Using Categorial Grammar to Label Translation Rules

Jonathan Weese and **Chris Callison-Burch** and **Adam Lopez**
Human Language Technology Center of Excellence
Johns Hopkins University

Abstract

Adding syntactic labels to synchronous context-free translation rules can improve performance, but labeling with phrase structure constituents, as in GHKM (Galley et al., 2004), excludes potentially useful translation rules. SAMT (Zollmann and Venugopal, 2006) introduces heuristics to create new non-constituent labels, but these heuristics introduce many complex labels and tend to add rarely-applicable rules to the translation grammar. We introduce a labeling scheme based on categorial grammar, which allows syntactic labeling of many rules with a minimal, well-motivated label set. We show that our labeling scheme performs comparably to SAMT on an Urdu–English translation task, yet the label set is an order of magnitude smaller, and translation is twice as fast.

1 Introduction

The Hiero model of Chiang (2007) popularized the usage of synchronous context-free grammars (SCFGs) for machine translation. SCFGs model translation as a process of isomorphic syntactic derivation in the source and target language. But the Hiero model is formally, not linguistically syntactic. Its derivation trees use only a single non-terminal label X, carrying no linguistic information. Consider Rule 1.

$$X \rightarrow \langle \text{ maison ; house } \rangle \tag{1}$$

We can add syntactic information to the SCFG rules by parsing the parallel training data and projecting parse tree labels onto the spans they yield and their translations. For example, if *house* was parsed as a noun, we could rewrite Rule 1 as

$$N \rightarrow \langle \text{ maison ; house } \rangle$$

But we quickly run into trouble: how should we label a rule that translates *pour l'établissement de* into *for the establishment of*? There is no phrase structure constituent that corresponds to this English fragment. This raises a model design question: what label do we assign to spans that are natural translations of each other, but have no natural labeling under a syntactic parse? One possibility would be to discard such translations from our model as implausible. However, such non-compositional translations are important in translation (Fox, 2002), and they have been repeatedly shown to improve translation performance (Koehn et al., 2003; DeNeefe et al., 2007).

Syntax-Augmented Machine Translation (SAMT; Zollmann and Venugopal, 2006) solves this problem with heuristics that create new labels from the phrase structure parse: it labels *for the establishment of* as IN+NP+IN to show that it is the concatenation of a noun phrase with a preposition on either side. While descriptive, this label is unsatisfying as a concise description of linguistic function, fitting uneasily alongside more natural labels in the phrase structure formalism. SAMT introduces many thousands of such labels, most of which are seen very few times. While these heuristics are effective (Zollmann et al., 2008), they inflate grammar size, hamper effective parameter estimation due to feature sparsity, and slow translation speed.

Our objective is to find a syntactic formalism that enables us to label most translation rules without relying on heuristics. Ideally, the label should be small in order to improve feature estimation and reduce translation time. Furthering an insight that informs SAMT, we show that *combinatory categorial grammar* (CCG) satisfies these requirements.

Under CCG, *for the establishment of* is labeled with ((S\NP)\(S\NP))/NP. This seems complex, but it describes exactly how the fragment should combine with other English words to create a complete sentence in a linguistically meaningful way. We show that CCG is a viable formalism to add syntax to SCFG-based translation.

- We introduce two models for labeling SCFG rules. One uses labels from a 1-best CCG parse tree of training data; the second uses the top labels in each cell of a CCG parse chart.
- We show that using 1-best parses performs as well as a syntactic model using phrase structure derivations.
- We show that using chart cell labels performs almost as well than SAMT, but the nonterminal label set is an order of magnitude smaller and translation is twice as fast.

2 Categorial grammar

Categorial grammar (CG) (Adjukiewicz, 1935; Bar-Hillel et al., 1964) is a grammar formalism in which words are assigned grammatical types, or *categories*. Once categories are assigned to each word of a sentence, a small set of universal combinatory rules uses them to derive a sentence-spanning syntactic structure.

Categories may be either atomic, like N, VP, S, and other familiar types, or they may be complex *function types*. A function type looks like A/B and takes an argument of type B and returns a type A. The categories A and B may themselves be either primitives or functions. A lexical item is assigned a function category when it takes an *argument* — for example, a verb may be function that needs to be combined with its subject and object, or an a adjective may be a function that takes the noun it modifies as an argument.

Lexical item	Category
and	conj
cities	NP
in	(NP\NP)/NP
own	(S\NP)/NP
properties	NP
they	NP
various	NP/NP
villages	NP

Table 1: An example lexicon, mapping words to categories.

We can combine two categories with *function application*. Formally, we write

$$X/Y \quad Y \quad \Rightarrow \quad X \tag{2}$$

to show that a function type may be combined with its argument type to produce the result type. *Backward* function application also exists, where the argument occurs to the left of the function.

Combinatory categorial grammar (CCG) is an extension of CG that includes more *combinators* (operations that can combine categories). Steedman and Baldridge (2011) give an excellent overview of CCG.

As an example, suppose we want to analyze the sentence "They own properties in various cities and villages" using the lexicon shown in Table 1. We assign categories according to the lexicon, then combine the categories using function application and other combinators to get an analysis of S for the complete sentence. Figure 1 shows the derivation.

As a practical matter, very efficient CCG parsers are available (Clark and Curran, 2007). As shown by Fowler and Penn (2010), in many cases CCG is context-free, making it an ideal fit for our problem.

2.1 Labels for phrases

Consider the German–English phrase pair *der große Mann – the tall man*. It is easily labeled as an NP and included in the translation table. By contrast, *der große– the tall*, doesn't typically correspond to a complete subtree in a phrase structure parse. Yet translating *the tall* is likely to be more useful than translating *the tall man*, since it is more general—it can be combined with any other noun translation.

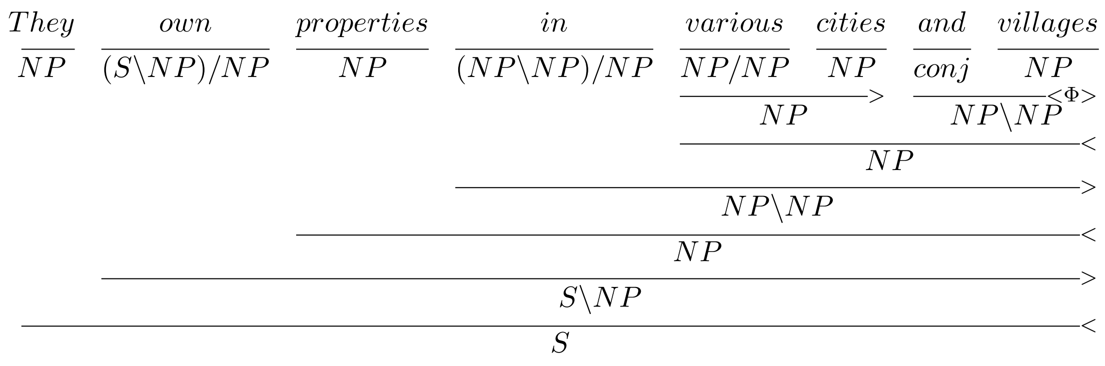


Figure 1: An example CCG derivation for the sentence "They own properties in various cities and villages" using the lexicon from Table 1. Φ indicates a conjunction operation; $>$ and $<$ are forward and backward function application, respectively.

Using CG-style labels with function types, we can assign the type (for example) NP/N to *the tall* to show that it can be combined with a noun on its right to create a complete noun phrase.[1] In general, CG can produce linguistically meaningful labels of most spans in a sentence simply as a matter of course.

2.2 Minimal, well-motivated label set

By allowing slashed categories with CG, we increase the number of labels allowed. Despite the increase in the number of labels, CG is advantageous for two reasons:

1. Our labels are derived from CCG derivations, so phrases with slashed labels represent well-motivated, linguistically-informed derivations, and the categories can be naturally combined.

2. The set of labels is small, relative to SAMT — it's restricted to the labels seen in CCG parses of the training data.

In short, using CG labels allows us to keep more linguistically-informed syntactic rules without making the set of syntactic labels too big.

3 Translation models

3.1 Extraction from parallel text

To extract SCFG rules, we start with a heuristic to extract phrases from a word-aligned sentence pair

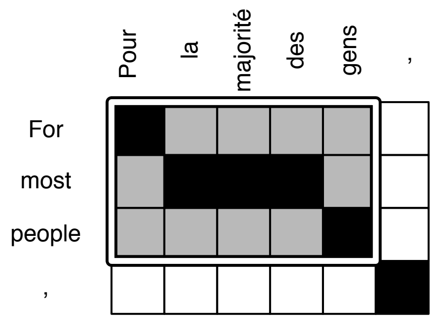


Figure 2: A word-aligned sentence pair fragment, with a box indicating a consistent phrase pair.

(Tillmann, 2003). Figure 2 shows a such a pair, with a *consistent phrase pair* inside the box. A phrase pair (f, e) is said to be consistent with the alignment if none of the words of f are aligned outside the phrase e, and vice versa – that is, there are no alignment points directly above, below, or to the sides of the box defined by f and e.

Given a consistent phrase pair, we can immediately extract the rule

$$X \rightarrow \langle f, e \rangle \tag{3}$$

as we would in a phrase-based MT system. However, whenever we find a consistent phrase pair that is a sub-phrase of another, we may extract a hierarchical rule by treating the inner phrase as a gap in the larger phrase. For example, we may extract the rule

$$X \rightarrow \langle \text{ Pour X ; For X } \rangle \tag{4}$$

from Figure 3.

[1]We could assign NP/N to the determiner *the* and N/N to the adjective *tall*, then combine those two categories using *function composition* to get a category NP/N for the two words together.

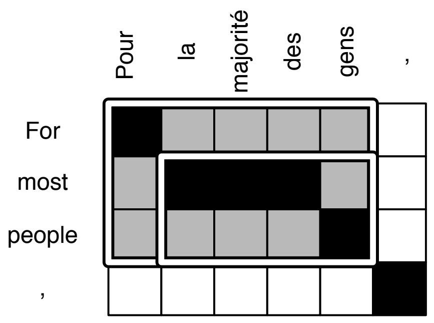


Figure 3: A consistent phrase pair with a sub-phrase that is also consistent. We may extract a hierarchical SCFG rule from this training example.

The focus of this paper is how to assign labels to the left-hand non-terminal X and to the non-terminal gaps on the right-hand side. We discuss five models below, of which two are novel CG-based labeling schemes.

3.2 Baseline: Hiero

Hiero (Chiang, 2007) uses the simplest labeling possible: there is only one non-terminal symbol, X, for all rules. Its advantage over phrase-based translation in its ability to model phrases with gaps in them, enabling phrases to reorder subphrases. However, since there's only one label, there's no way to include syntactic information in its translation rules.

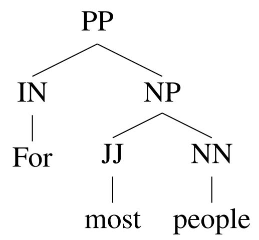
3.3 Phrase structure parse tree labeling

One first step for adding syntactic information is to get syntactic labels from a phrase structure parse tree. For each word-aligned sentence pair in our training data, we also include a parse tree of the target side.

Then we can assign syntactic labels like this: for each consistent phrase pair (representing either the left-hand non-terminal or a gap in the right hand side) we see if the target-language phrase is the exact span of some subtree of the parse tree.

If a subtree exactly spans the phrase pair, we can use the root label of that subtree to label the non-terminal symbol. If there is no such subtree, we throw away any rules derived from the phrase pair.

As an example, suppose the English side of the phrase pair in Figure 3 is analyzed as



Then we can assign syntactic labels to Rule 4 to produce

$$\text{PP} \rightarrow \langle \text{ Pour NP ; For NP } \rangle \tag{5}$$

The rules extracted by this scheme are very similar to those produced by GHKM (Galley et al., 2004), in particular resulting in the "composed rules" of Galley et al. (2006), though we use simpler heuristics for handling of unaligned words and scoring in order to bring the model in line with both Hiero and SAMT baselines. Under this scheme we throw away a lot of useful translation rules that don't translate exact syntactic constituents. For example, we can't label

$$X \rightarrow \langle \text{ Pour la majorité des ; For most } \rangle \tag{6}$$

because no single node exactly spans *For most*: the PP node includes *people*, and the NP node doesn't include *For*.

We can alleviate this problem by changing the way we get syntactic labels from parse trees.

3.4 SAMT

The Syntax-Augmented Machine Translation (SAMT) model (Zollmann and Venugopal, 2006) extracts more rules than the other syntactic model by allowing different labels for the rules. In SAMT, we try several different ways to get a label for a span, stopping the first time we can assign a label:

- As in simple phrase structure labeling, if a subtree of the parse tree exactly spans a phrase, we assign that phrase the subtree's root label.
- If a phrase can be covered by two adjacent subtrees with labels A and B, we assign their concatenation A+B.
- If a phrase spans part of a subtree labeled A that could be completed with a subtree B to its right, we assign A/B.

- If a phrase spans part of a subtree A but is missing a B to its left, we assign A\B.
- Finally, if a phrase spans three adjacent subtrees with labels A, B, and C, we assign A+B+C.

Only if all of these assignments fail do we throw away the potential translation rule.

Under SAMT, we can now label Rule 6. *For* is spanned by an IN node, and *most* is spanned by a JJ node, so we concatenate the two and label the rule as

$$\text{IN+JJ} \rightarrow \langle \text{ Pour la majorité des ; For most } \rangle \quad (7)$$

3.5 CCG 1-best derivation labeling

Our first CG model is similar to the first phrase structure parse tree model. We start with a word-aligned sentence pair, but we parse the target sentence using a CCG parser instead of a phrase structure parser.

When we extract a rule, we see if the consistent phrase pair is exactly spanned by a category generated in the 1-best CCG derivation of the target sentence. If there is such a category, we assign that category label to the non-terminal. If not, we throw away the rule.

To continue our extended example, suppose the English side of Figure 3 was analyzed by a CCG parser to produce

$$\begin{array}{ccc}
For & most & people \\
\hline (S/S)/N & \overline{N/N} & \overline{N} \\
 & \multicolumn{2}{c}{\overline{\qquad\qquad\qquad}>} \\
 & \multicolumn{2}{c}{N} \\
\multicolumn{3}{c}{\overline{\qquad\qquad\qquad\qquad\qquad}>} \\
\multicolumn{3}{c}{S/S}
\end{array}$$

Then just as in the phrase structure model, we project the syntactic labels down onto the extractable rule yielding

$$\text{S/S} \rightarrow \langle \text{ Pour N ; For N } \rangle \quad (8)$$

This does not take advantage of CCG's ability to label almost any fragment of language: the fragments with labels in any particular sentence depend on the order that categories were combined in the sentence's 1-best derivation. We can't label Rule 6, because no single category spanned *For most* in the derivation. In the next model, we increase the number of spans we can label.

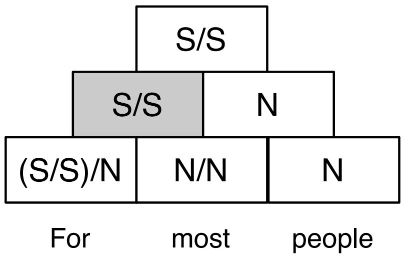


Figure 4: A portion of the parse chart for a sentence starting with "For most people . . . ." Note that the gray chart cell is not included in the 1-best derivation of this fragment in Section 3.5.

3.6 CCG parse chart labeling

For this model, we do not use the 1-best CCG derivation. Instead, when parsing the target sentence, for each cell in the parse chart, we read the most likely label according to the parsing model. This lets us assign a label for almost any span of the sentence just by reading the label from the parse chart.

For example, Figure 4 represents part of a CCG parse chart for our example fragment of "For most people." Each cell in the chart shows the most probable label for its span. The white cells of the chart are in fact present in the 1-best derivation, which means we could extract Rule 8 just as in the previous model.

But the 1-best derivation model cannot label Rule 6, and this model can. The shaded chart cell in Figure 4 holds the most likely category for the span *For most*. So we assign that label to the X:

$$S/S \rightarrow \langle \text{ Pour la majorité des ; For most } \rangle \quad (9)$$

By including labels from cells that weren't used in the 1-best derivation, we can greatly increase the number of rules we can label.

4 Comparison of resulting grammars

4.1 Effect of grammar size and label set on parsing efficiency

There are sound theoretical reasons for reducing the number of non-terminal labels in a grammar. Translation with a synchronous context-free grammar requires first parsing with the source-language projection of the grammar, followed by intersection of the target-language projection of the resulting grammar with a language model. While there are many possible algorithms for these operations, they all depend on the size of the grammar.

Consider for example the popular cube pruning algorithm of Chiang (2007), which is a simple extension of CKY. It works by first constructing a set of items of the form $\langle A, i, j \rangle$, where each item corresponds to (possibly many) partial analyses by which nonterminal A generates the sequence of words from positions i through j of the source sentence. It then produces an augmented set of items $\langle A, i, j, u, v \rangle$, in which items of the first type are augmented with left and right language model states u and v. In each pass, the number of items is linear in the number of nonterminal symbols of the grammar. This observation has motivated work in grammar transformations that reduce the size of the nonterminal set, often resulting in substantial gains in parsing or translation speed (Song et al., 2008; DeNero et al., 2009; Xiao et al., 2009).

More formally, the upper bound on parsing complexity is always at least linear in the size of the grammar constant G, where G is often loosely defined as a *grammar constant*; Iglesias et al. (2011) give a nice analysis of the most common translation algorithms and their dependence on G. Dunlop et al. (2010) provide a more fine-grained analysis of G, showing that for a variety of implementation choices that it depends on either or both the number of rules in the grammar and the number of nonterminals in the grammar. Though these are worst-case analyses, it should be clear that grammars with fewer rules or nonterminals can generally be processed more efficiently.

4.2 Number of rules and non-terminals

Table 2 shows the number of rules we can extract under various labeling schemes. The rules were extracted from an Urdu–English parallel corpus with 202,019 translations, or almost 2 million words in each language.

As we described before, moving from the phrase-structure syntactic model to the extended SAMT model vastly increases the number of translation rules — from about 7 million to 40 million rules. But the increased rule coverage comes at a cost: the non-terminal set has increased in size from 70 (the size of the set of Penn Treebank tags) to over 18,000.

Model	Rules	NTs
Hiero	4,171,473	1
Syntax	7,034,393	70
SAMT	40,744,439	18,368
CG derivations	8,042,683	505
CG parse chart	28,961,161	517

Table 2: Number of translation rules and non-terminal labels in an Urdu–English grammar under various models.

Comparing the phrase structure syntax model to the 1-best CCG derivation model, we see that the number of extracted rules increases slightly, and the grammar uses a set of about 500 non-terminal labels. This does not seem like a good trade-off; since we are extracting from the 1-best CCG derivation there really aren't many more rules we can label than with a 1-best phrase structure derivation.

But when we move to the full CCG parse chart model, we see a significant difference: when reading labels off of the entire parse chart, instead of the 1-best derivation, we don't see a significant increase in the non-terminal label set. That is, most of the labels we see in parse charts of the training data already show up in the top derivations: the complete chart doesn't contain many new labels that have never been seen before.

But by using the chart cells, we are able to assign syntactic information to many more translation rules: over 28 million rules, for a grammar about $\frac{3}{4}$ the size of SAMT's. The parse chart lets us extract many more rules without significantly increasing the size of the syntactic label set.

4.3 Sparseness of nonterminals

Examining the histograms in Figure 5 gives us a different view of the non-terminal label sets in our models. In each histogram, the horizontal axis measures label frequency in the corpus. The height of each bar shows the number of non-terminals with that frequency.

For the phrase structure syntax model, we see there are maybe 20 labels out of 70 that show up on rules less than 1000 times. All the other labels show up on very many rules.

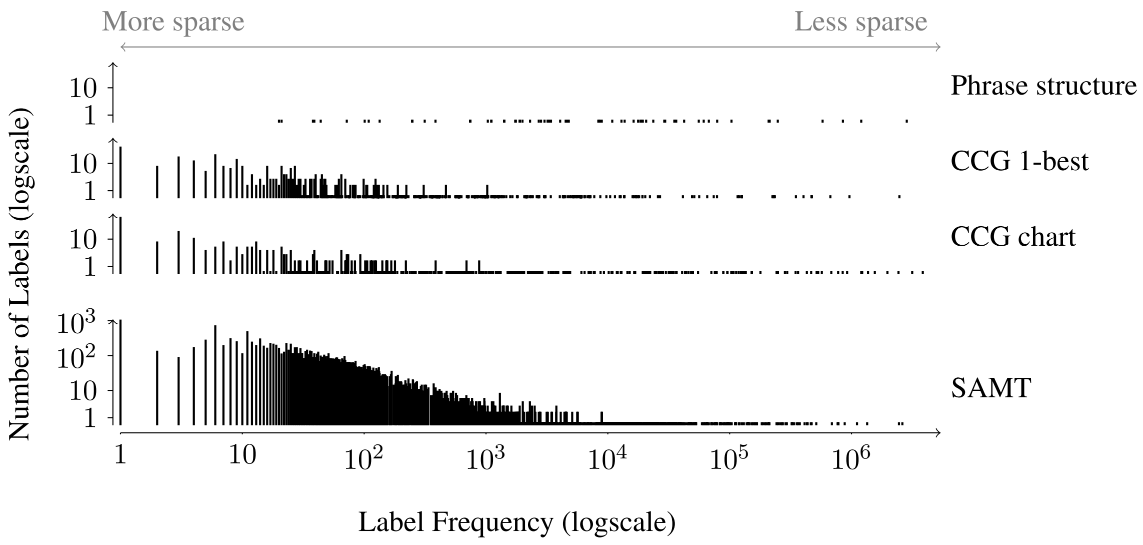


Figure 5: Histograms of label frequency for each model, illustrating the sparsity of each model.

Moving to SAMT, with its heuristically-defined labels, shows a very different story. Not only does the model have over 18,000 non-terminal labels, but thousands of them show up on fewer than 10 rules apiece. If we look at the rare label types, we see that a lot of them are improbable three way concatenations A+B+C.

The two CCG models have similar sparseness profiles. We do see some rare labels occurring only a few times in the grammars, but the number of singleton labels is an order of magnitude smaller than SAMT. Most of the CCG labels show up in the long tail of very common occurrences. Interestingly, when we move to extracting labels from parse charts rather than derivations, the number of labels increases only slightly. However, we also obtain a great deal more evidence for each observed label, making estimates more reliable.

5 Experiments

5.1 Data

We tested our models on an Urdu–English translation task, in which syntax-based systems have been quite effective (Baker et al., 2009; Zollmann et al., 2008). The training corpus was the National Institute of Standards and Technology Open Machine Translation 2009 Evaluation (NIST Open MT09). According to the MT09 Constrained Training Conditions Resources list[2] this data includes NIST Open MT08 Urdu Resources[3] and the NIST Open MT08 Current Test Set Urdu–English[4]. This gives us 202,019 parallel translations, for approximately 2 million words of training data.

5.2 Experimental design

We used the scripts included with the Moses MT toolkit (Koehn et al., 2007) to tokenize and normalize the English data. We used a tokenizer and normalizer developed at the SCALE 2009 workshop (Baker et al., 2009) to preprocess the Urdu data. We used GIZA++ (Och and Ney, 2000) to perform word alignments.

For phrase structure parses of the English data, we used the Berkeley parser (Petrov and Klein, 2007). For CCG parses, and for reading labels out of a parse chart, we used the C&C parser (Clark and Curran, 2007).

After aligning and parsing the training data, we used the Thrax grammar extractor (Weese et al., 2011) to extract all of the translation grammars.

We used the same feature set in all the translation grammars. This includes, for each rule $C \rightarrow \langle f; e \rangle$, relative-frequency estimates of the probabil-

[2]`http://www.itl.nist.gov/iad/mig/tests/mt/2009/MT09_ConstrainedResources.pdf`

[3]LDC2009E12

[4]LDC2009E11

Model	BLEU	sec./sent.
Hiero	25.67 (0.9781)	0.05
Syntax	27.06 (0.9703)	3.04
SAMT	28.06 (0.9714)	63.48
CCG derivations	27.3 (0.9770)	5.24
CCG parse chart	27.64 (0.9673)	33.6

Table 3: Results of translation experiments on Urdu–English. Higher BLEU scores are better. BLEU's brevity penalty is reported in parentheses.

ities $p(f|A)$, $p(f|e)$, $p(f|e, A)$, $p(e|A)$, $p(e|f)$, and $p(e|f, A)$.

The feature set also includes lexical weighting for rules as defined by Koehn et al. (2003) and various binary features as well as counters for the number of unaligned words in each rule.

To train the feature weights we used the Z-MERT implementation (Zaidan, 2009) of the Minimum Error-Rate Training algorithm (Och, 2003).

To decode the test sets, we used the Joshua machine translation decoder (Weese et al., 2011). The language model is a 5-gram LM trained on English GigaWord Fourth Edition.[5]

5.3 Evaluation criteria

We measure machine translation performance using the BLEU metric (Papineni et al., 2002). We also report the translation time for the test set in seconds per sentence. These results are shown in Table 3.

All of the syntactic labeling schemes show an improvement over the Hiero model. Indeed, they all fall in the range of approximately 27–28 BLEU. We can see that the 1-best derivation CCG model performs slightly better than the phrase structure model, and the CCG parse chart model performs a little better than that. SAMT has the highest BLEU score. The models with a larger number of rules perform better; this supports our assertion that we shouldn't throw away too many rules.

When it comes to translation time, the three smaller models (Hiero, phrase structure syntax, and CCG 1-best derivations) are significantly faster than the two larger ones. However, even though the CCG parse chart model is almost $\frac{3}{4}$ the size of SAMT in terms of number of rules, it doesn't take $\frac{3}{4}$ of the time. In fact, it takes only half the time of the SAMT model, thanks to the smaller rule label set.

[5]LDC2009T13

6 Discussion and Future Work

Finding an appropriate mechanism to inform phrase-based translation models and their hierarchical variants with linguistic syntax is a difficult problem that has attracted intense interest, with a variety of promising approaches including unsupervised clustering (Zollmann and Vogel, 2011), merging (Hanneman et al., 2011), and selection (Mylonakis and Sima'an, 2011) of labels derived from phrase-structure parse trees very much like those used by our baseline systems. What we find particularly attractive about CCG is that it naturally assigns linguistically-motivated labels to most spans of a sentence using a reasonably concise label set, possibility obviating the need for further refinement. Indeed, the analytical flexibility of CCG has motivated its increasing use in MT, from applications in language modeling (Birch et al., 2007; Hassan et al., 2007) to more recent proposals to incorporate it into phrase-based (Mehay, 2010) and hierarchical translation systems (Auli, 2009).

Our new model builds on these past efforts, representing a more fully instantiated model of CCG-based translation. We have shown that the label scheme allows us to keep many more translation rules than labels based on phrase structure syntax, extracting almost as many rules as the SAMT model, but keeping the label set an order of magnitude smaller, which leads to more efficient translation. This simply scratches the surface of possible uses of CCG in translation. In future work, we plan to move from a formally context-free to a formally CCG-based model of translation, implementing combinatorial rules such as application, composition, and type-raising.

Acknowledgements

Thank you to Michael Auli for providing code to inspect the full chart from the C&C parser. This research was supported in part by the NSF under grant IIS-0713448 and in part by the EuroMatrix-Plus project funded by the European Commission (7th Framework Programme). Opinions, interpretations, and conclusions are the authors' alone.

References

Kazimierz Adjukiewicz. 1935. Die syntaktische konnexität. In Storrs McCall, editor, *Polish Logic 1920–1939*, pages 207–231. Oxford University Press.

Michael Auli. 2009. CCG-based models for statistical machine translation. Ph.D. Proposal, University of Edinburgh.

Kathy Baker, Steven Bethard, Michael Bloodgood, Ralf Brown, Chris Callison-Burch, Glen Coppersmith, Bonnie Dorr, Wes Filardo, Kendall Giles, Ann Irvine, Mike Kayser, Lori Levin, Justin Marinteau, Jim Mayfield, Scott Miller, Aaron Phillips, Andrew Philpot, Christine Piatko, Lane Schwartz, and David Zajic. 2009. Semantically informed machine translation: Final report of the 2009 summer camp for advanced language exploration (scale). Technical report, Human Language Technology Center of Excellence.

Yehoshua Bar-Hillel, Chaim Gaifman, and Eliyahu Shamir. 1964. On categorial and phrase-structure grammars. In Yehoshua Bar-Hillel, editor, *Language and Information*, pages 99–115. Addison-Wesley.

Alexandra Birch, Miles Osborne, and Philipp Koehn. 2007. CCG supertags in factored statistical machine translation. In *Proc. of WMT*.

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

Stephen Clark and James R. Curran. 2007. Wide-coverage efficient statistical parsing with CCG and log-linear models. *Computational Linguistics*, 33(4).

Steve DeNeefe, Kevin Knight, Wei Wang, and Daniel Marcu. 2007. What can syntax-based MT learn from phrase-based MT? In *Proc. EMNLP*.

John DeNero, Mohit Bansal, Adam Pauls, and Dan Klein. 2009. Efficient parsing for transducer grammars. In *Proc. NAACL*, pages 227–235.

Aaron Dunlop, Nathan Bodenstab, and Brian Roark. 2010. Reducing the grammar constant: an analysis of CYK parsing efficiency. Technical report CSLU-2010-02, OHSU.

Timothy A. D. Fowler and Gerald Penn. 2010. Accurate context-free parsing with combinatory categorial grammar. In *Proc. ACL*, pages 335–344.

Heidi J. Fox. 2002. Phrasal cohesion and statistical machine translation. In *Proc. of EMNLP*.

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

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 *In ACL*, pages 961–968.

Greg Hanneman, Michelle Burroughs, and Alon Lavie. 2011. A general-purpose rule extractor for SCFG-based machine translation. In *Proc. of WMT*.

Hany Hassan, Khalil Sima'an, and Andy Way. 2007. Supertagged phrase-based statistical machine translation. In *Proc. of ACL*.

Gonzalo Iglesias, Cyril Allauzen, William Byrne, Adrià de Gispert, and Michael Riley. 2011. Hierarchical phrase-based translation representations. In *Proc. EMNLP*, pages 1373–1383.

Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proc. of HLT-NAACL*.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Frederico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proc. ACL Demonstration Session*.

Dennis Mehay. 2010. Linguistically motivated syntax for machine translation. Ph.D. Proposal, Ohio State University.

Markos Mylonakis and Khalil Sima'an. 2011. Learning hierarchical translation structure with linguistic annotations. In *Proc. of ACL-HLT*.

F. J. Och and H. Ney. 2000. Improved statistical alignment models. In *Proc. of ACL*.

Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proc. ACL*, pages 160–167.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proc. of ACL*.

Slav Petrov and Dan Klein. 2007. Improved inference for unlexicalized parsing. In *Proc. HLT-NAACL*.

Xinying Song, Shilin Ding, and Chin-Yew Lin. 2008. Better binarization for the CKY parsing. In *Proc. EMNLP*, pages 167–176.

Mark Steedman and Jason Baldridge. 2011. Combinatory categorial grammar. In Robert Borsley and Kersti Börjars, editors, *Non-Transformational Syntax*. Wiley-Blackwell.

Christoph Tillmann. 2003. A projection extension algorithm for statistical machine translation. In *Proc. of EMNLP*.

Jonathan Weese, Juri Ganitkevitch, Chris Callison-Burch, Matt Post, and Adam Lopez. 2011. Joshua 3.0: Syntax-based machine translation with the thrax grammar extractor. In *Proc. of WMT*.

Tong Xiao, Mu Li, Dongdong Zhang, Jingbo Zhu, and Ming Zhou. 2009. Better synchronous binarization for machine translation. In *Proc. EMNLP*, pages 362–370.

Omar F. Zaidan. 2009. Z-MERT: A fully configurable open source tool for minimum error rate training of machine translation systems. *The Prague Bulletin of Mathematical Linguistics*, 91(1):79–88.

Andreas Zollmann and Ashish Venugopal. 2006. Syntax augmented machine translation via chart parsing. In *Proceedings on the Workshop on Statistical Machine Translation*.

Andreas Zollmann and Stephan Vogel. 2011. A word-class approach to labeling PSCFG rules for machine translation. In *Proc. of ACL-HLT*.

Andreas Zollmann, Ashish Venugopal, Franz Och, and Jay Ponte. 2008. A systematic comparison of phrase-based, hierarchical and syntax-augmented statistical MT. In *Proc. of COLING*.