Loosely Tree-Based Alignment for Machine Translation

Daniel Gildea
University of Pennsylvania
dgildea@cis.upenn.edu

Abstract

We augment a model of translation based on re-ordering nodes in syntactic trees in order to allow alignments not conforming to the original tree structure, while keeping computational complexity polynomial in the sentence length. This is done by adding a new subtree cloning operation to either tree-to-string or tree-to-tree alignment algorithms.

1 Introduction

Systems for automatic translation between languages have been divided into transfer-based approaches, which rely on interpreting the source string into an abstract semantic representation from which text is generated in the target language, and statistical approaches, pioneered by Brown et al. (1990), which estimate parameters for a model of word-to-word correspondences and word re-orderings directly from large corpora of parallel bilingual text. Only recently have hybrid approaches begun to emerge, which apply probabilistic models to a structured representation of the source text. Wu (1997) showed that restricting word-level alignments between sentence pairs to observe syntactic bracketing constraints significantly reduces the complexity of the alignment problem and allows a polynomial-time solution. Alshawi et al. (2000) also induce parallel tree structures from unbracketed parallel text, modeling the generation of each node's children with a finite-state transducer. Yamada and Knight (2001) present an algorithm for estimating probabilistic parameters for a similar model which represents translation as a sequence of re-ordering operations over children of nodes in a syntactic tree, using automatic parser output for the initial tree structures. The use of explicit syntactic information for the target language in this model has led to excellent translation results (Yamada and Knight, 2002), and raises the prospect of training a statistical system using syntactic information for both sides of the parallel corpus.

Tree-to-tree alignment techniques such as probabilistic tree substitution grammars (Hajič et al., 2002) can be trained on parse trees from parallel treebanks. However, real bitexts generally do not exhibit parse-tree isomorphism, whether because of systematic differences between how languages express a concept syntactically (Dorr, 1994), or simply because of relatively free translations in the training material.

In this paper, we introduce "loosely" tree-based alignment techniques to address this problem. We present analogous extensions for both tree-to-string and tree-to-tree models that allow alignments not obeying the constraints of the original syntactic tree (or tree pair), although such alignments are dispreferred because they incur a cost in probability. This is achieved by introducing a clone operation, which copies an entire subtree of the source language syntactic structure, moving it anywhere in the target language sentence. Careful parameterization of the probability model allows it to be estimated at no additional cost in computational complexity. We expect our relatively unconstrained clone operation to allow for various types of structural divergence by providing a sort of hybrid between tree-based and unstructured, IBM-style models.

We first present the tree-to-string model, followed by the tree-to-tree model, before moving on to alignment results for a parallel syntactically annotated Korean-English corpus, measured in terms of alignment perplexities on held-out test data, and agreement with human-annotated word-level alignments.

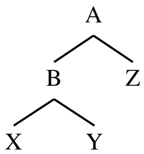
2 The Tree-to-String Model

We begin by summarizing the model of Yamada and Knight (2001), which can be thought of as representing translation as an Alexander Calder mobile. If we follow the process of an English sentence's transformation into French, the English sentence is first given a syntactic tree representation by a statistical parser (Collins, 1999). As the first step in the translation process, the children of each node in the tree can be re-ordered. For any node with m children, $m!$ re-orderings are possible, each of which is assigned a probability P_{order} conditioned on the syntactic categories of the parent node and its children. As the second step, French words can be inserted at each node of the parse tree. Insertions are modeled in two steps, the first predicting whether an insertion to the left, an insertion to the right, or no insertion takes place with probability P_{ins}, conditioned on the syntactic category of the node and that of its parent. The second step is the choice of the inserted word $P_t(f|\text{NULL})$, which is predicted without any conditioning information. The final step, a French translation of each original English word, at the leaves of the tree, is chosen according to a distribution $P_t(f|e)$. The French word is predicted conditioned only on the English word, and each English word can generate at most one French word, or can generate a NULL symbol, representing deletion. Given the original tree, the re-ordering, insertion, and translation probabilities at each node are independent of the choices at any other node. These independence relations are analogous to those of a stochastic context-free grammar, and allow for efficient parameter estimation by an inside-outside Expectation Maximization (EM) algorithm. The computation of inside probabilities β, outlined below, considers possible reordering of nodes in the original tree in a bottom-up manner:

for all nodes ε_i in input tree T **do**
  for all k, l such that $1 < k < l < N$ **do**
    for all orderings ρ of the children $\varepsilon_1 ... \varepsilon_m$ of ε_i **do**
      for all partitions of span k, l into $k_1, l_1 ... k_m, l_m$ **do**
        $\beta(\varepsilon_i, k, l)$+= $P_{order}(\rho|\varepsilon_i) \prod_{j=1}^{m} \beta(\varepsilon_j, k_j, l_j)$
      end for
    end for
  end for
end for

This algorithm has computational complexity $O(|T|N^{m+2})$, where m is the maximum number of children of any node in the input tree T, and N the length of the input string. By storing partially completed arcs in the chart and interleaving the inner two loops, complexity of $O(|T|n^3 m! 2^m)$ can be achieved. Thus, while the algorithm is exponential in m, the fan-out of the grammar, it is polynomial in the size of the input string. Assuming $|T| = O(n)$, the algorithm is $O(n^4)$.

The model's efficiency, however, comes at a cost. Not only are many independence assumptions made, but many alignments between source and target sentences simply cannot be represented. As a minimal example, take the tree:



Of the six possible re-orderings of the three terminals, the two which would involve crossing the bracketing of the original tree (XZY and YZX) are not allowed. While this constraint gives us a way of using syntactic information in translation, it may in many cases be too rigid. In part to deal with this problem, Yamada and Knight (2001) flatten the trees in a pre-processing step by collapsing nodes with the same lexical head-word. This allows, for example, an English subject-verb-object (SVO) structure, which is analyzed as having a VP node spanning the verb and object, to be re-ordered as VSO in a language such as Arabic. Larger syntactic divergences between the two trees may require further relaxation of this constraint, and in practice we expect such divergences to be frequent. For example, a nominal modifier in one language may show up as an adverbial in the other, or, due to choices such as which information is represented by a main verb, the syntactic correspondence between the two

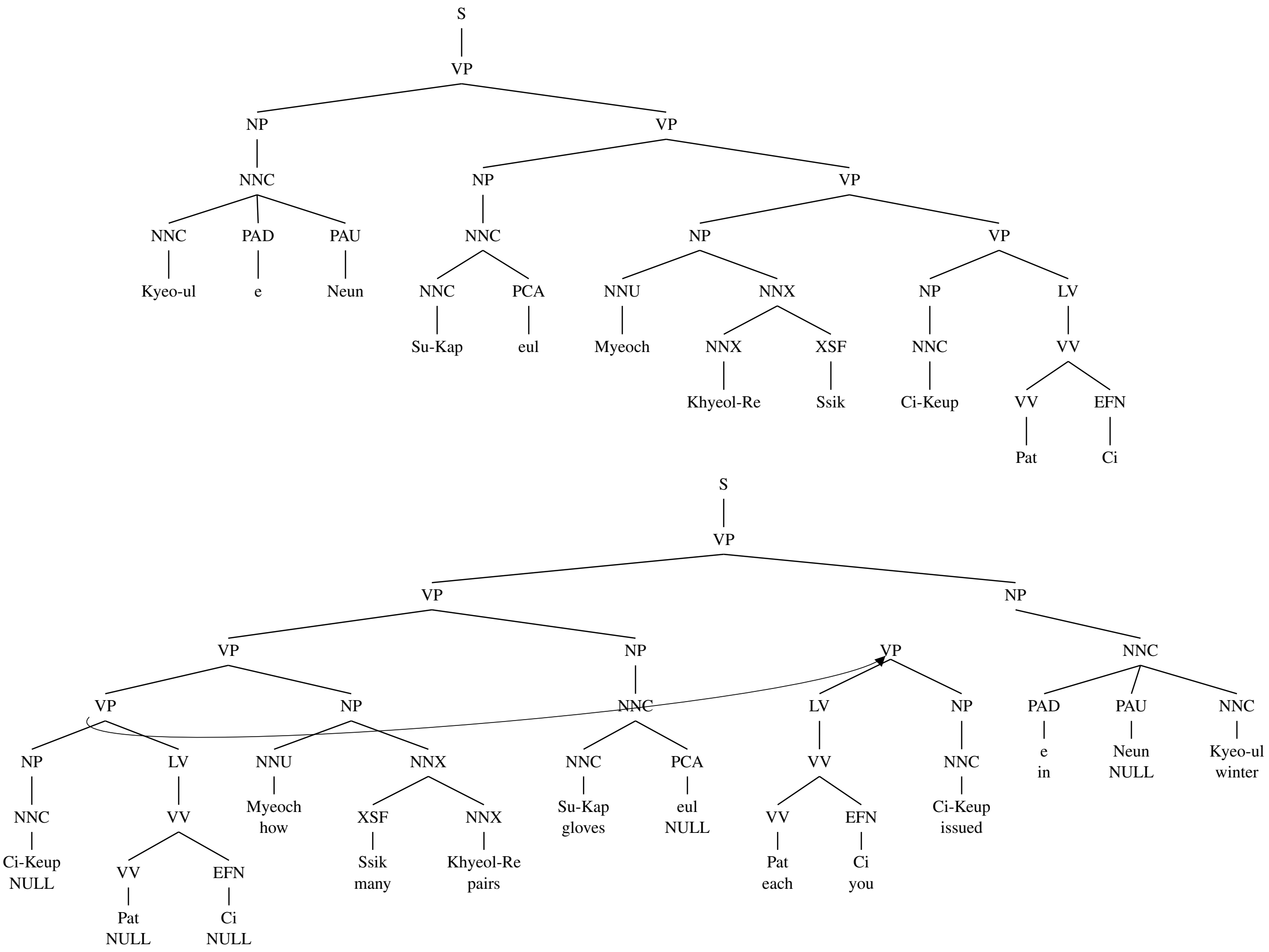
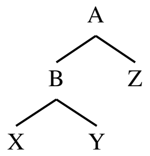


Figure 1: Original Korean parse tree, above, and transformed tree after reordering of children, subtree cloning (indicated by the arrow), and word translation. After the insertion operation (not shown), the tree's English yield is: *How many pairs of gloves is each of you issued in winter?*

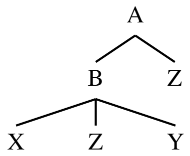
sentences may break down completely.

2.1 Tree-to-String Clone Operation

In order to provide some flexibility, we modify the model in order to allow for a copy of a (translated) subtree from the English sentences to occur, with some cost, at any point in the resulting French sentence. For example, in the case of the input tree



a clone operation making a copy of node 3 as a new child of B would produce the tree:



This operation, combined with the deletion of the original node Z, produces the alignment (XZY) that was disallowed by the original tree reordering model. Figure 1 shows an example from our Korean-English corpus where the clone operation allows the model to handle a case of *wh*-movement in the English sentence that could not be realized by any reordering of subtrees of the Korean parse.

The probability of adding a clone of original node ε_i as a child of node ε_j is calculated in two steps: first, the choice of whether to insert a clone under ε_j, with probability $P_{ins}(\text{clone}|\varepsilon_j)$, and the choice of which original node to copy, with probability

$$P_{clone}(\varepsilon_i|\text{clone} = 1) = \frac{P_{makeclone}(\varepsilon_i)}{\sum_k P_{makeclone}(\varepsilon_k)}$$

where $P_{makeclone}$ is the probability of an original node producing a copy. In our implementation, for simplicity, $P_{ins}(\text{clone})$ is a single number, estimated by the EM algorithm but not conditioned on the parent node ε_j, and $P_{makeclone}$ is a constant, meaning that the node to be copied is chosen from all the nodes in the original tree with uniform probability.

It is important to note that $P_{makeclone}$ is not dependent on whether a clone of the node in question has already been made, and thus a node may be "reused" any number of times. This independence assumption is crucial to the computational tractability of the algorithm, as the model can be estimated using the dynamic programming method above, keeping counts for the expected number of times each node has been cloned, at no increase in computational complexity. Without such an assumption, the parameter estimation becomes a problem of parsing with crossing dependencies, which is exponential in the length of the input string (Barton, 1985).

3 The Tree-to-Tree Model

The tree-to-tree alignment model has tree transformation operations similar to those of the tree-to-string model described above. However, the transformed tree must not only match the surface string of the target language, but also the tree structure assigned to the string by the treebank annotators. In order to provide enough flexibility to make this possible, additional tree transformation operations allow a single node in the source tree to produce two nodes in the target tree, or two nodes in the source tree to be grouped together and produce a single node in the target tree. The model can be thought of as a synchronous tree substitution grammar, with probabilities parameterized to generate the target tree conditioned on the structure of the source tree.

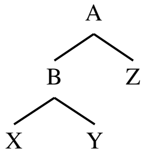
The probability $P(T_b|T_a)$ of transforming the source tree T_a into target tree T_b is modeled in a sequence of steps proceeding from the root of the target tree down. At each level of the tree:

1. At most one of the current node's children is grouped with the current node in a single *elementary tree*, with probability $P_{elem}(t_a|\varepsilon_a \Rightarrow children(\varepsilon_a))$, conditioned on the current node ε_a and its children (ie the CFG production expanding ε_a).

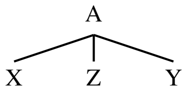
2. An alignment of the children of the current elementary tree is chosen, with probability $P_{align}(\alpha|\varepsilon_a \Rightarrow children(t_a))$. This alignment operation is similar to the re-order operation in the tree-to-string model, with the extension that 1) the alignment α can include insertions and deletions of individual children, as nodes in either the source or target may not correspond to anything on the other side, and 2) in the case where two nodes have been grouped into t_a, their children are re-ordered together in one step.

In the final step of the process, as in the tree-to-string model, lexical items at the leaves of the tree are translated into the target language according to a distribution $P_t(f|e)$.

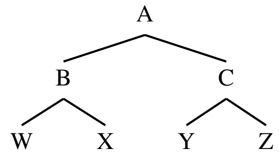
Allowing non-1-to-1 correspondences between nodes in the two trees is necessary to handle the fact that the depth of corresponding words in the two trees often differs. A further consequence of allowing elementary trees of size one or two is that some reorderings not allowed when reordering the children of each individual node separately are now possible. For example, with our simple tree



if nodes A and B are considered as one elementary tree, with probability $P_{elem}(t_a|\mathrm{A} \Rightarrow \mathrm{BZ})$, their collective children will be reordered with probability $P_{align}(\{(1,1)(2,3)(3,2)\}|\mathrm{A} \Rightarrow \mathrm{XYZ})$



giving the desired word ordering XZY. However, computational complexity as well as data sparsity prevent us from considering arbitrarily large elementary trees, and the number of nodes considered at once still limits the possible alignments. For example, with our maximum of two nodes, no transformation of the tree



is capable of generating the alignment WYXZ.

In order to generate the complete target tree, one more step is necessary to choose the structure on the target side, specifically whether the elementary tree has one or two nodes, what labels the nodes have, and, if there are two nodes, whether each child attaches to the first or the second. Because we are ultimately interested in predicting the correct target string, regardless of its structure, we do not assign probabilities to these steps. The nonterminals on the target side are ignored entirely, and while the alignment algorithm considers possible pairs of nodes as elementary trees on the target side during training, the generative probability model should be thought of as only generating single nodes on the target side. Thus, the alignment algorithm is constrained by the bracketing on the target side, but does not generate the entire target tree structure.

While the probability model for tree transformation operates from the top of the tree down, probability estimation for aligning two trees takes place by iterating through pairs of nodes from each tree in bottom-up order, as sketched below:

for all nodes ε_a in source tree T_a in bottom-up order **do**
  for all elementary trees t_a rooted in ε_a **do**
    for all nodes ε_b in target tree T_b in bottom-up order **do**
      for all elementary trees t_b rooted in ε_b **do**
        for all alignments α of the children of t_a and t_b **do**
          $\beta(\varepsilon_a, \varepsilon_b)$ += $P_{elem}(t_a|\varepsilon_a)P_{align}(\alpha|\varepsilon_i)\prod_{(i,j)\in\alpha}\beta(\varepsilon_i, \varepsilon_j)$
        end for
      end for
    end for
  end for
end for

The outer two loops, iterating over nodes in each tree, require $O(|T|^2)$. Because we restrict our elementary trees to include at most one child of the root node on either side, choosing elementary trees for a node pair is $O(m^2)$, where m refers to the maximum number of children of a node. Computing the alignment between the $2m$ children of the elementary tree on either side requires choosing which subset of source nodes to delete, $O(2^{2m})$, which subset of target nodes to insert (or clone), $O(2^{2m})$, and how to reorder the remaining nodes from source to target tree, $O((2m)!)$. Thus overall complexity of the algorithm is $O(|T|^2 m^2 4^{2m}(2m)!)$, quadratic in the size of the input sentences, but exponential in the fan-out of the grammar.

3.1 Tree-to-Tree Clone Operation

Allowing m-to-n matching of up to two nodes on either side of the parallel treebank allows for limited non-isomorphism between the trees, as in Hajič et al. (2002). However, even given this flexibility, requiring alignments to match two input trees rather than one often makes tree-to-tree alignment more constrained than tree-to-string alignment. For example, even alignments with no change in word order may not be possible if the structures of the two trees are radically mismatched. This leads us to think it may be helpful to allow departures from

	Tree-to-String	Tree-to-Tree
elementary tree grouping		$P_{elem}(t_a \vert \varepsilon_a \Rightarrow children(\varepsilon_a))$
re-order	$P_{order}(\rho \vert \varepsilon \Rightarrow children(\varepsilon))$	$P_{align}(\alpha \vert \varepsilon_a \Rightarrow children(t_a))$
insertion	$P_{ins}(\text{left, right, none} \vert \varepsilon)$	α can include "insertion" symbol
lexical translation	$P_t(f \vert e)$	$P_t(f \vert e)$
with cloning	$P_{ins}(\text{clone} \vert \varepsilon)$ $P_{makeclone}(\varepsilon)$	α can include "clone" symbol $P_{makeclone}(\varepsilon)$

Table 1: Model parameterization

the constraints of the parallel bracketing, if it can be done in without dramatically increasing computational complexity.

For this reason, we introduce a clone operation, which allows a copy of a node from the source tree to be made anywhere in the target tree. After the clone operation takes place, the transformation of source into target tree takes place using the tree decomposition and subtree alignment operations as before. The basic algorithm of the previous section remains unchanged, with the exception that the alignments α between children of two elementary trees can now include cloned, as well as inserted, nodes on the target side. Given that α specifies a new cloned node as a child of ε_j, the choice of which node to clone is made as in the tree-to-string model:

$$P_{clone}(\varepsilon_i|\text{clone} \in \alpha) = \frac{P_{makeclone}(\varepsilon_i)}{\sum_k P_{makeclone}(\varepsilon_k)}$$

Because a node from the source tree is cloned with equal probability regardless of whether it has already been "used" or not, the probability of a clone operation can be computed under the same dynamic programming assumptions as the basic tree-to-tree model. As with the tree-to-string cloning operation, this independence assumption is essential to keep the complexity polynomial in the size of the input sentences.

For reference, the parameterization of all four models is summarized in Table 1.

4 Data

For our experiments, we used a parallel Korean-English corpus from the military domain (Han et al., 2001). Syntactic trees have been annotated by hand for both the Korean and English sentences; in this paper we will be using only the Korean trees, modeling their transformation into the English text. The corpus contains 5083 sentences, of which we used 4982 as training data, holding out 101 sentences for evaluation. The average Korean sentence length was 13 words. Korean is an agglutinative language, and words often contain sequences of meaning-bearing suffixes. For the purposes of our model, we represented the syntax trees using a fairly aggressive tokenization, breaking multimorphemic words into separate leaves of the tree. This gave an average of 21 tokens for the Korean sentences. The average English sentence length was 16. The maximum number of children of a node in the Korean trees was 23 (this corresponds to a comma-separated list of items). 77% of the Korean trees had no more than four children at any node, 92% had no more than five children, and 96% no more than six children. The vocabulary size (number of unique types) was 4700 words in English, and 3279 in Korean — before splitting multi-morphemic words, the Korean vocabulary size was 10059. For reasons of computation speed, trees with more than 5 children were excluded from the experiments described below.

5 Experiments

We evaluate our translation models both in terms agreement with human-annotated word-level alignments between the sentence pairs. For scoring the viterbi alignments of each system against gold-standard annotated alignments, we use the alignment error rate (AER) of Och and Ney (2000), which measures agreement at the level of pairs of words:[1]

$$AER = 1 - \frac{2|A \cap G|}{|A| + |G|}$$

[1] While Och and Ney (2000) differentiate between *sure* and *possible* hand-annotated alignments, our gold standard alignments come in only one variety.

	Alignment Error Rate
IBM Model 1	.37
IBM Model 2	.35
IBM Model 3	.43
Tree-to-String	.42
Tree-to-String, Clone	.36
Tree-to-String, Clone $P_{ins} = .5$	.32
Tree-to-Tree	.49
Tree-to-Tree, Clone	.36

Table 2: Alignment error rate on Korean-English corpus

where A is the set of word pairs aligned by the automatic system, and G the set aligned in the gold standard. We provide a comparison of the tree-based models with the sequence of successively more complex models of Brown et al. (1993). Results are shown in Table 2.

The error rates shown in Table 2 represent the minimum over training iterations; training was stopped for each model when error began to increase. IBM Models 1, 2, and 3 refer to Brown et al. (1993). "Tree-to-String" is the model of Yamada and Knight (2001), and "Tree-to-String, Clone" allows the node cloning operation of Section 2.1. "Tree-to-Tree" indicates the model of Section 3, while "Tree-to-Tree, Clone" adds the node cloning operation of Section 3.1. Model 2 is initialized from the parameters of Model 1, and Model 3 is initialized from Model 2. The lexical translation probabilities $P_t(f|e)$ for each of our tree-based models are initialized from Model 1, and the node re-ordering probabilities are initialized uniformly. Figure 1 shows the viterbi alignment produced by the "Tree-to-String, Clone" system on one sentence from our test set.

We found better agreement with the human alignments when fixing $P_{ins}(\text{left})$ in the Tree-to-String model to a constant rather than letting it be determined through the EM training. While the model learned by EM tends to overestimate the total number of aligned word pairs, fixing a higher probability for insertions results in fewer total aligned pairs and therefore a better trade-off between precision and recall. As seen for other tasks (Carroll and Charniak, 1992; Merialdo, 1994), the likelihood criterion used in EM training may not be optimal when evaluating a system against human labeling. The approach of optimizing a small number of metaparameters has been applied to machine translation by Och and Ney (2002). It is likely that the IBM models could similarly be optimized to minimize alignment error – an open question is whether the optimization with respect to alignment error will correspond to optimization for translation accuracy.

Within the strict EM framework, we found roughly equivalent performance between the IBM models and the two tree-based models when making use of the cloning operation. For both the tree-to-string and tree-to-tree models, the cloning operation improved results, indicating that adding the flexibility to handle structural divergence is important when using syntax-based models. The improvement was particularly significant for the tree-to-tree model, because using syntactic trees on both sides of the translation pair, while desirable as an additional source of information, severely constrains possible alignments unless the cloning operation is allowed.

The tree-to-tree model has better theoretical complexity than the tree-to-string model, being quadratic rather than quartic in sentence length, and we found this to be a significant advantage in practice. This improvement in speed allows longer sentences and more data to be used in training syntax-based models. We found that when training on sentences of up 60 words, the tree-to-tree alignment was 20 times faster than tree-to-string alignment. For reasons of speed, Yamada and Knight (2002) limited training to sentences of length 30, and were able to use only one fifth of the available Chinese-English parallel corpus.

6 Conclusion

Our loosely tree-based alignment techniques allow statistical models of machine translation to make use of syntactic information while retaining the flexibility to handle cases of non-isomorphic source and target trees. This is achieved with a clone operation parameterized in such a way that alignment probabilities can be computed with no increase in asymptotic computational complexity.

We present versions of this technique both for tree-to-string models, making use of parse trees for one of the two languages, and tree-to-tree models, which make use of parallel parse trees. Results in terms of alignment error rate indicate that the clone operation results in better alignments in both cases. On our Korean-English corpus, we found roughly equivalent performance for the unstructured IBM models, and the both the tree-to-string and tree-to-tree models when using cloning. To our knowledge these are the first results in the literature for tree-to-tree statistical alignment. While we did not see a benefit in alignment error from using syntactic trees in both languages, there is a significant practical benefit in computational efficiency. We remain hopeful that two trees can provide more information than one, and feel that extensions to the "loosely" tree-based approach are likely to demonstrate this using larger corpora.

Another important question we plan to pursue is the degree to which these results will be borne out with larger corpora, and how the models may be refined as more training data is available. As one example, our tree representation is unlexicalized, but we expect conditioning the model on more lexical information to improve results, whether this is done by percolating lexical heads through the existing trees or by switching to a strict dependency representation.

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.

G. Edward Barton, Jr. 1985. On the complexity of ID/LP parsing. *Computational Linguistics*, 11(4):205–218.

Peter F. Brown, John Cocke, Stephen A. Della Pietra, Vincent J. Della Pietra, Frederick Jelinek, John D. Lafferty, 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.

Glenn Carroll and Eugene Charniak. 1992. Two experiments on learning probabilistic dependency grammars from corpora. In *Workshop Notes for Statistically-Based NLP Techniques*, pages 1–13. AAAI.

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

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

Jan Hajič, Martin Čmejrek, Bonnie Dorr, Yuan Ding, Jason Eisner, Daniel Gildea, Terry Koo, Kristen Parton, Gerald Penn, Dragomir Radev, and Owen Rambow. 2002. Natural language generation in the context of machine translation. Technical report, Center for Language and Speech Processing, Johns Hopkins University, Baltimore. Summer Workshop Final Report.

Chung-hye Han, Na-Rae Han, and Eon-Suk Ko. 2001. Bracketing guidelines for Penn Korean treebank. Technical Report IRCS-01-010, IRCS, University of Pennsylvania.

Bernard Merialdo. 1994. Tagging English text with a probabilistic model. *Computational Linguistics*, 20(2):155–172.

Franz Josef Och and Hermann Ney. 2000. Improved statistical alignment models. In *Proceedings of ACL-00*, pages 440–447, Hong Kong, October.

Franz Josef Och and Hermann Ney. 2002. Discriminative training and maximum entropy models for statistical machine translation. In *Proceedings of ACL-02*, Philadelphia, PA.

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

Kenji Yamada and Kevin Knight. 2001. A syntax-based statistical translation model. In *Proceedings of ACL-01*, Toulouse, France.

Kenji Yamada and Kevin Knight. 2002. A decoder for syntax-based statistical MT. In *Proceedings of ACL-02*, Philadelphia, PA.