Stochastic Lexicalized Inversion Transduction Grammar for Alignment

Hao Zhang and **Daniel Gildea**
Computer Science Department
University of Rochester
Rochester, NY 14627

Abstract

We present a version of Inversion Transduction Grammar where rule probabilities are lexicalized throughout the synchronous parse tree, along with pruning techniques for efficient training. Alignment results improve over unlexicalized ITG on short sentences for which full EM is feasible, but pruning seems to have a negative impact on longer sentences.

1 Introduction

The Inversion Transduction Grammar (ITG) of Wu (1997) is a syntactically motivated algorithm for producing word-level alignments of pairs of translationally equivalent sentences in two languages. The algorithm builds a synchronous parse tree for both sentences, and assumes that the trees have the same underlying structure but that the ordering of constituents may differ in the two languages.

This probabilistic, syntax-based approach has inspired much subsequent reasearch. Alshawi et al. (2000) use hierarchical finite-state transducers. In the tree-to-string model of Yamada and Knight (2001), a parse tree for one sentence of a translation pair is projected onto the other string. Melamed (2003) presents algorithms for synchronous parsing with more complex grammars, discussing how to parse grammars with greater than binary branching and lexicalization of synchronous grammars.

Despite being one of the earliest probabilistic syntax-based translation models, ITG remains state-of-the art. Zens and Ney (2003) found that the constraints of ITG were a better match to the decoding task than the heuristics used in the IBM decoder of Berger et al. (1996). Zhang and Gildea (2004) found ITG to outperform the tree-to-string model for word-level alignment, as measured against human gold-standard alignments. One explanation for this result is that, while a tree representation is helpful for modeling translation, the trees assigned by the traditional monolingual parsers (and the treebanks on which they are trained) may not be optimal for translation of a specific language pair. ITG has the advantage of being entirely data-driven – the trees are derived from an expectation maximization procedure given only the original strings as input.

In this paper, we extend ITG to condition the grammar production probabilities on lexical information throughout the tree. This model is reminiscent of lexicalization as used in modern statistical parsers, in that a unique head word is chosen for each constituent in the tree. It differs in that the head words are chosen through EM rather than deterministic rules. This approach is designed to retain the purely data-driven character of ITG, while giving the model more information to work with. By conditioning on lexical information, we expect the model to be able capture the same systematic differences in languages' grammars that motive the tree-to-string model, for example, SVO vs. SOV word order or prepositions vs. postpositions, but to be able to do so in a more fine-grained manner. The interaction between lexical information and word order also explains the higher performance of IBM model 4 over IBM model 3 for alignment.

We begin by presenting the probability model in the following section, detailing how we address issues of pruning and smoothing that lexicalization introduces. We present alignment results on a parallel Chinese-English corpus in Section 3.

Proceedings of the 43rd Annual Meeting of the ACL, pages 475–482,
Ann Arbor, June 2005. ©2005 Association for Computational Linguistics

2 Lexicalization of Inversion Transduction Grammars

An Inversion Transduction Grammar can generate pairs of sentences in two languages by recursively applying context-free bilingual production rules. Most work on ITG has focused on the 2-normal form, which consists of unary production rules that are responsible for generating word pairs:

$$X \rightarrow e/f$$

and binary production rules in two forms that are responsible for generating syntactic subtree pairs:

$$X \rightarrow [YZ]$$

and

$$X \rightarrow \langle YZ \rangle$$

The rules with square brackets enclosing the right hand side expand the left hand side symbol into the two symbols on the right hand side in the same order in the two languages, whereas the rules with pointed brackets expand the left hand side symbol into the two right hand side symbols in reverse order in the two languages.

One special case of ITG is the bracketing ITG that has only one nonterminal that instantiates exactly one straight rule and one inverted rule. The ITG we apply in our experiments has more structural labels than the primitive bracketing grammar: it has a start symbol S, a single preterminal C, and two intermediate nonterminals A and B used to ensure that only one parse can generate any given word-level alignment, as discussed by Wu (1997) and Zens and Ney (2003).

As an example, Figure 1 shows the alignment and the corresponding parse tree for the sentence pair *Je les vois / I see them* using the unambiguous bracketing ITG.

A stochastic ITG can be thought of as a stochastic CFG extended to the space of bitext. The independence assumptions typifying S-CFGs are also valid for S-ITGs. Therefore, the probability of an S-ITG parse is calculated as the product of the probabilities of all the instances of rules in the parse tree. For instance, the probability of the parse in Figure 1 is:

$$\begin{aligned} &P(S \rightarrow A) \cdot P(A \rightarrow [CB]) \\ &\quad \cdot P(B \rightarrow \langle CC \rangle) \cdot P(C \rightarrow I/Je) \\ &\quad \cdot P(C \rightarrow see/vois) \cdot P(C \rightarrow them/les) \end{aligned}$$

It is important to note that besides the bottom-level word-pairing rules, the other rules are all nonlexical, which means the structural alignment component of the model is not sensitive to the lexical contents of subtrees. Although the ITG model can effectively restrict the space of alignment to make polynomial time parsing algorithms possible, the preference for inverted or straight rules only passively reflect the need of bottom level word alignment. We are interested in investigating how much help it would be if we strengthen the structural alignment component by making the orientation choices dependent on the real lexical pairs that are passed up from the bottom.

The first step of lexicalization is to associate a lexical pair with each nonterminal. The head word pair generation rules are designed for this purpose:

$$X \rightarrow X(e/f)$$

The word pair e/f is representative of the lexical content of X in the two languages.

For binary rules, the mechanism of head selection is introduced. Now there are 4 forms of binary rules:

$$X(e/f) \rightarrow [Y(e/f)Z]$$

$$X(e/f) \rightarrow [YZ(e/f)]$$

$$X(e/f) \rightarrow \langle Y(e/f)Z \rangle$$

$$X(e/f) \rightarrow \langle YZ(e/f) \rangle$$

determined by the four possible combinations of head selections (Y or Z) and orientation selections (straight or inverted).

The rules for generating lexical pairs at the leaves of the tree are now predetermined:

$$X(e/f) \rightarrow e/f$$

Putting them all together, we are able to derive a lexicalized bilingual parse tree for a given sentence pair. In Figure 2, the example in Figure 1 is revisited. The probability of the lexicalized parse is:

$$\begin{aligned} &P(S \rightarrow S(see/vois)) \\ &\quad \cdot P(S(see/vois) \rightarrow A(see/vois)) \\ &\quad \cdot P(A(see/vois) \rightarrow [CB(see/vois)]) \\ &\quad \cdot P(C \rightarrow C(I/Je)) \end{aligned}$$

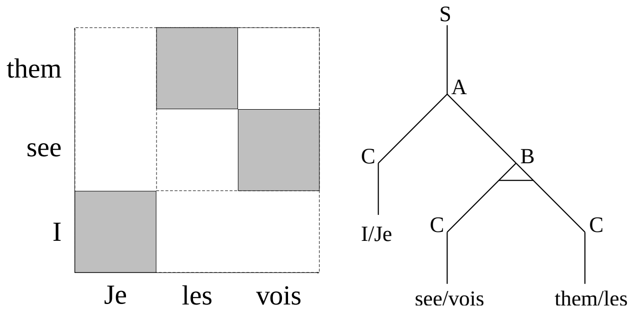

Figure 1: ITG Example

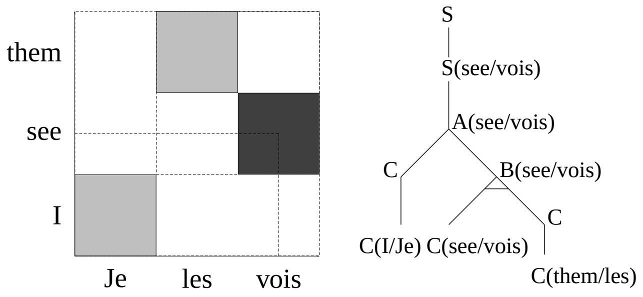

Figure 2: Lexicalized ITG Example. *see/vois* is the headword of both the 2x2 cell and the entire alignment.

$$\cdot P(B(see/vois) \rightarrow \langle C(see/vois)C\rangle)$$
$$\cdot P(C \rightarrow C(them/les))$$

The factors of the product are ordered to show the generative process of the most probable parse. Starting from the start symbol S, we first choose the head word pair for S, which is *see/vois* in the example. Then, we recursively expand the lexicalized head constituents using the lexicalized structural rules. Since we are only lexicalizing rather than bilexicalizing the rules, the non-head constituents need to be lexicalized using head generation rules so that the top-down generation process can proceed in all branches. By doing so, word pairs can appear at all levels of the final parse tree in contrast with the unlexicalized parse tree in which the word pairs are generated only at the bottom.

The binary rules are lexicalized rather than bilexicalized.[1] This is a trade-off between complexity and expressiveness. After our lexicalization, the number of lexical rules, thus the number of parameters in the statistical model, is still at the order of $O(|V||T|)$, where $|V|$ and $|T|$ are the vocabulary sizes of the two languages.

2.1 Parsing

Given a bilingual sentence pair, a synchronous parse can be built using a two-dimensional extension of chart parsing, where chart items are indexed by their nonterminal X, head word pair e/f if specified, beginning and ending positions l, m in the source language string, and beginning and ending positions i, j in the target language string. For Expectation Maximization training, we compute lexicalized inside probabilities $\beta(X(e/f), l, m, i, j)$, as well as unlexicalized inside probabilities $\beta(X, l, m, i, j)$, from the bottom up as outlined in Algorithm 1.

The algorithm has a complexity of $O(N_s^4 N_t^4)$, where N_s and N_t are the lengths of source and target sentences respectively. The complexity of parsing for an unlexicalized ITG is $O(N_s^3 N_t^3)$. Lexicalization introduces an additional factor of $O(N_s N_t)$, caused by the choice of headwords e and f in the pseudocode.

Assuming that the lengths of the source and target sentences are proportional, the algorithm has a complexity of $O(n^8)$, where n is the average length of the source and target sentences.

[1] In a sense our rules are bilexicalized in that they condition on words from both languages; however they do not capture head-modifier relations within a language.

Algorithm 1 LexicalizedITG(s, t)

```
for all $l, m$ such that $0 \leq l \leq m \leq N_s$ do
  for all $i, j$ such that $0 \leq i \leq j \leq N_t$ do
    for all $e \in \{e_{l+1} \dots e_m\}$ do
      for all $f \in \{f_{i+1} \dots f_j\}$ do
        for all $n$ such that $l \leq n \leq m$ do
          for all $k$ such that $i \leq k \leq j$ do
            for all rules $X \rightarrow YZ \in G$ do
              $\beta(X(e/f), l, m, i, j)$ +=
              ▷ straight rule, where $Y$ is head
                 $P([Y(e/f)Z] \mid X(e/f)) \cdot \beta(Y(e/f), l, n, i, k) \cdot \beta(Z, n, m, k, j)$
              ▷ inverted rule, where $Y$ is head
               $+ P(\langle Y(e/f)Z \rangle \mid X(e/f)) \cdot \beta(Y(e/f), n, m, i, k) \cdot \beta(Z, l, n, k, j)$
              ▷ straight rule, where $Z$ is head
               $+ P([YZ(e/f)] \mid X(e/f)) \cdot \beta(Y, l, n, i, k) \cdot \beta(Z(e/f), n, m, k, j)$
              ▷ inverted rule, where $Z$ is head
               $+ P(\langle YZ(e/f) \rangle \mid X(e/f)) \cdot \beta(Y, n, m, i, k) \cdot \beta(Z(e/f), l, n, k, j)$
            end for
          end for
        end for
        ▷ word pair generation rule
        $\beta(X, l, m, i, j)$ += $P(X(e/f) \mid X) \cdot \beta(X(e/f), l, m, i, j)$
      end for
    end for
  end for
end for
```

2.2 Pruning

We need to further restrict the space of alignments spanned by the source and target strings to make the algorithm feasible. Our technique involves computing an estimate of how likely each of the n^4 cells in the chart is before considering all ways of building the cell by combining smaller subcells. Our figure of merit for a cell involves an estimate of both the inside probability of the cell (how likely the words within the box in both dimensions are to align) and the outside probability (how likely the words outside the box in both dimensions are to align). In including an estimate of the outside probability, our technique is related to A* methods for monolingual parsing (Klein and Manning, 2003), although our estimate is not guaranteed to be lower than complete outside probabity assigned by ITG. Figure 3(a) displays the tic-tac-toe pattern for the inside and outside components of a particular cell. We use IBM Model 1 as our estimate of both the inside and outside probabilities. In the Model 1 estimate of the outside probability, source and target words can align using any combination of points from the four outside corners of the tic-tac-toe pattern. Thus in Figure 3(a), there is one solid cell (corresponding to the Model 1 Viterbi alignment) in each column, falling either in the upper or lower outside shaded corner. This can be also be thought of as squeezing together the four outside corners, creating a new cell whose probability is estimated using IBM Model 1. Mathematically, our figure of merit for the cell (l, m, i, j) is a product of the inside Model 1 probability and the outside Model 1 probability:

$$
\begin{aligned}
P(\mathbf{f}_{(i,j)} \mid \mathbf{e}_{(l,m)}) \cdot P(\mathbf{f}_{\overline{(i,j)}} \mid \mathbf{e}_{\overline{(l,m)}}) \qquad (1) \\
= \lambda_{|(l,m)|,|(i,j)|} \prod_{t \in (i,j)} \sum_{s \in \{0,(l,m)\}} t(f_t \mid e_s) \\
\cdot \lambda_{|\overline{(l,m)}|,|\overline{(i,j)}|} \prod_{t \in \overline{(i,j)}} \sum_{s \in \{0,\overline{(l,m)}\}} t(f_t \mid e_s)
\end{aligned}
$$

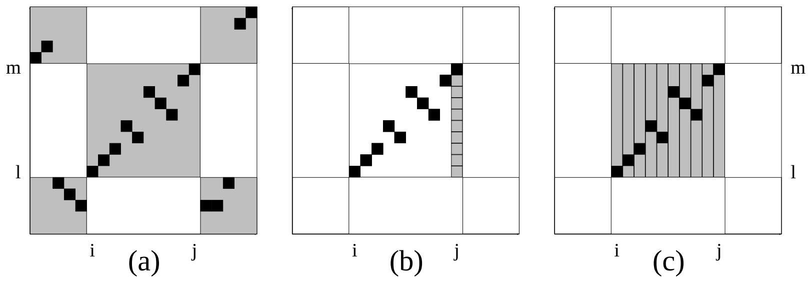

Figure 3: The tic-tac-toe figure of merit used for pruning bitext cells. The shaded regions in (a) show alignments included in the figure of merit for bitext cell (l, m, i, j) (Equation 1); solid black cells show the Model 1 Viterbi alignment within the shaded area. (b) shows how to compute the inside probability of a unit-width cell by combining basic cells (Equation 2), and (c) shows how to compute the inside probability of any cell by combining unit-width cells (Equation 3).

where $\overline{(l, m)}$ and $\overline{(i, j)}$ represent the complementary spans in the two languages. λ_{L_1,L_2} is the probability of any word alignment template for a pair of L_1-word source string and L_2-word target string, which we model as a uniform distribution of word-for-word alignment patterns after a Poisson distribution of target string's possible lengths, following Brown et al. (1993). As an alternative, the $\sum$ operator can be replaced by the $\max$ operator as the inside operator over the translation probabilities above, meaning that we use the Model 1 Viterbi probability as our estimate, rather than the total Model 1 probability.[2]

A naïve implementation would take $O(n^6)$ steps of computation, because there are $O(n^4)$ cells, each of which takes $O(n^2)$ steps to compute its Model 1 probability. Fortunately, we can exploit the recursive nature of the cells. Let $\mathrm{INS}(l, m, i, j)$ denote the major factor of our Model 1 estimate of a cell's inside probability, $\prod_{t\in(i,j)} \sum_{s\in\{0,(l,m)\}} t(f_t \mid e_s)$. It turns out that one can compute cells of width one ($i = j$) in constant time from a cell of equal width and lower height:

$$\begin{aligned}\mathrm{INS}(l, m, j, j) &= \prod_{t\in(j,j)} \sum_{s\in\{0,(l,m)\}} t(f_t \mid e_s) \\ &= \sum_{s\in\{0,(l,m)\}} t(f_j \mid e_s) \\ &= \mathrm{INS}(l, m-1, j, j) \\ &\quad + t(f_j \mid e_m) \qquad (2)\end{aligned}$$

Similarly, one can compute cells of width greater than one by combining a cell of one smaller width with a cell of width one:

$$\begin{aligned}\mathrm{INS}(l, m, i, j) &= \prod_{t\in(i,j)} \sum_{s\in\{0,(l,m)\}} t(f_t \mid e_s) \\ &= \prod_{t\in(i,j)} \mathrm{INS}(l, m, t, t) \\ &= \mathrm{INS}(l, m, i, j-1) \\ &\quad \cdot \mathrm{INS}(l, m, j, j) \qquad (3)\end{aligned}$$

Figure 3(b) and (c) illustrate the inductive computation indicated by the two equations. Each of the $O(n^4)$ inductive steps takes one additive or multiplicative computation. A similar dynammic programing technique can be used to efficiently compute the outside component of the figure of merit. Hence, the algorithm takes just $O(n^4)$ steps to compute the figure of merit for all cells in the chart.

Once the cells have been scored, there can be many ways of pruning. In our experiments, we applied beam ratio pruning to each individual bucket of cells sharing a common source substring. We prune cells whose probability is lower than a fixed ratio below the best cell for the same source substring. As a result, at least one cell will be kept for each source substring. We safely pruned more than 70% of cells using 10^{-5} as the beam ratio for sentences up to 25 words. Note that this pruning technique is applicable to both the lexicalized ITG and the conventional ITG.

In addition to pruning based on the figure of merit described above, we use top-k pruning to limit the number of hypotheses retained for each cell. This is necessary for lexicalized ITG because the number of distinct hypotheses in the two-dimensional ITG

[2] The experimental difference of the two alternatives was small. For our results, we used the $\max$ version.

chart has increased to $O(N_s^3 N_t^3)$ from $O(N_s^2 N_t^2)$ due to the choice one of $O(N_s)$ source language words and one of $O(N_t)$ target language words as the head. We keep only the top-k lexicalized items for a given chart cell of a certain nonterminal Y contained in the cell l, m, i, j. Thus the additional complexity of $O(N_s N_t)$ will be replaced by a constant factor.

The two pruning techniques can work for both the computation of expected counts during the training process and for the Viterbi-style algorithm for extracting the most probable parse after training. However, if we initialize EM from a uniform distribution, all probabilties are equal on the first iteration, giving us no basis to make pruning decisions. So, in our experiments, we initialize the head generation probabilities of the form $P(X(e/f) \mid X)$ to be the same as $P(e/f \mid C)$ from the result of the unlexicalized ITG training.

2.3 Smoothing

Even though we have controlled the number of parameters of the model to be at the magnitude of $O(|V||T|)$, the problem of data sparseness still renders a smoothing method necessary. We use backing off smoothing as the solution. The probabilities of the unary head generation rules are in the form of $P(X(e/f) \mid X)$. We simply back them off to the uniform distribution. The probabilities of the binary rules, which are conditioned on lexicalized nonterminals, however, need to be backed off to the probabilities of generalized rules in the following forms:

$$P([Y(*)Z] \mid X(*))$$

$$P([YZ(*)] \mid X(*))$$

$$P(\langle Y(*)Z \rangle \mid X(*))$$

$$P(\langle YZ(*) \rangle \mid X(*))$$

where $*$ stands for any lexical pair. For instance,

$$\begin{aligned} P([Y(e/f)Z] \mid X(e/f)) = \\ &(1-\lambda)P_{EM}([Y(e/f)Z] \mid X(e/f)) \\ &+ \lambda P([Y(*)Z] \mid X(*)) \end{aligned}$$

where

$$\lambda = 1/(1 + Expected\_Counts(X(e/f)))$$

The more often $X(e/f)$ occurred, the more reliable are the estimated conditional probabilities with the condition part being $X(e/f)$.

3 Experiments

We trained both the unlexicalized and the lexicalized ITGs on a parallel corpus of Chinese-English newswire text. The Chinese data were automatically segmented into tokens, and English capitalization was retained. We replaced words occurring only once with an unknown word token, resulting in a Chinese vocabulary of 23,783 words and an English vocabulary of 27,075 words.

In the first experiment, we restricted ourselves to sentences of no more than 15 words in either language, resulting in a training corpus of 6,984 sentence pairs with a total of 66,681 Chinese words and 74,651 English words. In this experiment, we didn't apply the pruning techniques for the lexicalized ITG.

In the second experiment, we enabled the pruning techniques for the LITG with the beam ratio for the tic-tac-toe pruning as 10^{-5} and the number k for the top-k pruning as 25. We ran the experiments on sentences up to 25 words long in both languages. The resulting training corpus had 18,773 sentence pairs with a total of 276,113 Chinese words and 315,415 English words.

We evaluate our translation models in terms of 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:

$$AER = 1 - \frac{|A \cap G_P| + |A \cap G_S|}{|A| + |G_S|}$$

where A is the set of word pairs aligned by the automatic system, G_S is the set marked in the gold standard as "sure", and G_P is the set marked as "possible" (including the "sure" pairs). In our Chinese-English data, only one type of alignment was marked, meaning that $G_P = G_S$.

In our hand-aligned data, 20 sentence pairs are less than or equal to 15 words in both languages, and were used as the test set for the first experiment, and 47 sentence pairs are no longer than 25 words in either language and were used to evaluate the pruned

	Precision	*Recall*	*Alignment Error Rate*
IBM Model 1	.59	.37	.54
IBM Model 4	.63	.43	.49
ITG	.62	.47	.46
Lexicalized ITG	.66	.50	.43

Table 1: Alignment results on Chinese-English corpus ($\leq$ 15 words on both sides). Full ITG vs. Full LITG

	Precision	*Recall*	*Alignment Error Rate*
IBM Model 1	.56	.42	.52
IBM Model 4	.67	.43	.47
ITG	.68	.52	.40
Lexicalized ITG	.69	.51	.41

Table 2: Alignment results on Chinese-English corpus ($\leq$ 25 words on both sides). Full ITG vs. Pruned LITG

LITG against the unlexicalized ITG.

A separate development set of hand-aligned sentence pairs was used to control overfitting. The subset of up to 15 words in both languages was used for cross-validating in the first experiment. The subset of up to 25 words in both languages was used for the same purpose in the second experiment.

Table 1 compares results using the full (unpruned) model of unlexicalized ITG with the full model of lexicalized ITG.

The two models were initialized from uniform distributions for all rules and were trained until AER began to rise on our held-out cross-validation data, which turned out to be 4 iterations for ITG and 3 iterations for LITG.

The results from the second experiment are shown in Table 2. The performance of the full model of unlexicalized ITG is compared with the pruned model of lexicalized ITG using more training data and evaluation data.

Under the same check condition, we trained ITG for 3 iterations and the pruned LITG for 1 iteration.

For comparison, we also included the results from IBM Model 1 and Model 4. The numbers of iterations for the training of the IBM models were chosen to be the turning points of AER changing on the cross-validation data.

4 Discussion

As shown by the numbers in Table 1, the full lexicalized model produced promising alignment results on sentence pairs that have no more than 15 words on both sides. However, due to its prohibitive $O(n^8)$ computational complexity, our C++ implementation of the unpruned lexicalized model took more than 500 CPU hours, which were distributed over multiple machines, to finish one iteration of training. The number of CPU hours would increase to a point that is unacceptable if we doubled the average sentence length. Some type of pruning is a must-have. Our pruned version of LITG controlled the running time for one iteration to be less than 1200 CPU hours, despite the fact that both the number of sentences and the average length of sentences were more than doubled. To verify the safety of the tic-tac-toe pruning technique, we applied it to the unlexicalized ITG using the same beam ratio (10^{-5}) and found that the AER on the test data was not changed. However, whether or not the top-k lexical head pruning technique is equally safe remains a question. One noticeable implication of this technique for training is the reliance on initial probabilities of lexical pairs that are discriminative enough. The comparison of results for ITG and LITG in Table 2 and the fact that AER began to rise after only one iteration of training seem to indicate that keeping few distinct lexical heads caused convergence on a suboptimal set of parameters, leading to a form of overfitting. In contrast, overfitting did not seem to be a problem for LITG in the unpruned experiment of Table 1, despite the much larger number of parameters for LITG than for ITG and the smaller training set.

We also want to point out that for a pair of long sentences, it would be hard to reflect the inherent bilingual syntactic structure using the lexicalized binary bracketing parse tree. In Figure 2, $A(see/vois)$ echoes $IP(see/vois)$ and $B(see/vois)$ echoes $VP(see/vois)$ so that it means $IP(see/vois)$ is not inverted from English to French but its right child $VP(see/vois)$ is inverted. However, for longer sentences with more than 5 levels of bracketing and the same lexicalized nonterminal repeatedly appearing at different levels, the correspondences would become less linguistically plausible. We think the limitations of the bracketing grammar are another reason for not being able to improve the AER of longer sentence pairs after lexicalization.

The space of alignments that is to be considered by LITG is exactly the space considered by ITG since the structural rules shared by them define the alignment space. The lexicalized ITG is designed to be more sensitive to the lexical influence on the choices of inversions so that it can find better alignments. Wu (1997) demonstrated that for pairs of sentences that are less than 16 words, the ITG alignment space has a good coverage over all possibilities. Hence, it's reasonable to see a better chance of improving the alignment result for sentences less than 16 words.

5 Conclusion

We presented the formal description of a Stochastic Lexicalized Inversion Transduction Grammar with its EM training procedure, and proposed specially designed pruning and smoothing techniques. The experiments on a parallel corpus of Chinese and English showed that lexicalization helped for aligning sentences of up to 15 words on both sides. The pruning and the limitations of the bracketing grammar may be the reasons that the result on sentences of up to 25 words on both sides is not better than that of the unlexicalized ITG.

Acknowledgments We are very grateful to Rebecca Hwa for assistance with the Chinese-English data, to Kevin Knight and Daniel Marcu for their feedback, and to the authors of GIZA. This work was partially supported by NSF ITR IIS-09325646 and NSF ITR IIS-0428020.

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.

Adam Berger, Peter Brown, Stephen Della Pietra, Vincent Della Pietra, J. R. Fillett, Andrew Kehler, and Robert Mercer. 1996. Language translation apparatus and method of using context-based tanslation models. United States patent 5,510,981.

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.

Dan Klein and Christopher D. Manning. 2003. A* parsing: Fast exact viterbi parse selection. In *Proceedings of the 2003 Meeting of the North American chapter of the Association for Computational Linguistics (NAACL-03)*.

I. Dan Melamed. 2003. Multitext grammars and synchronous parsers. In *Proceedings of the 2003 Meeting of the North American chapter of the Association for Computational Linguistics (NAACL-03)*, Edmonton.

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.

Dekai Wu. 1997. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computational Linguistics*, 23(3):377–403.

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.

Richard Zens and Hermann Ney. 2003. A comparative study on reordering constraints in statistical machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, Sapporo, Japan.

Hao Zhang and Daniel Gildea. 2004. Syntax-based alignment: Supervised or unsupervised? In *Proceedings of the 20th International Conference on Computational Linguistics (COLING-04)*, Geneva, Switzerland, August.