the training set and cumulative error rate on the testing set as the algorithm goes through 20 iterations. The error rate function is defined in Section 6. Note that as the treelets are getting smaller, cumulative error rate is inevitably increasing. Also we see the number of treelets collected is converging.

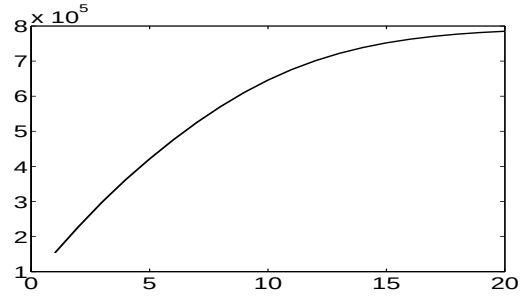


Figure 8. Treelet numbers

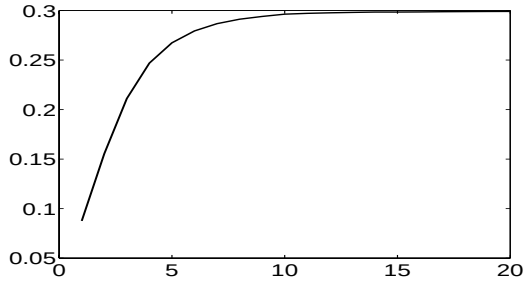


Figure 9. Cumulative error rate

We see that the cumulative error rate is converging to 30%. With a comparison to the findings in (Fox, 2002), which stated that the inherent head crossing in dependencies is around 12.62%, we find this result to be reasonable.

The following figure shows error count increase per synchronous tree partitioning operation in each iteration, namely: $\frac{\Delta(|G-P|)}{\Delta(number\_of\_treelets)}$

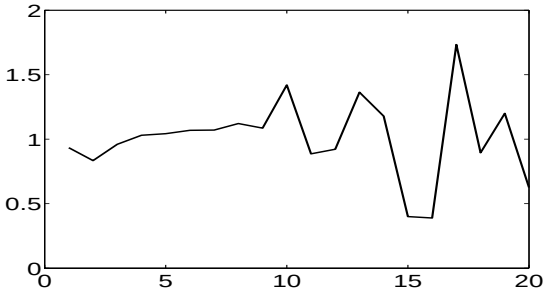


Figure 10.
Error mappings per synchronous partitioning operation

Considering each synchronous tree partitioning operation would cause significant reduction in treelet sizes, this per operation error rate is acceptable. We do notice however, the algorithm showed signs of overfitting after iteration 16.

After 20 iterations we collected 784924 treelet pairs from the training set. This means the treelets are converging to an average size of 5.26 words per treelet.

So how good are the dependency treelet pairs we learnt? To test the quality of the treelet pairs we used "content word alignment error rate". For each content word $Category(f_j) \in Content$, $f_j \in f$, we find its alignment in the corresponding treelet that maximize the translation probability. $e_{a_j} = \arg\max_{\hat{e}} t(f_j \mid \hat{e})$. We find that even with such a simple alignment model we can improve alignment results due to the use of more fine grained treelet pairs.

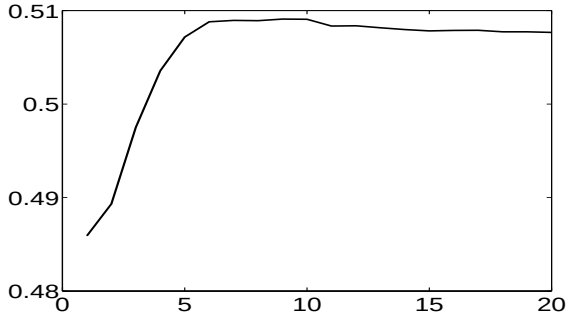


Figure 11. Content word alignment precision

The algorithm in our previous work (Ding, 2003) based on the similar framework showed an overall better performance than IBM Models 1 to 4 in word level alignment tasks.

Subjective tests show that quite often the treelets themselves are correct, but the simple max probability alignment makes mistakes inside the treelets. Overall, the treelets we collected have reasonable quality.

8 Conclusion and Future Work

In this paper, we show a statistical approach to learn parallel syntactic structures from parallel corpora. The new algorithm introduced in this paper automatically learns parallel dependency treelet pairs from loosely matched non-isomorphic dependency trees.

We evaluated the quality of the learnt dependency treelets and results showed that the more fine grained treelets helped content word alignments.

Future work includes inducing a formalized synchronous dependency grammar from the learnt treelets. Ideally, once the synchronous dependency grammar can be induced, it can be made a machine translation system.

References

Hiyan Alshawi, Srinivas Bangalore, and Shona Douglas. 2000. Learning dependency translation models as collections of finite state head transducers. *Computational Linguistics*, 26(1): 45-60.

Daniel M. Bikel. 2002. Design of a multi-lingual, parallel-processing statistical parsing engine. In *Proceedings of HLT 2002*.

Peter F. Brown, John Cocke, Stephen A. Della Pietra, Vincent J. Della Pietra, Frederick Jelinek, John D. Laferty, Robert L. Mercer, and Paul S. Roossin. 1990. A statistical approach to machine translation. *Computational Linguistics*, 16(2): 79-85, June.

Peter F. Brown, Stephen A. Della Pietra, Vincent J. Della Pietra, and Robert L. Mercer. 1993. The mathematics of statistical machine translation: parameter estimation. *Computational Linguistics*, 19(2): 263-311.

Michael John Collins. 1999. Head-driven Statistical Models for Natural Language Parsing. Ph.D. thesis, University of Pennsylvania, Philadelphia.

Yuan Ding, Daniel Gildea and Martha Palmer. An Algorithm for Word-level Alignment of Parallel Dependency Trees. *Proceedings of the 9th Machine Translation Summit of International Association of Machine Translation*, New Orleans, 2003

Bonnie J. Dorr. 1994. Machine translation divergences: A formal description and proposed solution. *Computational Linguistics*, 20(4): 597-633.

Heidi J. Fox. 2002. Phrasal cohesion and statistical machine translation. In *Proceedings of EMNLP-02*, pages 304-311

Daniel Gildea. 2003. Loosely tree based alignment for machine translation. In *Proceedings of the 41th Annual Conference of the Association for Computational Linguistics* (ACL-03)

Jan Hajic, et al. 2002. Natural language generation in the context of machine translation. Summer workshop final report, Center for Language and Speech Processing, Johns Hopkins University, Baltimore.

Franz Josef Och and Hermann Ney. 2000. Improved statistical alignment models. *In Proceedings of the 38th Annual Conference of the Association for Computational Linguistics* (ACL-00), pages 440-447, Hong Kong, October 2000.

Franz Josef Och. 2003. Minimum Error Rate Training in Statistical Machine Translation. *In Proceedings of the 41th Annual Conference of the Association for Computational Linguistics* (ACL-03), pages 160-167.

S. M. Shieber and Y. Schabes. 1990. *Synchronous Tree-Adjoining Grammars*, Proceedings of the 13th COLING, pp. 253-258, August 1990.

Dekai Wu. 1997. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computational Linguistics*, 23(3):3-403.

Fei Xia. 2001. Automatic grammar generation from two different perspectives. Ph.D. thesis, University of Pennsylvania, Philadelphia.

Kenji Yamada and Kevin Knight. 2001. A syntax based statistical translation model. In *Proceedings of the 39th Annual Conference of the Association for Computational Linguistics* (ACL-01), Toulouse, France.

Kenji Yamada and Kevin Knight. 2002. A decoder for syntax-based statistical MT. In *Proceedings of the 40th Annual Conference of the Association for Computational Linguistics* (ACL-02), Philadelphia.