Learning to Translate with Source and Target Syntax

David Chiang
USC Information Sciences Institute
4676 Admiralty Way, Suite 1001
Marina del Rey, CA 90292 USA
chiang@isi.edu

Abstract

Statistical translation models that try to capture the recursive structure of language have been widely adopted over the last few years. These models make use of varying amounts of information from linguistic theory: some use none at all, some use information about the grammar of the target language, some use information about the grammar of the source language. But progress has been slower on translation models that are able to learn the relationship between the grammars of both the source and target language. We discuss the reasons why this has been a challenge, review existing attempts to meet this challenge, and show how some old and new ideas can be combined into a simple approach that uses both source and target syntax for significant improvements in translation accuracy.

1 Introduction

Statistical translation models that use synchronous context-free grammars (SCFGs) or related formalisms to try to capture the recursive structure of language have been widely adopted over the last few years. The simplest of these (Chiang, 2005) make no use of information from syntactic theories or syntactic annotations, whereas others have successfully incorporated syntactic information on the target side (Galley et al., 2004; Galley et al., 2006) or the source side (Liu et al., 2006; Huang et al., 2006). The next obvious step is toward models that make full use of syntactic information on both sides. But the natural generalization to this setting has been found to underperform phrase-based models (Liu et al., 2009; Ambati and Lavie, 2008), and researchers have begun to explore solutions (Zhang et al., 2008; Liu et al., 2009).

In this paper, we explore the reasons why tree-to-tree translation has been challenging, and how source syntax and target syntax might be used together. Drawing on previous successful attempts to relax syntactic constraints during grammar extraction in various ways (Zhang et al., 2008; Liu et al., 2009; Zollmann and Venugopal, 2006), we compare several methods for extracting a synchronous grammar from tree-to-tree data. One confounding factor in such a comparison is that some methods generate many new syntactic categories, making it more difficult to satisfy syntactic constraints at decoding time. We therefore propose to move these constraints from the formalism into the model, implemented as features in the hierarchical phrase-based model Hiero (Chiang, 2005). This augmented model is able to learn from data whether to rely on syntax or not, or to revert back to monotone phrase-based translation.

In experiments on Chinese-English and Arabic-English translation, we find that when both source and target syntax are made available to the model in an unobtrusive way, the model chooses to build structures that are more syntactically well-formed and yield significantly better translations than a nonsyntactic hierarchical phrase-based model.

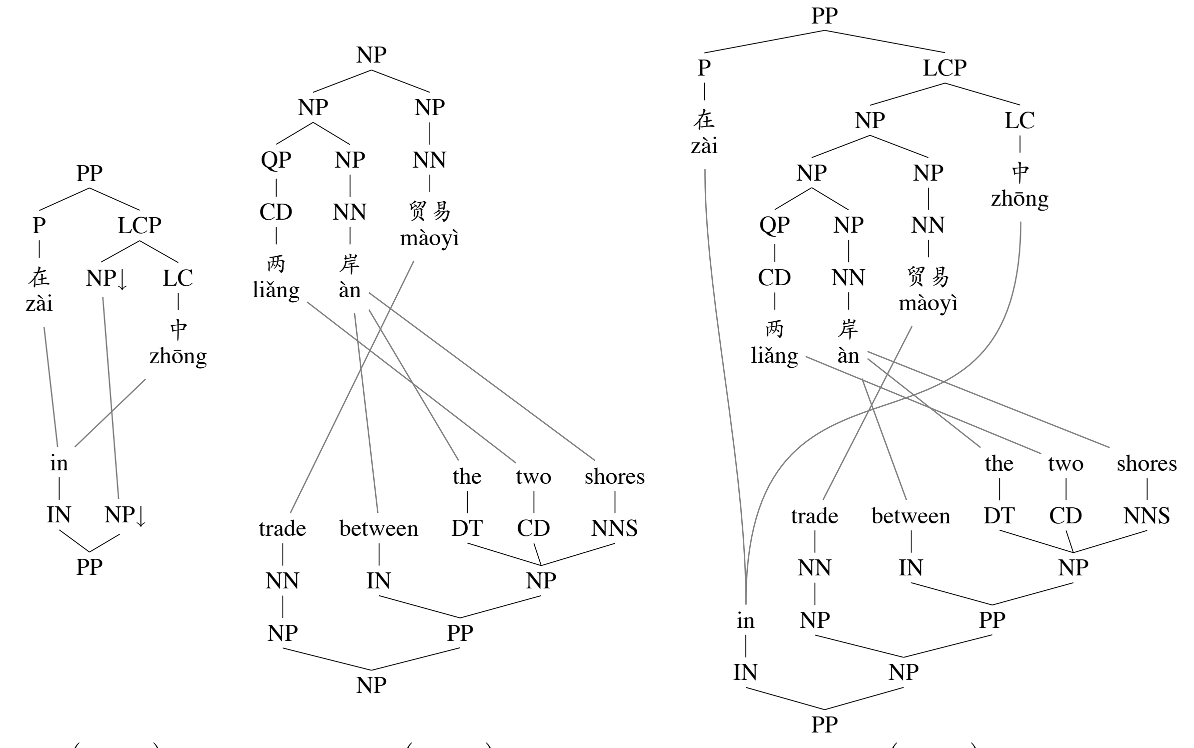
2 Grammar extraction

A *synchronous tree-substitution grammar* (STSG) is a set of rules or *elementary tree pairs* (γ, α), where:

- γ is a tree whose interior labels are source-language nonterminal symbols and whose frontier labels are source-language nonterminal symbols or terminal symbols (words). The nonterminal-labeled frontier nodes are called *substitution nodes*, conventionally marked with an arrow ($\downarrow$).

- α is a tree of the same form except with

Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, pages 1443–1452,
Uppsala, Sweden, 11-16 July 2010. ©2010 Association for Computational Linguistics



(γ_1, α_1) (γ_2, α_2) (γ_3, α_3)

Figure 1: Synchronous tree substitution. Rule (γ_2, α_2) is substituted into rule (γ_1, α_1) to yield (γ_3, α_3).

target-language instead of source-language symbols.

- The substitution nodes of γ are aligned bijectively with those of α.

- The terminal-labeled frontier nodes of γ are aligned (many-to-many) with those of α.

In the *substitution* operation, an aligned pair of substitution nodes is rewritten with an elementary tree pair. The labels of the substitution nodes must match the root labels of the elementary trees with which they are rewritten (but we will relax this constraint below). See Figure 1 for examples of elementary tree pairs and substitution.

2.1 Exact tree-to-tree extraction

The use of STSGs for translation was proposed in the Data-Oriented Parsing literature (Poutsma, 2000; Hearne and Way, 2003) and by Eisner (2003). Both of these proposals are more ambitious about handling spurious ambiguity than approaches derived from phrase-based translation usually have been (the former uses random sampling to sum over equivalent derivations during decoding, and the latter uses dynamic programming to sum over equivalent derivations during training). If we take a more typical approach, which generalizes that of Galley et al. (2004; 2006) and is similar to Stat-XFER (Lavie et al., 2008), we obtain the following grammar extraction method, which we call *exact* tree-to-tree extraction.

Given a pair of source- and target-language parse trees with a word alignment between their leaves, identify all the *phrase pairs* $(\bar{f}, \bar{e})$, i.e., those substring pairs that respect the word align-

	human	automatic
string-to-string	198,445	142,820
max nested	78,361	64,578
tree-to-string	60,939 (78%)	48,235 (75%)
string-to-tree	59,274 (76%)	46,548 (72%)
tree-to-tree	53,084 (68%)	39,049 (60%)

Table 1: Analysis of phrases extracted from Chinese-English newswire data with human and automatic word alignments and parses. As tree constraints are added, the number of phrase pairs drops. Errors in automatic annotations also decrease the number of phrase pairs. Percentages are relative to the maximum number of nested phrase pairs.

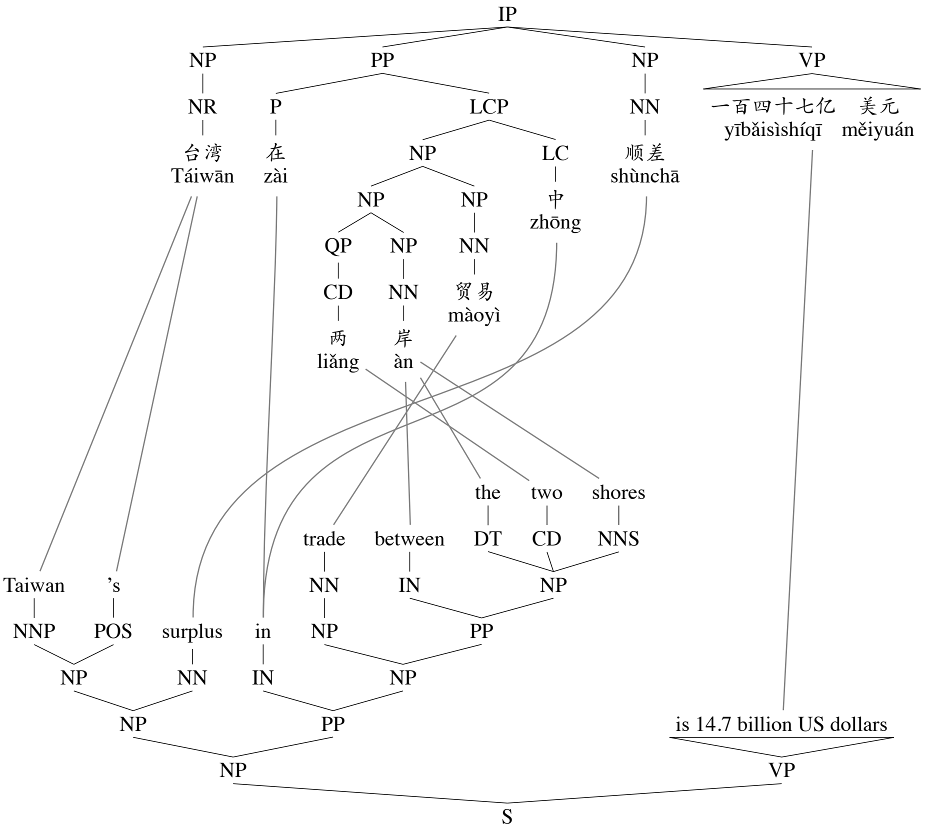


Figure 2: Example Chinese-English sentence pair with human-annotated parse trees and word alignments.

ment in the sense that at least one word in $\bar{f}$ is aligned to a word in $\bar{e}$, and no word in $\bar{f}$ is aligned to a word outside of $\bar{e}$, or vice versa. Then the extracted grammar is the smallest STSG G satisfying:

- If (γ, α) is a pair of subtrees of a training example and the frontiers of γ and α form a phrase pair, then (γ, α) is a rule in G.
- If $(\gamma_2, \alpha_2) \in G$, $(\gamma_3, \alpha_3) \in G$, and (γ_1, α_1) is an elementary tree pair such that substituting (γ_2, α_2) into (γ_1, α_1) results in (γ_3, α_3), then (γ_1, α_1) is a rule in G.

For example, consider the training example in Figure 2, from which the elementary tree pairs shown in Figure 1 can be extracted. The elementary tree pairs (γ_2, α_2) and (γ_3, α_3) are rules in G because their yields are phrase pairs, and (γ_1, α_1) results from subtracting (γ_2, α_2) from (γ_3, α_3).

2.2 Fuzzy tree-to-tree extraction

Exact tree-to-tree translation requires that translation rules deal with syntactic constituents on both the source and target side, which reduces the number of eligible phrases. Table 1 shows an analysis of phrases extracted from human word-aligned and parsed data and automatically word-aligned and parsed data.[1] The first line shows the number of phrase-pair occurrences that are extracted in the absence of syntactic constraints,[2] and the second line shows the maximum number of *nested* phrase-pair occurrences, which is the most that exact syntax-based extraction can achieve. Whereas tree-to-string extraction and string-to-tree extraction permit 70–80% of the maximum possible number of phrase pairs, tree-to-tree extraction only permits 60–70%.

Why does this happen? We can see that moving from human annotations to automatic annotations decreases not only the absolute number of phrase pairs, but the percentage of phrases that pass the syntactic filters. Wellington et al. (2006), in a more systematic study, find that, of sentences where the tree-to-tree constraint blocks rule extraction, the majority are due to parser errors. To address this problem, Liu et al. (2009) extract rules from pairs

[1] The first 2000 sentences from the GALE Phase 4 Chinese Parallel Word Alignment and Tagging Part 1 (LDC2009E83) and the Chinese News Translation Text Part 1 (LDC2005T06), respectively.

[2] Only counting phrases that have no unaligned words at their endpoints.

of packed forests instead of pairs of trees. Since a packed forest is much more likely to include the correct tree, it is less likely that parser errors will cause good rules to be filtered out.

However, even on human-annotated data, tree-to-tree extraction misses many rules, and many such rules would seem to be useful. For example, in Figure 2, the whole English phrase "Taiwan's...shores" is an NP, but its Chinese counterpart is not a constituent. Furthermore, neither "surplus...shores" nor its Chinese counterpart are constituents. But both rules are arguably useful for translation. Wellington et al. therefore argue that in order to extract as many rules as possible, a more powerful formalism than synchronous CFG/TSG is required: for example, generalized multitext grammar (Melamed et al., 2004), which is equivalent to synchronous set-local multicomponent CFG/TSG (Weir, 1988).

But the problem illustrated in Figure 2 does not reflect a very deep fact about syntax or cross-lingual divergences, but rather choices in annotation style that interact badly with the exact tree-to-tree extraction heuristic. On the Chinese side, the IP is too flat (because 台湾/Táiwān has been analyzed as a topic), whereas the more articulated structure

(1) [$_{\text{NP}}$ Táiwān [$_{\text{NP}}$ [$_{\text{PP}}$ zaì ...] shùnchā]]

would also be quite reasonable. On the English side, the high attachment of the PP disagrees with the corresponding Chinese structure, but low attachment also seems reasonable:

(2) [$_{\text{NP}}$ [$_{\text{NP}}$ Taiwan's] [$_{\text{NP}}$ surplus in trade...]]

Thus even in the gold-standard parse trees, phrase structure can be underspecified (like the flat IP above) or uncertain (like the PP attachment above).

For this reason, some approaches work with a more flexible notion of constituency. *Synchronous tree-sequence–substitution grammar (STSSG)* allows either side of a rule to comprise a sequence of trees instead of a single tree (Zhang et al., 2008). In the substitution operation, a sequence of sister substitution nodes is rewritten with a tree sequence of equal length (see Figure 3a). This extra flexibility effectively makes the analysis (1) available to us.

Any STSSG can be converted into an equivalent STSG via the creation of virtual nodes (see Figure 3b): for every elementary tree sequence with roots $X_1, \ldots, X_n$, create a new root node with a complex label $X_1 * \cdots * X_n$ immediately dominating the old roots, and replace every sequence of substitution sites $X_1, \ldots, X_n$ with a single substitution site $X_1 * \cdots * X_n$. This is essentially what *syntax-augmented MT (SAMT)* does, in the string-to-tree setting (Zollmann and Venugopal, 2006). In addition, SAMT drops the requirement that the X_i are sisters, and uses categories $X\,/\,Y$ (an X missing a Y on the right) and $Y \setminus X$ (an X missing a Y on the left) in the style of categorial grammar (Bar-Hillel, 1953). Under this flexible notion of constituency, both (1) and (2) become available, albeit with more complicated categories.

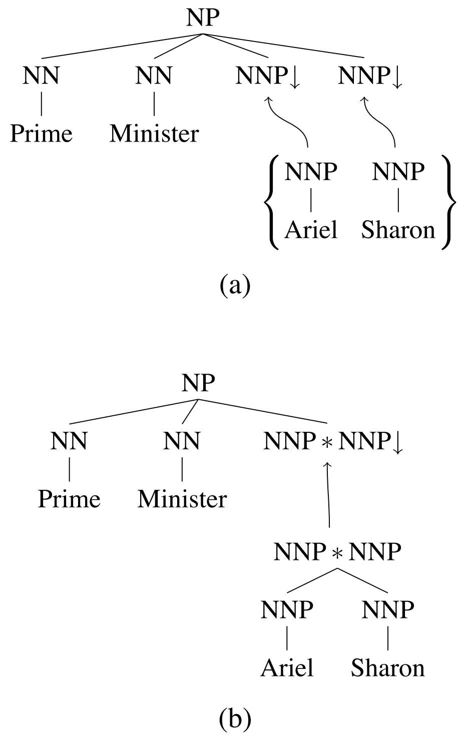


Figure 3: (a) Example tree-sequence substitution grammar and (b) its equivalent SAMT-style tree-substitution grammar.

Both STSSG and SAMT are examples of what we might call *fuzzy tree-to-tree extraction*. We follow this approach here as well: as in STSSG, we work on tree-to-tree data, and we use the complex categories of SAMT. Moreover, we allow the product categories $X_1 * \cdots * X_n$ to be of any length n, and we allow the slash categories to take any number of arguments on either side. Thus every phrase can be assigned a (possibly very complex) syntactic category, so that fuzzy tree-to-tree extraction does not lose any rules relative to string-to-string extraction.

On the other hand, if several rules are extracted that differ only in their nonterminal labels, only the most-frequent rule is kept, and its count is the total count of all the rules. This means that there is a one-to-one correspondence between the rules extracted by fuzzy tree-to-tree extraction and hierarchical string-to-string extraction.

2.3 Nesting phrases

Fuzzy tree-to-tree extraction (like string-to-string extraction) generates many times more rules than exact tree-to-tree extraction does. In Figure 2, we observed that the flat structure of the Chinese IP prevented exact tree-to-tree extraction from extracting a rule containing just part of the IP, for example:

(3) [$_{\mathrm{PP}}$ zaì . . .] [$_{\mathrm{NP}}$ shùnchā]

(4) [$_{\mathrm{NP}}$ Táiwān] [$_{\mathrm{PP}}$ zaì . . .] [$_{\mathrm{NP}}$ shùnchā]

(5) [$_{\mathrm{PP}}$ zaì . . .] [$_{\mathrm{NP}}$ shùnchā] [$_{\mathrm{VP}}$ . . . měiyuán]

Fuzzy tree-to-tree extraction allows any of these to be the source side of a rule. We might think of it as effectively restructuring the trees by inserting nodes with complex labels. However, it is not possible to represent this restructuring with a single tree (see Figure 4). More formally, let us say that two phrases $w_i \cdots w_{j-1}$ and $w_{i'} \cdots w_{j'-1}$ *nest* if $i \leq i' < j' \leq j$ or $i' \leq i < j < j'$; otherwise, they *cross*. The two Chinese phrases (4) and (5) cross, and therefore cannot both be constituents in the same tree. In other words, exact tree-to-tree extraction commits to a single structural analysis but fuzzy tree-to-tree extraction pursues many restructured analyses at once.

We can strike a compromise by continuing to allow SAMT-style complex categories, but committing to a single analysis by requiring all phrases to nest. To do this, we use a simple heuristic. Iterate through all the phrase pairs $(\bar{f}, \bar{e})$ in the following order:

1. sort by whether $\bar{f}$ and $\bar{e}$ can be assigned a simple syntactic category (both, then one, then neither); if there is a tie,

2. sort by how many syntactic constituents $\bar{f}$ and $\bar{e}$ cross (low to high); if there is a tie,

3. give priority to $(\bar{f}, \bar{e})$ if neither $\bar{f}$ nor $\bar{e}$ begins or ends with punctuation; if there is a tie, finally

4. sort by the position of $\bar{f}$ in the source-side string (right to left).

For each phrase pair, accept it if it does not cross any previously accepted phrase pair; otherwise, reject it.

Because this heuristic produces a set of nesting phrases, we can represent them all in a single restructured tree. In Figure 4, this heuristic chooses structure (a) because the English-side counterpart of IP/VP has the simple category NP.

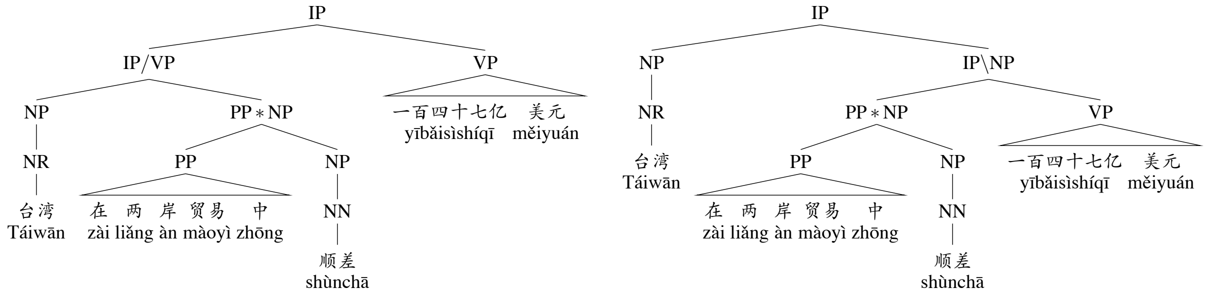
3 Decoding

In decoding, the rules extracted during training must be reassembled to form a derivation whose source side matches the input sentence. In the exact tree-to-tree approach, whenever substitution is performed, the root labels of the substituted trees must match the labels of the substitution nodes—call this the *matching constraint*. Because this constraint must be satisfied on both the source and target side, it can become difficult to generalize well from training examples to new input sentences.

Venugopal et al. (2009), in the string-to-tree setting, attempt to soften the data-fragmentation effect of the matching constraint: instead of trying to find the single derivation with the highest probability, they sum over derivations that differ only in their nonterminal labels and try to find the single derivation-class with the highest probability. Still, only derivations that satisfy the matching constraint are included in the summation.

But in some cases we may want to soften the matching constraint itself. Some syntactic categories are similar enough to be considered compatible: for example, if a rule rooted in VBD (past-tense verb) could substitute into a site labeled VBZ (present-tense verb), it might still generate correct output. This is all the more true with the addition of SAMT-style categories: for example, if a rule rooted in ADVP $*$ VP could substitute into a site labeled VP, it would very likely generate correct output.

Since we want syntactic information to help the model make good translation choices, not to rule out potentially correct choices, we can change the way the information is used during decoding: we allow any rule to substitute into any site, but let the model learn which substitutions are better than others. To do this, we add the following features to the model:



(a) (b)

Figure 4: Fuzzy tree-to-tree extraction effectively restructures the Chinese tree from Figure 2 in two ways but does not commit to either one.

- $match^f$ counts the number of substitutions where the label of the source side of the substitution site matches the root label of the source side of the rule, and $\neg match^f$ counts those where the labels do not match.

- $subst^f_{X\to Y}$ counts the number of substitutions where the label of the source side of the substitution site is X and the root label of the source side of the rule is Y.

- $match^e$, $\neg match^e$, and $subst^e_{X\to Y}$ do the same for the target side.

- $root_{X,X'}$ counts the number of rules whose root label on the source side is X and whose root label on the target side is X'.[3]

For example, in the derivation of Figure 1, the following features would fire:

$$\begin{aligned} match^f &= 1 \\ subst^f_{\mathrm{NP}\to\mathrm{NP}} &= 1 \\ match^e &= 1 \\ subst^e_{\mathrm{NP}\to\mathrm{NP}} &= 1 \\ root_{\mathrm{NP},\mathrm{NP}} &= 1 \end{aligned}$$

The decoding algorithm then operates as in hierarchical phrase-based translation. The decoder has to store in each hypothesis the source and target root labels of the partial derivation, but these labels are used for calculating feature vectors only and not for checking well-formedness of derivations. This additional state does increase the search space of the decoder, but we did not change any pruning settings.

4 Experiments

To compare the methods described above with hierarchical string-to-string translation, we ran experiments on both Chinese-English and Arabic-English translation.

4.1 Setup

The sizes of the parallel texts used are shown in Table 2. We word-aligned the Chinese-English parallel text using GIZA++ followed by link deletion (Fossum et al., 2008), and the Arabic-English parallel text using a combination of GIZA++ and LEAF (Fraser and Marcu, 2007). We parsed the source sides of both parallel texts using the Berkeley parser (Petrov et al., 2006), trained on the Chinese Treebank 6 and Arabic Treebank parts 1–3, and the English sides using a reimplementation of the Collins parser (Collins, 1997).

For string-to-string extraction, we used the same constraints as in previous work (Chiang, 2007), with differences shown in Table 2. Rules with nonterminals were extracted from a subset of the data (labeled “Core” in Table 2), and rules without nonterminals were extracted from the full parallel text. Fuzzy tree-to-tree extraction was performed using analogous constraints. For exact tree-to-tree extraction, we used simpler settings: no limit on initial phrase size or unaligned words, and a maximum of 7 frontier nodes on the source side.

All systems used the glue rule (Chiang, 2005), which allows the decoder, working bottom-up, to stop building hierarchical structure and instead concatenate partial translations without any reordering. The model attaches a weight to the glue rule so that it can learn from data whether to build shallow or rich structures, but for efficiency's sake the decoder has a hard limit, called the *distortion*

[3] Thanks to Adam Pauls for suggesting this feature class.

		Chi-Eng	Ara-Eng
Core	training words	32+38M	28+34M
	initial phrase size	10	15
	final rule size	6	6
	nonterminals	2	2
	loose source	0	∞
	loose target	0	2
Full	training words	240+260M	190+220M
	final rule size	6	6
	nonterminals	0	0
	loose source	∞	∞
	loose target	1	2

Table 2: Rule extraction settings used for experiments. "Loose source/target" is the maximum number of unaligned source/target words at the endpoints of a phrase.

limit, above which the glue rule must be used.

We trained two 5-gram language models: one on the combined English halves of the bitexts, and one on two billion words of English. These were smoothed using modified Kneser-Ney (Chen and Goodman, 1998) and stored using randomized data structures similar to those of Talbot and Brants (2008).

The base feature set for all systems was similar to the expanded set recently used for Hiero (Chiang et al., 2009), but with bigram features (source and target word) instead of trigram features (source and target word and neighboring source word). For all systems but the baselines, the features described in Section 3 were added. The systems were trained using MIRA (Crammer and Singer, 2003; Chiang et al., 2009) on a tuning set of about 3000 sentences of newswire from NIST MT evaluation data and GALE development data, disjoint from the training data. We optimized feature weights on 90% of this and held out the other 10% to determine when to stop.

4.2 Results

Table 3 shows the scores on our development sets and test sets, which are about 3000 and 2000 sentences, respectively, of newswire drawn from NIST MT evaluation data and GALE development data and disjoint from the tuning data.

For Chinese, we first tried increasing the distortion limit from 10 words to 20. This limit controls how deeply nested the tree structures built by the decoder are, and we want to see whether adding syntactic information leads to more complex structures. This change by itself led to an increase in the BLEU score. We then compared against two systems using tree-to-tree grammars. Using exact tree-to-tree extraction, we got a much smaller grammar, but decreased accuracy on all but the Chinese-English test set, where there was no significant change. But with fuzzy tree-to-tree extraction, we obtained an improvement of +0.6 on both Chinese-English sets, and +0.7/+0.8 on the Arabic-English sets.

Applying the heuristic for nesting phrases reduced the grammar sizes dramatically (by a factor of 2.4 for Chinese and 4.2 for Arabic) but, interestingly, had almost no effect on translation quality: a slight decrease in BLEU on the Arabic-English development set and no significant difference on the other sets. This suggests that the strength of fuzzy tree-to-tree extraction lies in its ability to break up flat structures and to reconcile the source and target trees with each other, rather than multiple restructurings of the training trees.

4.3 Rule usage

We then took a closer look at the behavior of the string-to-string and fuzzy tree-to-tree grammars (without the nesting heuristic). Because the rules of these grammars are in one-to-one correspondence, we can analyze the string-to-string system's derivations as though they had syntactic categories. First, Table 4 shows that the system using the tree-to-tree grammar used the glue rule much less and performed more matching substitutions. That is, in order to minimize errors on the tuning set, the model learned to build syntactically richer and more well-formed derivations.

Tables 5 and 6 show how the new syntax features affected particular substitutions. In general we see a shift towards more matching substitutions; correct placement of punctuation is particularly emphasized. Several changes appear to have to do with definiteness of NPs: on the English side, adding the syntax features encourages matching substitutions of type DT\NP-C (anarthrous NP), but discourages DT\NP-C and NN from substituting into NP-C and vice versa. For example, a translation with the rewriting NP-C → DT\NP-C begins with "24th meeting of the Standing Committee...," but the system using the fuzzy tree-to-tree grammar changes this to "The 24th meeting of the Standing Committee...."

The *root* features had a less noticeable effect on

					BLEU	
task	extraction	dist. lim.	rules	features	dev	test
Chi-Eng	string-to-string	10	440M	1k	32.7	23.4
	string-to-string	20	440M	1k	33.3	23.7
	tree-to-tree exact	20	50M	5k	32.8	23.9
	tree-to-tree fuzzy	20	440M	160k	33.9	24.3
	+ nesting	20	180M	79k	33.9	24.3
Ara-Eng	string-to-string	10	790M	1k	48.7	48.9
	tree-to-tree exact	10	38M	5k	46.6	47.5
	tree-to-tree fuzzy	10	790M	130k	49.4	49.7
	+ nesting	10	190M	66k	49.2	49.8

Table 3: On both the Chinese-English and Arabic-English translation tasks, fuzzy tree-to-tree extraction outperforms exact tree-to-tree extraction and string-to-string extraction. Brackets indicate statistically insignificant differences ($p \geq 0.05$).

rule choice; one interesting change was that the frequency of rules with Chinese root VP / IP and English root VP / S-C increased from 0.2% to 0.7%: apparently the model learned that it is good to use rules that pair Chinese and English verbs that subcategorize for sentential complements.

5 Conclusion

Though exact tree-to-tree translation tends to hamper translation quality by imposing too many constraints during both grammar extraction and decoding, we have shown that using both source and target syntax improves translation accuracy when the model is given the opportunity to learn from data how strongly to apply syntactic constraints. Indeed, we have found that the model learns on its own to choose syntactically richer and more well-formed structures, demonstrating that source- and target-side syntax can be used together profitably as long as they are not allowed to overconstrain the translation model.

Acknowledgements

Thanks to Steve DeNeefe, Adam Lopez, Jonathan May, Miles Osborne, Adam Pauls, Richard Schwartz, and the anonymous reviewers for their valuable help. This research was supported in part by DARPA contract HR0011-06-C-0022 under subcontract to BBN Technologies and DARPA contract HR0011-09-1-0028. *S. D. G.*

			frequency (%)	
task	side	kind	s-to-s	t-to-t
Chi-Eng	source	glue	25	18
		match	17	30
		mismatch	58	52
	target	glue	25	18
		match	9	23
		mismatch	66	58
Ara-Eng	source	glue	36	19
		match	17	34
		mismatch	48	47
	target	glue	36	19
		match	11	29
		mismatch	53	52

Table 4: Moving from string-to-string (s-to-s) extraction to fuzzy tree-to-tree (t-to-t) extraction decreases glue rule usage and increases the frequency of matching substitutions.

	frequency (%)	
kind	s-to-s	t-to-t
NP → NP	16.0	20.7
VP → VP	3.3	5.9
NN → NP	3.1	1.3
NP → VP	2.5	0.8
NP → NN	2.0	1.4
NP → entity	1.4	1.6
NN → NN	1.1	1.0
QP → entity	1.0	1.3
VV → VP	1.0	0.7
PU → NP	0.8	1.1
VV → VP ∗ PU	0.2	1.2
PU → PU	0.1	3.8

Table 5: Comparison of frequency of source-side rewrites in Chinese-English translation between string-to-string (s-to-s) and fuzzy tree-to-tree (t-to-t) grammars. All rewrites occurring more than 1% of the time in either system are shown. The label "entity" stands for handwritten rules for named entities and numbers.

	frequency (%)	
kind	s-to-s	t-to-t
NP-C → NP-C	5.3	8.7
NN → NN	1.7	3.0
NP-C → entity	1.1	1.4
DT \ NP-C → DT \ NP-C	1.1	2.6
NN → NP-C	0.8	0.4
NP-C → VP	0.8	1.1
DT \ NP-C → NP-C	0.8	0.5
NP-C → DT \ NP-C	0.6	0.4
JJ → JJ	0.5	1.8
NP-C → NN	0.5	0.3
PP → PP	0.4	1.7
VP-C → VP-C	0.4	1.2
VP → VP	0.4	1.4
IN → IN	0.1	1.8
, → ,	0.1	1.7

Table 6: Comparison of frequency of target-side rewrites in Chinese-English translation between string-to-string (s-to-s) and fuzzy tree-to-tree (t-to-t) grammars. All rewrites occurring more than 1% of the time in either system are shown, plus a few more of interest. The label "entity" stands for handwritten rules for named entities and numbers.

References

Vamshi Ambati and Alon Lavie. 2008. Improving syntax driven translation models by re-structuring divergent and non-isomorphic parse tree structures. In *Proc. AMTA-2008 Student Research Workshop*, pages 235–244.

Yehoshua Bar-Hillel. 1953. A quasi-arithmetical notation for syntactic description. *Language*, 29(1):47–58.

Stanley F. Chen and Joshua Goodman. 1998. An empirical study of smoothing techniques for language modeling. Technical Report TR-10-98, Harvard University Center for Research in Computing Technology.

David Chiang, Wei Wang, and Kevin Knight. 2009. 11,001 new features for statistical machine translation. In *Proc. NAACL HLT 2009*, pages 218–226.

David Chiang. 2005. A hierarchical phrase-based model for statistical machine translation. In *Proc. ACL 2005*, pages 263–270.

David Chiang. 2007. Hierarchical phrase-based translation. *Computational Linguistics*, 33(2):201–228.

Michael Collins. 1997. Three generative lexicalised models for statistical parsing. In *Proc. ACL-EACL*, pages 16–23.

Koby Crammer and Yoram Singer. 2003. Ultraconservative online algorithms for multiclass problems. *Journal of Machine Learning Research*, 3:951–991.

Jason Eisner. 2003. Learning non-isomorphic tree mappings for machine translation. In *Proc. ACL 2003 Companion Volume*, pages 205–208.

Victoria Fossum, Kevin Knight, and Steven Abney. 2008. Using syntax to improve word alignment for syntax-based statistical machine translation. In *Proc. Third Workshop on Statistical Machine Translation*, pages 44–52.

Alexander Fraser and Daniel Marcu. 2007. Getting the structure right for word alignment: LEAF. In *Proc. EMNLP 2007*, pages 51–60.

Michel Galley, Mark Hopkins, Kevin Knight, and Daniel Marcu. 2004. What's in a translation rule? In *Proc. HLT-NAACL 2004*, pages 273–280.

Michel Galley, Jonathan Graehl, Kevin Knight, Daniel Marcu, Steve DeNeefe, Wei Wang, and Ignacio Thayer. 2006. Scalable inference and training of context-rich syntactic translation models. In *Proc. COLING-ACL 2006*, pages 961–968.

Mary Hearne and Andy Way. 2003. Seeing the wood for the trees: Data-Oriented Translation. In *Proc. MT Summit IX*, pages 165–172.

Liang Huang, Kevin Knight, and Aravind Joshi. 2006. Statistical syntax-directed translation with extended domain of locality. In *Proc. AMTA 2006*, pages 65–73.

Alon Lavie, Alok Parlikar, and Vamshi Ambati. 2008. Syntax-driven learning of sub-sentential translation equivalents and translation rules from parsed parallel corpora. In *Proc. SSST-2*, pages 87–95.

Yang Liu, Qun Liu, and Shouxun Lin. 2006. Tree-to-string alignment template for statistical machine translation. In *Proc. COLING-ACL 2006*, pages 609–616.

Yang Liu, Yajuan Lü, and Qun Liu. 2009. Improving tree-to-tree translation with packed forests. In *Proc. ACL 2009*, pages 558–566.

I. Dan Melamed, Giorgio Satta, and Ben Wellington. 2004. Generalized multitext grammars. In *Proc. ACL 2004*, pages 661–668.

Slav Petrov, Leon Barrett, Romain Thibaux, and Dan Klein. 2006. Learning accurate, compact, and interpretable tree annotation. In *Proc. COLING-ACL 2006*, pages 433–440.

Arjen Poutsma. 2000. Data-Oriented Translation. In *Proc. COLING 2000*, pages 635–641.

David Talbot and Thorsten Brants. 2008. Randomized language models via perfect hash functions. In *Proc. ACL-08: HLT*, pages 505–513.

Ashish Venugopal, Andreas Zollmann, Noah A. Smith, and Stephan Vogel. 2009. Preference grammars: Softening syntactic constraints to improve statistical machine translation. In *Proc. NAACL HLT 2009*, pages 236–244.

David J. Weir. 1988. *Characterizing Mildly Context-Sensitive Grammar Formalisms*. Ph.D. thesis, University of Pennsylvania.

Benjamin Wellington, Sonjia Waxmonsky, and I. Dan Melamed. 2006. Empirical lower bounds on the complexity of translational equivalence. In *Proc. COLING-ACL 2006*, pages 977–984.

Min Zhang, Hongfei Jiang, Aiti Aw, Haizhou Li, Chew Lim Tan, and Sheng Li. 2008. A tree sequence alignment-based tree-to-tree translation model. In *Proc. ACL-08: HLT*, pages 559–567.

Andreas Zollmann and Ashish Venugopal. 2006. Syntax augmented machine translation via chart parsing. In *Proc. Workshop on Statistical Machine Translation*, pages 138–141.