Unsupervised Syntactic Alignment with Inversion Transduction Grammars

Adam Pauls  Dan Klein
Computer Science Division
University of California at Berkeley
{adpauls,klein}@cs.berkeley.edu

David Chiang  Kevin Knight
Information Sciences Institute
University of Southern California
{chiang,knight}@isi.edu

Abstract

Syntactic machine translation systems currently use word alignments to infer syntactic correspondences between the source and target languages. Instead, we propose an unsupervised ITG alignment model that directly aligns syntactic structures. Our model aligns spans in a source sentence to nodes in a target parse tree. We show that our model produces syntactically consistent analyses where possible, while being robust in the face of syntactic divergence. Alignment quality and end-to-end translation experiments demonstrate that this consistency yields higher quality alignments than our baseline.

1 Introduction

Syntactic machine translation has advanced significantly in recent years, and multiple variants currently achieve state-of-the-art translation quality. Many of these systems exploit linguistically-derived syntactic information either on the target side (Galley et al., 2006), the source side (Huang et al., 2006), or both (Liu et al., 2009). Still others induce their syntax from the data (Chiang, 2005). Despite differences in detail, the vast majority of syntactic methods share a critical dependence on word alignments. In particular, they infer syntactic correspondences between the source and target languages through word alignment patterns, sometimes in combination with constraints from parser outputs.

However, word alignments are not perfect indicators of syntactic alignment, and syntactic systems are very sensitive to word alignment behavior. Even a single spurious word alignment can invalidate a large number of otherwise extractable rules, while unaligned words can result in an exponentially large set of extractable rules to choose from. Researchers have worked to incorporate syntactic information into word alignments, resulting in improvements to both alignment quality (Cherry and Lin, 2006; DeNero and Klein, 2007), and translation quality (May and Knight, 2007; Fossum et al., 2008).

In this paper, we remove the dependence on word alignments and instead directly model the syntactic correspondences in the data, in a manner broadly similar to Yamada and Knight (2001). In particular, we propose an unsupervised model that aligns nodes of a parse tree (or forest) in one language to spans of a sentence in another. Our model is an instance of the inversion transduction grammar (ITG) formalism (Wu, 1997), constrained in such a way that one side of the synchronous derivation respects a syntactic parse. Our model is best suited to systems which use source- or target-side trees only.

The design of our model is such that, for divergent structures, a structurally integrated backoff to flatter word-level (or null) analyses is available. Therefore, our model is empirically robust to the case where syntactic divergence between languages prevents syntactically accurate ITG derivations.

We show that, with appropriate pruning, our model can be efficiently trained on large parallel corpora. When compared to standard word-alignment-backed baselines, our model produces more consistent analyses of parallel sentences, leading to high-count, high-quality transfer rules. End-to-end translation experiments demonstrate that these higher quality rules improve translation quality by 1.0 BLEU over a word-alignment-backed baseline.

2 Syntactic Rule Extraction

Our model is intended for use in syntactic translation models which make use of syntactic parses on either the target (Galley et al., 2006) or source side (Huang et al., 2006; Liu et al., 2006). Our model’s
Figure 1: A single incorrect alignment removes an extractable node, and hence several desirable rules. We represent correct extractable nodes in bold, spurious extractable nodes with an *, and incorrectly blocked extractable nodes in bold strike-through.

Chief purpose is to align nodes in the syntactic parse in one language to spans in the other – an alignment we will refer to as a “syntactic” alignment. These alignments are employed by standard syntactic rule extraction algorithms, for example, the GHKM algorithm of Galley et al. (2004). Following that work, we will assume parses are present in the target language, though our model applies in either direction.

Currently, although syntactic systems make use of syntactic alignments, these alignments must be induced indirectly from word-level alignments. Previous work has discussed at length the poor interaction of word-alignments with syntactic rule extraction (DeNero and Klein, 2007; Fossum et al., 2008). While syntactically aware, these models remain limited by the word alignment models that underly them.

Here, we describe a model which directly infers alignments of nodes in the target-language parse tree to spans of the source sentence. Formally, our model is an instance of a Synchronous Context-Free Grammar (see Chiang (2004) for a review), or SCFG, which generates an English (target) parse tree $T$ and foreign (source) sentence $f$ given a target sentence $e$. The generative process underlying this model produces a derivation $d$ of SCFG rules, from which $T$ and $f$ can be read off; because we condition on $e$, the derivations produce $e$ with probability 1. This model places a distribution over $T$ and $f$ given by

$$p(T, f | e) = \sum_d p(d | e) = \sum_d \prod_{r \in d} p(r | e)$$

where the sum is over derivations $d$ which yield $T$ and $f$. The SCFG rules $r$ come from one of 4 types, pictured in Table 1. In general, because our model can generate English trees, it permits inference over forests. Although we will restrict ourselves to a single parse tree for our experiments, in this section, we discuss the more general case.

1One notable exception is May and Knight (2007), who produces syntactic alignments using syntactic rules derived from word-aligned data.
<table>
<thead>
<tr>
<th>Rule Type</th>
<th>Root</th>
<th>English</th>
<th>Foreign</th>
<th>Example Instantiation</th>
</tr>
</thead>
<tbody>
<tr>
<td>TERMINAL</td>
<td>E</td>
<td>∈</td>
<td>f₀</td>
<td>FOUR → four ; 四</td>
</tr>
<tr>
<td>UNARY</td>
<td>A</td>
<td>B</td>
<td>f₁</td>
<td>CD → FOUR ; ∈ FOUR 个</td>
</tr>
<tr>
<td>BINARYMONO</td>
<td>A</td>
<td>B C</td>
<td>f₂ B m C f₅</td>
<td>NP → NN NN ; ∈ NN 的 NN ∈</td>
</tr>
<tr>
<td>BINARYINV</td>
<td>A</td>
<td>B C</td>
<td>f₅ C m C B ∈</td>
<td>PP → IN NP ; 在 NP ∈ IN ∈</td>
</tr>
</tbody>
</table>

Table 1: Types of rules present in the SCFG describing our model, along with some sample instantiations of each type. Empty word sequences f have been explicitly marked with an $\epsilon$.

The first rule type is the TERMINAL production, which rewrites a terminal symbol typically $E$ as its English word $\epsilon$ and a (possibly empty) sequence of foreign words $f₀$. Generally speaking, the majority of foreign words are generated using this rule. It is only when a straightforward word-to-word correspondence cannot be found that our model resorts to generating foreign words elsewhere.

We can also rewrite a non-terminal symbol $A$ using a UNARY production, which on the English side produces a single symbol $B$, and on the foreign side produces the symbol $B$, with sequences of words $f₁$ to its left and $f₅$ to its right.

Finally, there are two binary productions: BINARYMONO rewrites $A$ with two non-terminals $B$ and $C$ on the English side, and the same non-terminals $B$ and $C$ in monotonic order on the foreign side, with sequences of words $f₂$, $f₅$, and $f₅$ to the left, right, and the middle. BINARYINV inverts the order in which the non-terminals $B$ and $C$ are written on the source side, allowing our model to capture a large subset of possible reorderings (Wu, 1997).

Derivations from this model have two key properties: first, the English side of a derivation is constrained to form a valid constituency parse, as is required in a syntax system with target-side syntax; and second, for each parse node in the English projection, there is exactly one (possibly empty) contiguous span of the foreign side which was generated from that non-terminal or one of its descendants. Identifying extractable nodes from a derivation is thus trivial: any node aligned to a non-empty foreign span is extractable.

In Figure 2, we show a sample sentence pair fragment under our model. The English projection of the derivation represents a valid constituency parse, while the foreign projection is less constrained. We connect each foreign terminal with a dashed line to the node in the English side of the synchronous derivation at which it is generated. The foreign span assigned to each English node is indicated with indices. All nodes with non-empty spans, shown in boldface, are extractable nodes. BOTTOM: The SCFG rules used in the derivation.

---

2For notational convenience, we imagine that for each particular English word $\epsilon$, there is a special preterminal symbol $E$ which produces it. These symbols $E$ act like any other non-terminal in the grammar with respect to the parameterization in Section 3.1. To denote standard non-terminals, we will use $A$, $B$, and $C$.

3While our model does not explicitly produce many-to-one alignments, many-to-one rules can be discovered via rule composition (Galley et al., 2006).
3.1 Parameterization

In principle, our model could have one parameter for each instantiation \( r \) of a rule type. This model would have an unmanageable number of parameters, producing both computational and modeling issues—it is well known that unsupervised models with large numbers of parameters are prone to degenerate analyses of the data (DeNero et al., 2006). One solution might be to apply an informed prior with a computationally tractable inference procedure (e.g. Cohn and Blunsom (2009) or Liu and Gildea (2009)). We opt here for the simpler, statistically more robust solution of making independence assumptions to keep the number of parameters at a reasonable level.

Concretely, we define the probability of the BinaryMono rule,

\[
p(r = A \rightarrow B C; f_l B f_m C f_r | A, e_A)
\]

which conditions on the root of the rule \( A \) and the English yield \( e_A \), as

\[
p_g(A \rightarrow B C | A, e_A) \cdot p_{\text{inv}}(I | B, C) \cdot p_{l e f t}(f_l | A, e_A) \cdot p_{\text{mid}}(f_m | A, e_A) \cdot p_{\text{right}}(f_r | A, e_A)
\]

In words, we assume that the rule probability decomposes into a monolingual PCFG grammar probability \( p_g \), an inversion probability \( p_{\text{inv}} \), and a probability of left, middle, and right word sequences \( p_{l e f t}, p_{\text{mid}}, \) and \( p_{\text{right}} \). Because we condition on \( e \), the monolingual grammar probability \( p_g \) must form a distribution which produces \( e \) with probability 1.\(^4\)

We further assume that the probability of producing a foreign word sequence \( f_l \) decomposes as:

\[
p_{l e f t}(f_l | A, e_A) = p_l(|(f_l| = m | A) \prod_{j=1}^{m} p(f_j | A, e_A)
\]

where \( m \) is the length of the sequence \( f_l \). The parameter \( p_l \) is a left length distribution. The probabilities \( p_{\text{mid}}, p_{\text{right}} \) decompose in the same way, except substituting a separate length distribution \( p_m \) and \( p_r \) for \( p_l \). For the TERMINAL rule, we emit \( f_l \) with a similarly decomposed distribution \( p_{\text{term}} \) using length distribution \( p_{w} \).

We define the probability of generating a foreign word \( f_j \) as

\[
p(f_j | A, e_A) = \sum_{i \in e_A} \frac{1}{e_A} p_t(f_j | e_i)
\]

with \( i \in e_A \) denoting an index ranging over the indices of the English words contained in \( e_A \). The reader may recognize the above expressions as the probability assigned by IBM Model 1 (Brown et al., 1993) of generating the words \( f_l \) given the words \( e_A \), with one important difference—the length \( m \) of the foreign sentence is often not modeled; so the term \( p_l(|(f_l| = m | A) \) is set to a constant and ignored. Parameterizing this length allows our model to effectively control the number of words produced at different levels of the derivation.

It is worth noting how each parameter affects the model’s behavior. The \( p_t \) distribution is a standard “translation” table, familiar from the IBM Models. The \( p_{\text{inv}} \) distribution is a “distortion” parameter, and models the likelihood of inverting non-terminals \( B \) and \( C \). This parameter can capture, for example, the high likelihood that prepositions IN and noun phrases NP often invert in Chinese due to its use of postpositions. The non-terminal length distributions \( p_l, p_m, \) and \( p_r \) model the probability of “backing off” and emitting foreign words at non-terminals when a more refined analysis cannot be found. If these parameters place high mass on 0 length word sequences, this heavily penalizes this backoff behaviour. For the TERMINAL rule, the length distribution \( p_{w} \) parameterizes the number of words produced for a particular English word \( e \), functioning similarly to the “fertilities” employed by IBM Models 3 and 4 (Brown et al., 1993). This allows us
to model, for example, the tendency of English determiners the and a to translate to nothing in the Chinese, and of English names to align to multiple Chinese words. In general, we expect an English word to usually align to one Chinese word, and so we place a weak Dirichlet prior on on the $p_e$ distribution which puts extra mass on 1-length word sequences. This is helpful for avoiding the “garbage collection” (Moore, 2004) problem for rare words.

3.2 Exploiting Non-Terminal Labels

There are often foreign words that do not correspond well to any English word, which our model must also handle. We elected for a simple augmentation to our model to account for these words. When generating foreign word sequences $f$ at a non-terminal (i.e. via the Unary or Binary productions), we also allow for the production of foreign words from the non-terminal symbol $A$. We modify $p(f_j \mid e_A)$ from the previous section to allow production of $f_j$ directly from the non-terminal$^7$ $A$:

$$p(f_j \mid e_A) = p_{nt} \cdot p(f_j \mid A)$$

$$+ (1 - p_{nt}) \cdot \sum_{i \in e_n} \frac{1}{|e_n|} p_i(f_j \mid e_i)$$

where $p_{nt}$ is a global binomial parameter which controls how often such alignments are made.

This necessitates the inclusion of parameters like $p_t(的 \mid NP)$ into our translation table. Generally, these parameters do not contain much information, but rather function like a traditional NULL rooted at some position in the tree. However, in some cases, the particular annotation used by the Penn Treebank (Marcus et al., 1993) (and hence most parsers) allows for some interesting parameters to be learned. For example, we found that our aligner often matched the Chinese word 了, which marks the past tense (among other things), to the preterminals VBD and VBN, which denote the English simple past and perfect tense. Additionally, Chinese measure words like 个 and 米 often align to the CD (numeral) preterminal. These generalizations can be quite useful – where a particular number might predict a measure word quite poorly, the generalization that measure words co-occur with the CD tag is very robust.

3.3 Membership in ITG

The generative process which describes our model contains a class of grammars larger than the computationally efficient class of ITG grammars. Fortunately, the parameterization described above not only reduces the number of parameters to a manageable level, but also introduces independence assumptions which permit synchronous binarization (Zhang et al., 2006) of our grammar. Any SCFG that can be synchronously binarized is an ITG, meaning that our parameterization permits efficient inference algorithms which we will make use of in the next section. Although several binarizations are possible, we give one such binarization and its associated probabilities in Table 2.

3.4 Robustness to Syntactic Divergence

Generally speaking, ITG grammars have proven more useful without the monolingual syntactic constraints imposed by a target parse tree. When derivations are restricted to respect a target-side parse tree, many desirable alignments are ruled out when the syntax of the two languages diverges, and alignment quality drops precipitously (Zhang and Gildea, 2004), though attempts have been made to address this issue (Gildea, 2003).

Our model is designed to degrade gracefully in the case of syntactic divergence. Because it can produce foreign words at any level of the derivation, our model can effectively back off to a variant of Model 1 in the case where an ITG derivation that both respects the target parse tree and the desired word-level alignments cannot be found.

For example, consider the sentence pair fragment in Figure 3. It is not possible to produce an ITG derivation of this fragment that both respects the English tree and also aligns all foreign words to their obvious English counterparts. Our model handles this case by attaching the troublesome 明天 at the uppermost VP. This analysis captures 3 of the 4 word-level correspondences, and also permits extraction of abstract rules like ($S \rightarrow NP \ VP ; NP \ VP$) and ($NP \rightarrow the \ NN ; NN$).

Unfortunately, this analysis leaves the English word tomorrow with an empty foreign span, permitting extraction of the incorrect translation (VP $\rightarrow$ announced tomorrow; 公布), among others. Our
There are possibilities for improving our model’s treatment of syntactic divergence. One option is to allow the model to select trees which are more consistent with the alignment (Burkett et al., 2010), which our model can do since it permits efficient inference over forests. The second is to modify the generative process slightly, perhaps by including the “clone” operator of Gildea (2003).

### 4 Learning and Inference

#### 4.1 Parameter Estimation

The parameters of our model can be efficiently estimated in an unsupervised fashion using the Expectation-Maximization (EM) algorithm. The E-step requires the computation of expected counts under our model for each multinomial parameter. We omit the details of obtaining expected counts for each distribution, since they can be obtained using simple arithmetic from a single quantity, namely, the expected count of a particular instantiation of a synchronous rule \( r \). This expectation is a standard quantity that can be computed in \( O(n^6) \) time using the bitext Inside-Outside dynamic program (Wu, 1997).

#### 4.2 Dynamic Program Pruning

While our model permits \( O(n^6) \) inference over a forest of English trees, inference over a full forest would be very slow, and so we fix a single \( n \)-ary English tree obtained from a monolingual parser. However, it is worth noting that the English side of the ITG derivation is not completely fixed. Where our English trees are more than binary branching, we permit any binarization in our dynamic program.

For efficiency, we also ruled out span alignments that are extremely lopsided, for example, a 1-word English span aligned to a 20-word foreign span. Specifically, we pruned any span alignment in which one side is more than 5 times larger than the other.

Finally, we employ pruning based on high-precision alignments from simpler models (Cherry and Lin, 2007; Haghhighi et al., 2009). We compute word-to-word alignments by finding all word pairs which have a posterior of at least 0.7 according to both forward and reverse IBM Model 1 parameters, and prune any span pairs which invalidate more than 3 of these alignments. In total, this pruning re-
Table 3: Alignment quality results for our syntactic aligner and our GIZA++ baseline.

<table>
<thead>
<tr>
<th></th>
<th>P</th>
<th>R</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Syntactic Align</td>
<td>50.9</td>
<td>83.0</td>
<td>63.1</td>
</tr>
<tr>
<td>GIZA++</td>
<td>56.1</td>
<td>67.3</td>
<td>61.2</td>
</tr>
<tr>
<td>Rule</td>
<td>P</td>
<td>R</td>
<td>F1</td>
</tr>
<tr>
<td>Syntactic Align</td>
<td>39.6</td>
<td>40.3</td>
<td>39.9</td>
</tr>
<tr>
<td>GIZA++</td>
<td>46.2</td>
<td>34.7</td>
<td>39.6</td>
</tr>
</tbody>
</table>

We reduced computation from approximately 1.5 seconds per sentence to about 0.3 seconds per sentence, a speed-up of a factor of 5.

4.3 Decoding

Given a trained model, we extract a tree-to-string alignment as follows: we compute, for each node in the English tree, the posterior probability of a particular foreign span assignment using the same dynamic program needed for EM. We then compute the set of span assignments which maximizes the sum of these posteriors, constrained such that the foreign span assignments nest in the obvious way. This algorithm is a natural synchronous generalization of the monolingual Maximum Constituents Parse algorithm of Goodman (1996).

5 Experiments

5.1 Alignment Quality

We first evaluated our alignments against gold standard annotations. Our training data consisted of the 2261 manually aligned and translated sentences of the Chinese Treebank (Bies et al., 2007) and approximately half a million unlabeled sentences of parallel Chinese-English newswire. The unlabeled data was subsampled (Li et al., 2009) from a larger corpus by selecting sentences which have good tune and test set coverage, and limited to sentences of length at most 40. We parsed the English side of the training data with the Berkeley parser. For our baseline alignments, we used GIZA++, trained in the standard way. We used the grow-diag-final alignment heuristic, as we found it outperformed union in early experiments.

We trained our unsupervised syntactic aligner on the concatenation of the labelled and unlabelled data. As is standard in unsupervised alignment models, we initialized the translation parameters $p_t$ by first training 5 iterations of IBM Model 1 using the joint training algorithm of Liang et al. (2006), and then trained our model for 5 EM iterations. We extracted syntactic rules using a re-implementation of the Galley et al. (2006) algorithm from both our syntactic alignments and the GIZA++ alignments. We handle null-aligned words by extracting every consistent derivation, and extracted composed rules consisting of at most 3 minimal rules.

We evaluate our alignments against the gold standard in two ways. We calculated Span F-score, which compares the set of extractable nodes paired with a foreign span, and Rule F-score (Fossum et al., 2008) over minimal rules. The results are shown in Table 3. By both measures, our syntactic aligner effectively trades recall for precision when compared to our baseline, slightly increasing overall F-score.

5.2 Translation Quality

For our translation system, we used a re-implementation of the syntactic system of Galley et al. (2006). For the translation rules extracted from our data, we computed standard features based on relative frequency counts, and tuned their weights using MERT (Och, 2003). We also included a language model feature, using a 5-gram language model trained on 220 million words of English text using the SRILM Toolkit (Stolcke, 2002).

For tuning and test data, we used a subset of the NIST MT04 and MT05 with sentences of length at most 40. We used the first 1000 sentences of this set for tuning and the remaining 642 sentences as test data. We used the decoder described in DeNero et al. (2009) during both tuning and testing.

We provide final tune and test set results in Table 4. Our alignments produce a 1.0 BLEU improvement over the baseline. Our reported syntactic results were obtained when rules were thresholded by count; we discuss this in the next section.

5.3 Analysis

As discussed in Section 3.4, our aligner is designed to extract many rules, which risks inadvertently extracting low-quality rules. To quantify this, we first examined the number of rules extracted by our aligner as compared with GIZA++. After relativiz-
Table 4: Final tune and test set results for our grammars extracted using the baseline GIZA++ alignments and our syntactic aligner. When we filter the GIZA++ grammars with the same count thresholds used for our aligner (“high count”), BLEU score drops substantially.

<table>
<thead>
<tr>
<th></th>
<th>Tune</th>
<th>Test</th>
</tr>
</thead>
<tbody>
<tr>
<td>Syntactic Alignment</td>
<td>29.78</td>
<td>29.83</td>
</tr>
<tr>
<td>GIZA++</td>
<td>28.76</td>
<td>28.84</td>
</tr>
<tr>
<td>GIZA++ high count</td>
<td>25.51</td>
<td>25.38</td>
</tr>
</tbody>
</table>

Of course, high-count rules are not guaranteed to be high quality. To verify that frequent rules were better for translation, we experimented with various methods of thresholding to remove rules with low count extracted from using aligner. We found in early development that removing low-count rules improved translation performance substantially. In particular, we settled on the following scheme: we kept all rules with a single foreign terminal on the right-hand side. For entirely lexical (gapless) rules, we kept all rules occurring at least 3 times. For unlexicalized rules, we kept all rules occurring at least 20 times per gap. For rules which mixed gaps and lexical items, we kept all rules occurring at least 10 times per gap. This left us with a grammar about 600 000 rules, the same grammar which gave us our final results reported in Table 4.

In contrast to our syntactic aligner, rules extracted using GIZA++ could not be so aggressively pruned. When pruned using the same count thresholds, accuracy dropped by more than 3.0 BLEU on the tune set, and similarly on the test set (see Table 4). To obtain the accuracy shown in our final results (our best results with GIZA++), we had to adjust the count threshold to include all lexicalized rules, all unlexicalized rules, and mixed rules occurring at least twice per gap. With these count thresholds, the GIZA++ grammar contained about 580 000 rules, roughly the same number as our syntactic grammar.

We also manually searched the grammars for rules that had high count in the syntactically-

Figure 4: Number of extracted translation rules with a particular count. Grammars extracted from our syntactic aligner produce not only more low-count rules, but also more high-count rules than GIZA++.

Figure 6: Number of extracted translation rules with a particular count. Grammars extracted from our syntactic aligner produce not only more low-count rules, but also more high-count rules than GIZA++.

6 Conclusion

We have described a syntactic alignment model which explicitly aligns nodes of a syntactic parse in one language to spans in another, making it suitable for use in many syntactic translation systems. Our model is unsupervised and can be efficiently trained with a straightforward application of EM. We have demonstrated that our model can accurately capture many syntactic correspondences, and is robust in the face of syntactic divergence between language pairs. Our aligner permits the extraction of more reliable, high-count rules when compared to a standard word-alignment baseline. These high-count rules also produce improvements in BLEU score.

Acknowledgements

This project is funded in part by the NSF under grant 0643742; by BBN under DARPA contract HR0011-06-C-0022; and an NSERC Postgraduate Fellowship. The authors would like to thank Michael Auli for his input.
References


Liang Huang, Kevin Knight, and Aravind Joshi. 2006. A syntax-directed translator with extended domain of locality. In Proceedings of CHSLP.


