A Discriminative Approach to Tree Alignment

Jörg Tiedemann
Alpha-Informatica, Rijksuniversiteit Groningen,
Groningen, The Netherland,
Department of Linguistics and Philology
Uppsala University, Uppsala/Sweden
j.tiedemann@rug.nl

Gideon Kotzé
Alpha-Informatica
Rijksuniversiteit Groningen
Groningen, The Netherlands
g.j.kotze@rug.nl

Abstract

In this paper we propose a discriminative framework for automatic tree alignment. We use a rich feature set and a log-linear model trained on small amounts of hand-aligned training data. We include contextual features and link dependencies to improve the results even further. We achieve an overall F-score of almost 80% which is significantly better than other scores reported for this task.

1 Introduction

A parallel treebank consists of a collection of sentence pairs that have been grammatically tagged, syntactically annotated and aligned on sub-sentential level [12]. Large parallel treebanks are much sought after in present-day NLP applications but have been, until recently, only been built by hand and therefore tended to be small and expensive to create. Some areas of application for parallel treebanks are:

- knowledge source for transfer-rule induction

- training for data-driven machine translation

- reference for phrase-alignment

- knowledge source for corpus-based translation studies

- knowledge source for studies in contrastive linguistics

As for ourselves, we are interested in applying tree alignment in the context of a syntax-based machine translation (MT) approach. Since well-aligned treebanks will play a substantial role in our MT model, finding an optimal solution to the problem of tree alignment is very important. In the next section, we provide a brief background of recent findings on the topic before presenting our own approach thereafter.

2 Related Work

Most related work on tree alignment is done in the context of machine translation research. Several variants of syntax-based MT approaches have been proposed in recent years involving the alignment of syntactic structures. In general we can distinguish between tree-to-string (or vice versa) and tree-to-tree alignment approaches. [15] describe some recent attempts at sub-sentential alignment on the phrase level:

[9] use a stochastic inversion transduction grammar to parse a source sentence and use the output to build up a target language parse, while also inducing alignments. The latter are extracted and converted into translation templates. [16] use a method they call "bilingual chunking", where the words of a tree pair are aligned and during the process, chunks are extracted by using the tree structure, after which the chunks are POS tagged. However, the original tree structures are lost in the process. [3] proposes a method which alters the structure of non-isomorphic phrase-structure trees to impose isomorphism in order to align the trees using a stochastic tree substitution grammar (STSG). This, however, restricts its portability to other domains, according to [15]. [4] present a rule-based aligner which makes use of previously determined word alignments. However, the algorithm performed poorly when applied to other language pairs [15].

According to [12] there are two general approaches to tree alignment: finding correspondences between phrases through parsing or chunking (eg. [13]), or deriving phrase alignment through previous word alignment, the latter of which they have adopted themselves, where the best configuration yields an $F_{0.5}$ score of 65.84%. Lately, in [15] a better and faster method was proposed using 1:1 word alignment probabilities and parse trees. Trees can also be constructed automatically in the absence of a parser. In a more recent update [17] taking all links into account, a highest precision of 61,79% and a highest recall of 78,49% in the tree alignment task were achieved. Zhechev and Way define a set of principles (2008:1106) to be followed in their alignment method:

- independence with respect to language pair and constituent labelling schema

- preservation of the given tree structures

- minimal external resources required

- word-level alignments are guided by links higher up the trees, which provide more context information

In addition, the authors quote [6] in defining a set of well-formedness criteria and explaining that this should result in producing "enough information to allow the inference of complex translational patterns from a parallel treebank, including some idiosyncratic translational divergences" (2008:1106): (i) A node in a tree may only be linked once. (ii) Descendants/ancestors of a source linked node may only be linked to descendants/ancestors of its target linked counterpart. In short the alignment algorithm consists of the following steps:

- Each source node s can link to any target node t and vice versa. Initially all these links are hypothesized.

- Every one of these hypotheses is assigned a score $\gamma(\langle s, t \rangle) = \alpha(s_l|t_l)\alpha(t_l|s_l)\alpha(\overline{s_l}|\overline{t_l})\alpha(\overline{t_l}|\overline{s_l})$ based on the word-alignment probabilities of the words that are governed by the current nodes (s_l and t_l), as well as the probabilities of the words outside the span ($\overline{s_l}$ and $\overline{t_l}$):

$$\alpha(x|y) = \prod_{i=1}^{|x|} \frac{1}{|y|} \sum_{j=1}^{|y|} P(x_i|y_j)$$

- Using these scores a set of links is selected applying a greedy search algorithm that also satisfies the well-formedness criteria.

Since the system described here has produced promising results and has been released publicly, we have decided to use it as a baseline, as well as a source of input material, upon which we hope to improve. For this we apply a discriminative alignment approach that is presented below.

3 Tree Alignment

In our approach we only look at tree-to-tree alignment using phrase-structure trees on both sides. In the following we first introduce the general link prediction model. Thereafter, we give a detailed description of features applied in our experiments and the alignment search strategy applied.

3.1 Link Prediction

Similar to related work on discriminative word alignment we base our model on association features extracted for each possible alignment candidate. For tree alignment, each pair of nodes $\langle s_i, t_j \rangle$ from the source and the target language parse tree is considered and a score x_{ij} is computed that represents the degree to which both nodes should be aligned according to their features $f_k(s_i, t_j, a_{ij})$ and corresponding weights λ_k derived from training data. In our approach we use conditional likelihood using a log-linear model for estimating these values:

$$P(a_{ij}|s_i, t_j) = \frac{1}{Z(s_i, t_j)} exp\left(\sum_k \lambda_k f_k(s_i, t_j, a_{ij})\right)$$

Here, the mapping of data points to features is user provided (see section 3.2) and the corresponding weights are learned from aligned training data. We simplify the problem by predicting individual alignment points for each candidate pair instead of aiming at structured approaches. Hence, we can train our conditional model as a standard binary classification problem. Note that contextual features can easily be integrated even though first-order dependencies on surrounding alignments are not explicitly part of the model. More details will be given below in sections 3.2.5 and 3.2.7.

In our experiments we will use a maximum entropy classifier using the log-linear model as stated above. One of the advantages of maximum entropy classifiers is the flexibility of choosing features. No independence assumptions have to be made and state-of-the art toolboxes are available with efficient learning strategies. Here, we apply the freely available toolbox Megam [1] and train a global binary classification model predicting links between given node pairs.

3.2 Alignment Features

The selection of appropriate features for classification is crucial in our approach. The input to the tree aligner is sentence aligned parse tree pairs from which various features can be extracted. Another important source is word alignment and information derived from statistical word alignment models. In the following we describe the different feature types that we apply.

3.2.1 Lexical Equivalence Features

Lexical probabilities are used in unsupervised tree alignment approaches as explained earlier in section 2. We will also use the same *inside/outside scores* defined in [17] as our basic features as they have proven to be useful for tree alignment. However, we define additional features and feature combinations derived from automatic word alignment in order to enrich the alignment model. First of all, we use inside and outside scores as individual features besides their product. We also use individual $\alpha(x|y)$ scores as separate features. Furthermore, we define a variant of inside and outside scores using a slightly modified definition of the equivalence score α:

$$\alpha_{max}(x|y) = \prod_{i=1}^{|x|} max_j P(x_i|y_j)$$

We believe that this definition better reflects the relations between words in sentences than the original definition in which an average of the conditional lexical probabilities is used. We assume that most words are linked to only one target word (hence we look for the maximum) whereas averaging over all combinations punishes long phrases too much.

Another variant can be defined by replacing the product above by a sum and taking the average per source token of this score:

$$\alpha_{avgmax}(x|y) = \frac{1}{|x|} \sum_{i=1}^{|x|} max_j P(x_i|y_j)$$

In this way, the impact of source tokens for which no links can be found with any of the target language tokens is reduced. In the original formulation scores will be zero if there is such a token even if all the other ones show a strong relation. This is avoided with the new definition. Using the modified scores the same combinations of inside and outside scores can be defined as explained earlier.

3.2.2 Word Alignment Features

Important features can be derived from the Viterbi alignments produced by statistical word alignment. Apart from the lexical probabilities used in the previous section, the actual statistical word alignment takes additional parameters into account, for example, positional similarity and first-order dependencies between links. Using Viterbi alignments we can implicitly take advantage of these additional parameters. We define *word alignment features* as the proportion of consistent links $cons(l_{xy}, s_i, t_j)$ among all links l_{xy} involving either source (s_x) or target language words (t_y) dominated by the current tree nodes (which we will call relevant links $relev(l_{w_s,w_t}, s_i, t_j)$). Consistent links are links between words which are both dominated by the nodes under consideration (dominance is denoted as $s_x \leq s_i$).

$$
\begin{aligned}
align(s_i, t_j) &= \frac{\sum_{l_{xy}} cons(l_{xy}, s_i, t_j)}{\sum_{l_{xy}} relev(l_{xy}, s_i, t_j)} \\
cons(l_{xy}, s_i, t_j) &= \begin{cases} 1 & \text{if } s_x \leq s_i \wedge t_y \leq t_j \\ 0 & \text{otherwise} \end{cases} \\
relev(l_{xy}, s_i, t_j) &= \begin{cases} 1 & \text{if } s_x \leq s_i \vee t_y \leq t_j \\ 0 & \text{otherwise} \end{cases}
\end{aligned}
$$

Note that the definition above is not restricted to word alignment. Other types of existing links between nodes dominated by the current subtree pair could be used in the same way. However, using the results of automatic word alignment we can compute these features from the links between terminal nodes. We can use various types of automatic word alignments. In our experiments we apply the Viterbi alignments produced by Giza++ [11] using the IBM 4 model in both directions, the union of these links and the intersection. For the latter we use Moses [8] which is also used for the estimation of lexical probabilities applied for lexical features described in the previous section.

Yet another feature derived from word alignment can be used to improve the alignment of terminal nodes. This feature is set to one if and only if both nodes are terminal nodes and are linked in the underlying word alignment.

3.2.3 Sub-tree Features

Features can also be derived directly from the parse trees. Similar to statistical word alignment, positional similarity can be used to make alignment decisions. However, in tree alignment we look at hierarchical structures and therefore a second dimension has to be considered. Therefore, we define the following two *tree position features*: tree-level similarity (tls) and

tree span similarity (tss). For the former we use the distances $d(s_i, s_{root})$, $d(t_i, t_{root})$ from the current candidate nodes to the root nodes of source and target language tree, respectively. Furthermore, we use the size of a tree (defined as the maximum distance of any terminal node in the tree to the root) to compute the relative tree level of a given node. Finally, the tree-level similarity is then defined as the one minus the absolute value of the difference between relative tree levels:

$$
tll(s_i, t_j) = 1 - abs \left(\frac{d(s_i, s_{root})}{max_x d(s_x, s_{root})} - \frac{d(t_i, t_{root})}{max_x d(t_x, t_{root})} \right)
$$

The second measure, the source span similarity is defined as one minus the absolute value of the difference between the relative positions of the subtrees under consideration. The relative positions are computed from the subtree spans using the surface positions $pos(s_x)$, $pos(t_y)$ of words dominated by the root nodes of these subtrees divided by the lengths of source and target language sentence, respectively.

$$
tss(s_i, t_j) = 1 - abs \left(\frac{min\ pos(s_x) + max\ pos(s_x)}{2 * length(S)} - \frac{min\ pos(t_y) + max\ pos(t_y)}{2 * length(T)} \right)
$$

Another tree feature that we will use refers to the number of terminal nodes dominated by the candidate nodes. We define the ratio of leaf nodes as follows:

$$
leafratio(s_i, t_j) = \frac{min(|s_x \leq s_i|, |t_y \leq t_i|)}{max(|s_x \leq s_i|, |t_y \leq t_i|)}
$$

The intuition behind this feature is the assumption that nodes dominating a large number of terminal nodes are less likely to be aligned to nodes dominating a small number of terminal nodes.

3.2.4 Annotation Features

Finally, we can also define binary features describing the presence of certain annotations. For example, we can define pairs of category labels (for non-terminal nodes) or part-of-speech labels (for terminal nodes) as binary features. Each observed combination of labels in the training data is then used as a possible feature and the weights learned in training will determine if they influence alignment decisions in a positive or negative way.

3.2.5 Contextual Features

Using the tree structure, we can extract similar features from the context of candidate nodes. In this way, first order dependencies can implicitly be included in the model. For example, including inside/outside scores from the parent nodes partially includes the likelihood of these nodes being aligned. This may then increase the likelihood of the current nodes to

be aligned as well. Contextual features can be very flexible and may also show a negative correlation. For example, a positive feature extracted for the current source language node together with the parent node of the target language node may decrease the likelihood of the alignment between the two current nodes.

In our implementation we allow various kinds of contextual features. Any feature as defined in the previous section can also be extracted from the parent nodes (either both parents or just one of them together with the current node in the other language). Furthermore, we also allow to extract these features from sister nodes (nodes with the same parent) and child nodes. For these nodes we only use the feature that provides the largest value.

We also allow multiple steps in our feature definition, allowing for, for example, grandparent features to be included. Naturally, contextual features are only extracted if the specified contexts actually exist (i.e. if there is a grandparent node).

3.2.6 Complex Features

A drawback of log-linear models is that features are combined in a linear way only. However, the correlation between some features might be non-linear and a typical strategy to reduce the negative effects of such interactions is to combine features and to build complex ones[1]. We define two operations for the combination of features:

Multiplication: The values of two or more features are multiplied with each other. This is only used for non-binary features.

Concatenation: Binary *feature types* can be combined with other features in the following way: Each *instantiation* of that type (for example a category label pair) is concatenated with the name of the other feature and the average of feature values is used.

We do not attempt to perform an exhaustive search among all possible combinations. Many of them will fail anyway due to data sparseness. However, complex features provide valuable contributions as we will see in our experiments.

3.2.7 Link Dependency Features

The last category of features refers to link dependency features. As we explained earlier, first-order dependencies are not explicitly modeled in our classification-based approach. However, features may include such dependencies, for example link information of connected nodes. Such features can easily be included in training where the complete link information is given. However, we have to adjust the link strategy in order to use these features in the alignment phase.

In our experiments, we define first-order features in the following way. The *children_links* feature is the number of links between child nodes of the current node pair normalized by the maximum of the number of source language children and the number of target language children. Similarly, the *subtree_links* feature is the number of links between nodes in the entire subtrees dominated by the current nodes. This score is then normalized by the larger number of nodes in either the source subtree or the target subtree.

In the alignment phase corresponding link information is not available. However, from the classifier we obtain probabilities for creating links between given nodes. We will use these conditional probabilities as soft counts for computing the first-order features as defined above, i.e. we sum over the link probabilities and normalize again in the same way. Our features are defined in terms of descendents of the current nodes. Hence, we perform classification in a bottom-up breadth-first fashion starting at the terminal nodes that do not include any children.

We also tried a top-down classification strategy together with parent link dependencies. However, this did not give us any significant improvements. Therefore, we will not report these results here.

3.3 Alignment Search

Our tree alignment approach is based on a global binary classifier. This means that we actually classify individual node pairs even though we include contextual and first-order features as described above. Despite the fact that individual classification is possible in this way, the important notion of *alignment competition* is not explored in this way. That this is a strong drawback has already been pointed out in related research on word alignment [14]. However, similar to discriminative word alignment, competition can easily be integrated in the system by applying appropriate search strategies. Naturally, the best strategy would be to include competition explicitly in the alignment model and train parameters for a structural alignment approach. We will leave this for future research and concentrate our current work on feature selection in combination with simple greedy search heuristics. In particular, we will use a greedy best-first search similar to competitive linking used in early work on word alignment. One of the drawbacks in this technique is that it only allows one-to-one links. However, in tree alignment this is not necessarily a drawback and often even defined as a well-formedness criterion [15]. Another drawback is, of course, that we are not guaranteed to find the optimal solution. However, it should be rather straightforward to implement a graph-theoretic search approach as described by [14] defining tree alignment as a weighted bipartite graph matching problem. We will leave even this for future research.

Finally, we will also introduce additional constraints that may help to improve alignment accuracy. First of all, a threshold can be defined in order to stop the greedy link strategy if link probabilities obtained by the classifier are too low. Secondly, a number of well-formedness criteria can be added to avoid unusual link combinations. We will use the criteria as defined in [17], as already mentioned in section 2: De-

---

[1] Another possibility would be to switch to kernel-based methods and to apply, for example, support vector machines with non-linear kernels. This will be tested more thoroughly in future work. Our first experiments with SVMs were discouraging mainly due to the largely increased time necessary for training.

scendents/ancestors of a source linked node may only be linked to descendents/ancestors of its target linked counterparts. Furthermore, we will use another constraint which is similar to the collapsing strategy of unary productions used by the same authors. However, we do not collapse trees at these points but we simply do not align nodes with single children. Note that this still allows links between terminal nodes as they do not have any children at all. Node type specific constraints can also be applied. For example, we may restrict links to be assigned to nodes of the same type only (non-terminals to non-terminals and terminals to terminals). We may also restrict ourselves to non-terminal nodes only. Note that these restrictions change the behavior of the unary-production constraint in the following way: If these restrictions are applied the unary-production constraint is relaxed in such a way that these nodes are only skipped if the one and only child is not a terminal node. This relaxation is necessary to include valuable links near the leafs that otherwise would be skipped.

Our implementation allows to switch on and off any of the constraints described above. Search heuristics can also easily be altered within the framework described above. In the following section we will describe experiments using various settings and models trained on a given treebank.

4 Experiments

We ran a number of experiments using a pre-aligned treebank and various settings including features as described above. In the following, we will first briefly describe the data used for training and testing. Thereafter evaluation measures are defined and results of our experiments are summarized.

4.1 Data

Aligned parallel treebanks are rare and, hence, training material for a supervised tree alignment approach is hard to find. However, a number of parallel treebank projects have been initiated recently and their data and tools become available. For our experiments, we will use the Smultron treebank [5] that includes two trilingual parallel treebanks in English, Swedish and German. The corpus contains the alignment of English-Swedish and German-Swedish phrase structure trees from the first two chapters of the novel "Sophie's World" by Jostein Gaarder and from economical texts taken from three different sources. We will use the English-Swedish treebank of Sophie's World which includes roughly 500 sentences per language. The first 100 aligned parse trees are used for training and the remaining part for testing. The alignment has been done manually using the Stockholm Tree Aligner [10] which we also intend to use later on when working on our own corpora and language pairs. The alignment includes *good* links and *fuzzy* links. We will use both but give them different weights in training (good alignments get three times the weight of fuzzy and negative examples). Altogether, there are 6,671 good links and 1,141 fuzzy links in the corpus.

4.2 Evaluation

For evaluation we use the standard measures of precision, recall and F-scores. Due to the distinction between good and fuzzy alignments we compute values similar to word alignment evaluation scores in which "sure" and "possible" links are considered:

$$
\begin{aligned}
Prec(A, P) &= |P \cap A|/|A| \\
Rec(A, S) &= |S \cap A|/|S| \\
F(A, P, S, \alpha) &= 1/\left(\frac{\alpha}{Prec(A, P)} + \frac{(1 - \alpha)}{Rec(A, S)}\right)
\end{aligned}
$$

S refers here to the good alignments in the gold standard and P refers to the possible alignments which includes both, good and fuzzy. A are the links proposed by the system and α is used to define the balance between precision and recall in the F-score. We will only use a balanced F-score with $\alpha = 0.5$. We also omit alignment error rates due to the discussion about this measure in the word alignment literature. Note that the proportion of fuzzy links seems reasonable and we do not expect severe consequences on our evaluation as discussed in [2] for word alignment experiments with unbalanced gold standards.

4.3 Results

The selection of appropriate features is very important in our approach. We tested a number of feature sets and combinations in order to see the impact of features on alignment results. Table 1 summarizes our experiments with various sets. The upper part represents the performance of separate feature types on their own. The lower part shows results of combined feature types. Link dependency features are added in the right-hand side columns – either child link dependencies or dependencies on all subtree nodes.

As we can see in table 1, adding features consistently improves the scores even if their standalone performance is rather low. Especially the addition of label features improves the scores significantly. Contextual features are also very useful as we can see on the example of label features. Note, that we also use complex features such as combined inside/outside scores and alignment features. Also the concatenation of label features with alignment features is very successful.

For comparison we also ran the subtree aligner by [17] on the same data set. It yields a balanced F-score on our test set of 57.57% which is significantly lower than our best results. However, this comparison is not entirely fair as our training data is very small and the unsupervised subtree aligner relies on good estimates of lexical probabilities. Therefore, we also ran the aligner on our data with a lexical model extracted from a much larger data set. For this, we used the combination of the entire Swedish-English Europarl corpus [7] and the Smultron data. However, the scores improve only slightly to an F-score of 58.64%. The reason for this is probably that the Europarl data represents a very different type than the novel used in our test. However, it indicates the possibilities of discriminative tree alignment when trained on small amounts of aligned data.

features	no link dependencies			+ child link dependencies			+ subtree link dependencies		
	Prec	Rec	$F_{0.5}$	Prec	Rec	$F_{0.5}$	Prec	Rec	$F_{0.5}$
lexical	65.54	35.72	46.24	62.92	41.26	49.84	59.64	41.07	48.64
lexical$_{max}$	66.07	36.77	47.24	63.17	41.74	50.26	59.76	41.81	49.20
lexical$_{avgmax}$	63.76	43.04	51.39	60.95	41.96	49.70	60.92	41.94	49.68
tree	30.46	34.50	32.36	33.10	38.61	35.64	33.37	38.81	35.84
alignment	61.36	54.52	57.74	64.91	58.72	61.66	59.24	54.68	56.87
label	36.14	35.12	35.62	45.00	41.38	43.11	48.77	44.03	46.28
context-label	56.53	44.64	49.88	59.17	50.79	53.72	60.47	53.44	56.74
lexical$_{max}$ + tree	48.32	55.15	51.51	54.86	57.51	52.95	49.40	57.25	53.03
+ alignment	55.65	57.94	56.77	57.09	60.31	58.65	57.18	60.58	58.83
+ label	73.43	74.76	74.09	74.39	75.86	75.12	74.68	76.67	75.17
+ context-label	**76.65**	75.45	76.05	76.99	75.85	76.42	77.17	76.10	76.63
+ align-context	76.23	**77.43**	**76.83**	**77.07**	**78.16**	**77.61**	**78.12**	**78.42**	**78.27**

Table 1: *Results for different feature sets.*

Furthermore, we want to see the performance of our tree aligner on different node types. For this we computed separate evaluation scores for the different types using a run with all features (see table 2).

		Rec		Prec	$F_{0.5}$
type	*good*	*fuzzy*	*all*	*all*	*all*
non-terminals	**84.29**	**70.23**	**81.28**	78.04	**79.63**
terminals	75.08	59.11	73.80	**78.18**	75.93

Table 2: *Results for different node types (all features) including recall scores for different link types.*

From the table we can see that the aligner has more difficulties in finding links between terminal nodes than between non-terminals. This is especially true for fuzzy links. However, the precision is as high as for non-terminal nodes[2]. The reason for the drop in recall is probably due to the search algorithm which restricts our results to one-to-one links only. This constraint might be reasonable for non-terminal nodes but not for the alignment of words. A conclusion from this result is that we should either keep the external word alignment for establishing terminal links in our tree alignment or that we should use a separate model and search strategy for aligning terminal nodes.

Finally, we also want to look at the generality of our approach. A drawback of supervised methods is the risk of over-training especially if a rich feature set and small amounts of training data are used. Certainly, our approach is not language independent especially when label features are applied. However, we would like to know if the models learned can be applied to different text types without significant loss in performance. Therefore, we carried out an experiment training on one text type (novel or economy) and aligning the other one from the Smultron corpus. For reasons of fair comparison we also trained on the first 100 sentence pairs only but applied the model learned to the entire test corpus of the other type. Table 3 summarizes the results when applying the full-featured model in this way.

As we can see in the table, performance drops, especially in terms of recall. Precision is still comparable to

setting	Prec	Rec	$F_{0.5}$
train=novel, test=novel	78.12	78.42	78.27
train=novel, test=economy	77.39	73.50	75.39
train=economy, test=novel	76.66	74.62	75.62

Table 3: *Training on different text types*

the model trained on the same corpus (see line one in table 3). However, the drop is not dramatical and the models seem to capture enough general associations to make reasonable predictions. This is certainly encouraging especially considering the effort of human annotation necessary when preparing appropriate training data.

5 Conclusions & Future Work

In this paper we describe a discriminative framework for automatic tree alignment. A log-linear model is learned from small amounts of pre-aligned training data. We use a rich set of features coming from the annotation and from automatic word alignment. We include contextual features and link dependency information for further improvements. Our model performs significantly better than previous methods on the same task and we believe that our results can be further improved in various ways. Some ideas for future work include the optimization of the search algorithm (using a graph-theoretic matching approach), the exploration of automatic methods for feature selection and combination (using, for example, a genetic algorithm) and a better integration of link dependencies (using a structural model instead of a single binary classifier). We will also look at additional features and the application of this approach to other data sets and language pairs. Finally, we will also investigate the impact of alignment quality on machine translation models based on parallel treebanks.

References

[1] H. Daumé III. Notes on CG and LM-BFGS optimization of logistic regression. Paper available at http://pub.hal3.name#daume04cg-bfgs, implementation available at http://hal3.name/megam/, August 2004.

---

[2] Note that the aligner does not assign link types and therefore, precision cannot be measured for different types.

[2] A. Fraser and D. Marcu. Measuring word alignment quality for statistical machine translation. *Computational Linguistics*, 33(3):293–303, 2007.

[3] D. Gildea. Loosely tree-based alignment for machine translation. In *Proceedings of the 41st Annual Meeting of the Association of Computational Linguistics (ACL-03)*, pages 80–87, Sapporo, Japan, 2003.

[4] D. Groves, M. Hearne, and A. Way. Robust sub-sentential alignment of phrase-structure trees. In *Proceedings of the 20th International Conference on Computational Linguistics (CoLing 2004)*, pages 1072–1078, Geneva, Switzerland, 2004.

[5] S. Gustafson-Čapková, Y. Samuelsson, and M. Volk. SMULTRON (version 1.0) - The Stockholm MULtilingual parallel TReebank. http://www.ling.su.se/dali/research/smultron/index.htm, 2007. An English-German-Swedish parallel Treebank with sub-sentential alignments.

[6] M. Hearne, J. Tinsley, V. Zhechev, and A. Way. Capturing translational divergences with a statistical tree-to-tree aligner. In *Proceedings of the 11th International Conference on Theoretical and Methodological Issues in Machine Translation (TMI '07)*, pages 85–94, Skövde, Sweden, 2007.

[7] P. Koehn. Europarl: A parallel corpus for statistical machine translation. In *Proceedings of MT Summit*, 2005.

[8] P. Koehn, H. Hoang, A. Birch, C. Callison-Burch, M. Federico, N. Bertoldi, B. Cowan, W. Shen, C. Moran, R. Zens, C. Dyer, O. Bojar, A. Constantin, and E. Herbst. Moses: Open source toolkit for statistical machine translation. In *Proceedings of ACL*, Prague, Czech Republic, 2007.

[9] Y. Lü, M. Zhou, and S. Li. Automatic translation template acquisition based on bilingual structure alignment. *Computational Linguistics and Chinese Language Processing*, 6(1):83–108, 2001.

[10] J. Lundborg, T. Marek, M. Mettler, and M. Volk. Using the stockholm treealigner. In *Proceedings of the 6th Workshop on Treebanks and Linguistic Theories*, pages 73–78, Bergen, Norway, 2007.

[11] F. J. Och and H. Ney. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51, 2003.

[12] Y. Samuelsson and M. Volk. Automatic phrase alignment: Using statistical n-gram alignment for syntactic phrase alignment. In *Proceedings of the Sixth International Workshop on Treebanks and Linguistic Theories*, pages 139–150, Geneva, Switzerland, 2007.

[13] B. Schrader. *Exploiting Linguistic and Statistical Knowledge in a Text Alignment System*. PhD thesis, Universität Osnabrück, 2007.

[14] B. Taskar, S. Lacoste-Julien, and D. Klein. A discriminative matching approach to word alignment. In *Proceedings of HLT/EMNLP*, Vancouver, Canada, 2005.

[15] J. Tinsley, V. Zhechev, M. Hearne, and A. Way. Robust language pair-independent sub-tree alignment. In *Proceedings of Machine Translation Summit XI*, pages 467–474, Copenhagen, Denmark, 2007.

[16] W. Wang, J.-X. Huang, M. Zhou, and C.-N. Huang. Structure alignment using bilingual chunking. In *Proceedings of the 19th Conference on Computational Linguistics*, pages 1–7, Taipei, Taiwan, 2002.

[17] V. Zhechev and A. Way. Automatic generation of parallel treebanks. In *Proceedings of the 22nd International Conference on Computational Linguistics (CoLing)*, pages 1105–1112, 2008.