# Syntax-based Language Models for Statistical Machine Translation

**Eugene Charniak**[1], **Kevin Knight**[2] **and Kenji Yamada**[2]

[1]Department of Computer Science, Brown University
[2]Information Sciences Institute, University of Southern California

**Abstract**

We present a syntax-based language model for use in noisy-channel machine translation. In particular, a language model based upon that described in (Cha01) is combined with the syntax based translation-model described in (YK01). The resulting system was used to translate 347 sentences from Chinese to English and compared with the results of an IBM-model-4-based system, as well as that of (YK02), all trained on the same data. The translations were sorted into four groups: good/bad syntax crossed with good/bad meaning. While the total number of translations that preserved meaning were the same for (YK02) and the syntax-based system (and both higher than the IBM-model-4-based system), the syntax based system had 45% more translations that also had good syntax than did (YK02) (and approximately 70% more than IBM Model 4). The number of translations that did not preserve meaning, but at least had good grammar, also increased, though to less avail.

## 1 Introduction

Statistical machine translation (MT for short) typically takes as its basis a noisy channel model in which the target language sentence, by tradition $E$ is seen as distorted by the channel into the foreign language $F$. Thus the fundamental equation for decoding in these systems is as follows:[1]

$$\arg max_E p(E \mid F) = \arg max_E p(E) p(F \mid E) \tag{1}$$

Adopting the usual terminology we refer to the first component on the right-hand side as the language model, and the second as the translation model.

While Equation 1 leaves much unresolved concerning how the actual translation is to be performed, typically such systems are derived from the early IBM Models 1-5(BPPM93) and work at the word level. That is, the translation process is one of translating words and then rearranging them to recover the target language sentence.

Recently there has been considerable interest in MT systems based not upon words, but rather syntactic phrases (AMG00; YK01; MR01). One such system is that of (YK01), which performs the translation by assuming that during the training phase the target language (but not the source language) specifies not just the words, but rather the complete parse of the sentence. As this system has been shown to outperform IBM model 5 on the task of Chinese to English translation it seems worthwhile to take it as a starting point.

One sub-optimal part of this system is its incomplete use of the syntactic information available to it. In particular, while the decoder has been optimized for use in this environment (YK02), the language model has not. In this paper, we describe a system in which the translation model of (YK01) is "married" to the syntax-based language model of (Cha01) and show that it leads to a significant improvement in translation quality. We also show that as a "bonus" this system is able to integrate the decoding and language modeling components.

The remainder of this paper describes the base system (not using the parsing model), then gives some details of the parsing model (with particular emphasis on how it had to be changed from that in (Cha01) and on how the decoding fits here), and finally presents some evaluation results.

## 2 The Base MT System

We use a translation model as described in (YK01) and (YK02). It is a tree-to-string translation

[1]The one system of which we are aware that ostensibly rejects the noisy channel model is (ON02) However, even there, the actual system incorporates Equation 1 as a major component of the system.

model, i.e., it accepts an English parse tree as input, and produces a Chinese sentence as output. As such it operates on a more specific version of Equation 1

$$p(E \mid F) \propto \sum_{\pi} p(E, \pi) p(F \mid E, \pi) \qquad (2)$$

where $\pi$ is a parse tree of an English sentence, in this case produced by the parser described in (Col97).

If we restrict consideration to $\pi$s whose yield is the sentence $E$ (that is, $\{\pi \mid \mathcal{Y}(\pi) = E\}$) we get

$$p(E \mid F) \propto \sum_{\Pi} p(\pi) p(F \mid \pi) \qquad (3)$$

where $\Pi$ is the set of trees with yield $E$, and $\pi$ varies over this set.

Internally, the model performs three types of operations on each node of a parse tree. First, it *reorders* the child nodes, such as changing VP $\rightarrow$ VB NP PP into VP $\rightarrow$ NP PP VB. Second, it *inserts* an optional word at each node. Third, it *translates* the leaf English words into Chinese words. These operations are stochastic and their probabilities are assumed to depend only on the node, and are independent of other operations on the node, or other nodes. The probability of each operation is automatically obtained by a training algorithm, using about 780,000 English parse tree - Chinese sentence pairs. The probability of these operations $\theta(e^k_{i,j})$ are assumed to depend on the edge of the tree being modified, $e^k_{i,j}$ but independent of everything else, giving the following equation,

$$p(F \mid \pi) = \sum_{\Theta} \prod_{\theta(e^k_{i,j})} p(\theta(e^k_{i,j}) \mid e^k_{i,j}) \qquad (4)$$

where $\Theta$ varies over the possible alignments between the $F$ and $E$ and $\theta(e^k_{i,j})$ is the particular operations (in $\Theta$) for the edge $e^k_{i,j}$.

The model is further extended to incorporate phrasal translations performed at each node of the input parse tree (YK02). An English phrase covered by a node can be directly translated into a Chinese phrase without regular reorderings, insertions, and leaf-word translations.

As seen in Equation 1, the direction of the real translation task (decoding) is the reverse of the direction of the translation model. For the tree-to-string translation model, a decoder is given a sentence in Chinese, and looks for the best parse tree

S1.0-10 0.61 $\rightarrow$ NPB.0-10
S1.0-10 0.55 $\rightarrow$ S.0-10
S.0-10 0.40 $\rightarrow$ NPB.0-2 VP.2-10
NPB.0-10 7.7e-06 $\rightarrow$ NPB.3-10 NN.0-2
NPB.0-2 0.97 $\rightarrow$ NN.0-2
NN.0-2 0.00017 $\rightarrow$ ”firm”
VP.2-10 0.15 $\rightarrow$ VB.2-3 NPB.4-10 PP.3-4
PP.3-4 0.80 $\rightarrow$ ”against each other”

Figure 1: A small portion of a parse forest output by the translation model

of the English translation, according to the language model $p(E)$ and the translation model $p(F \mid E)$.

As the task of the decoder is to build a parse tree from a sentence, the decoding algorithm is quite similar to a regular parser, i.e., a parser building an English parse tree from an English sentence. As we want to parse in a exotic way in which we build an English parse tree from a Chinese sentence, we use a special grammar for decoding. We first extract CFG rules from the parsed corpus of English[2], and then extend it with the translation model. For each non-lexical English CFG rule (such as VP $\rightarrow$ VB NP PP), we supplement it with all possible reordered rules (e.g. VP $\rightarrow$ NP PP VB, and VP $\rightarrow$ PP NP VB, etc.) We also add extra rules such as “VP $\rightarrow$ VP X” and “X $\rightarrow$ *word*” for insertion operations of the model. For translation operations, we add rules such as “*englishWord* $\rightarrow$ *chineseWord*”. With this extended grammar, we can now parse a Chinese sentence to build a *decoded-tree*, from which we can extract an English parse tree by removing leaf Chinese words, and by recovering the reordered child nodes into the English order.[3]

In (YK02), a standard bottom-up parser is used for decoding. For all subtrees covering the same span, the cost of a subtree is calculated based on $p(E)$ and $p(F \mid E)$, and subtrees with a cost below a predefined threshold are discarded.

A simple trigram was used for $p(E)$ in (YK02). To experiment with more sophisticated language models, such as (Cha01), we break up the decod-

[2]The training corpus for the syntax-based model is a pair of English parse trees and Chinese sentences.

[3]The supplemented reordered rules keep a pointer to the original English CFG rule.

ing process into two steps. The first step is to build a forest using the bottom-up decoding parser as above, but using the cost only from $p(F \mid E)$. A small fragment of such a forest is shown in Figure 1. The second step is to pick the best tree from the forest with a language model, to which we now turn.

## 3 The Parser/Language-model

As noted previously, we used the syntax-based language-model described in (Cha00) and (Cha01) for both MT decoding and language modeling. As the language model used here is virtually identical to the one described in (Cha01),[4] here we give only the briefest introduction to that system, concentrating on those aspects which, while peripheral to that paper, are central to this project. We then describe the modifications needed to this system to enable it to work in an MT environment.

### 3.1 Parsing and Decoding

The statistical parser used here takes an English sentence and puts it through two parsing stages. In the first a simple, non-lexical, PCFG is used to create a large parse forest (hundreds of thousands of edges for a typical sentence). This is not a complete parse forest for the sentence, but it is sufficiently large that something very close to the correct parse is in there someplace, albeit hidden like the proverbial needle. In the second stage a much more sophisticated lexicalized PCFG is applied to the sentence. Because of its complexity it is not possible to apply the lexicalized PCFG to the entire phase-one parse forest. Rather a pruning step is inserted between the two phases. This pruning step is based upon the following equation:

$$p(e^k_{i,j} \mid w_{1,n}) = \frac{\alpha(n^k_{i,j}) p(\text{rule}(e^k_{i,j})) \prod_{n^n_{l,m} \in \text{rhs}(e^k_{i,j})} \beta(n^n_{l,m})}{p(w_{1,n})} \quad (5)$$

Here $e^k_{i,j}$ denotes a single expansion of a constituent in the parse forest: $n^k_{i,j} \rightarrow n^c_{i,a} \ldots$. Also, $\text{rule}(e^k_{i,j})$ is the PCFG rule that underlies this edge, and $\text{rhs}(e^k_{i,j})$ is the set of constituents into which the left-hand-side constituent expands.

While this equation should be familiar to those deeply acquainted with the PCFG literature, for those not so immersed we note that the left-hand side of the equation is the probability (given the sentence) of an edge $e^k_{i,j}$ which spans the words $w_{i,j}$ expanding into $n^c_{i,a} \ldots$ Thus the higher this probability, the more likely it is that this edge is used in the correct parse for the sentence. Obviously this number should be a good indicator of edges in the forest that deserve further considerations.

As our parser operates in a bottom-up manner (as do most parsers) the inside probabilities $\beta(n^k_{i,j})$ are easy to compute, but the outside probabilities $\alpha(n^k_{i,j})$ can only be computed once a parse is finished. While this is problematic in many cases, here there is no difficulty, as we do have the finished parse forest with which to work. Thus for every edge $e^k_{i,j}$ in the forest we compute $p(e^k_{i,j} \mid w_{1,n})$ and edges can be removed if it falls below an empirically set threshold. (For our experiments we used the threshold already established for parsing work, 0.00001.)

Note, however, that this probability is computed with respect to the simplified, non-lexical, PCFG — *not* with respect to the "true" lexicalized PCFG. Thus it is possible that some edge which looks implausible with respect to the first could look quite good with respect to the second. However, empirical work on parsing has shown that a threshold of 0.00001 will remove most edges, while having no discernible effect on parsing accuracy. The reason simply is that it will be the very rare lexical combination that will enable some parse to overcome a .00001 burden. We have assumed that a similar situation holds in the MT domain as well.

Once the parse forest has been pruned to a reasonable size, the second phase of the parser establishes the best parse by re-evaluating all of the remaining edges in light of the lexicalized PCFG.

Now consider the output of the translation model as shown in Figure 1. It is, in many respects, quite similar to the output of the first pass of the parser (we cover the differences in the next section). In particular, it is a forest of edges (each line in Figure

[4]In particular, it uses a trihead language-modeling scheme used in (Cha01) rather than the bi-head version used in (Cha00) for just parsing and not language modeling.

1 is a single edge) and there are a lot of them (it is common to have millions of edges for even a short sentence).

Now Equation 5 suggests a method for pruning these edges, except for one thing: it takes no cognizance of how likely these edges are given $F$, the foreign language string.

Fortunately it is not too hard to see how to fix this problem. Combining Equations 3 and 4 we get:

$$p(E \mid F) \propto \sum_{\Pi,\Theta} p(\pi) \prod_{e^k_{i,j} \in \pi} \theta(e^k_{i,j}) \qquad (6)$$

Next, the normal PCFG independence assumption allows us to break down $p(\pi)$ into the product of the probability of each of the edges in the tree:

$$p(\pi) = \prod_{e^k_{i,j} \in \pi} p(e^k_{i,j}) \qquad (7)$$

Combining Equation 6 and 7 we get:

$$p(E \mid F) \propto \sum_{\Pi,\Theta} \prod_{e^k_{i,j} \in \pi} p(e^k_{i,j}) p(\theta(e^k_{i,j}) \mid e^k_{i,j}) \qquad (8)$$

Thus we see that we want to find a parse in which the edges maximize the product of the probability of the edge $e_i$ times $p(\theta(i) \mid e_i)$, its translation-model probability.

This in turn suggests that we take as our figure of merit $\mathcal{F}(e_i)$ for pruning the forest

$$\mathcal{F}(e_i) = p(e_i \mid w_{0,n}) p(\theta(i) \mid e_i) \qquad (9)$$

where the first term is computed from equation 5 and the second comes to us from the probabilities that the translation model associates with each edge, as in Figure 1.

Thus, to use the parser as a translation decoder, we modify it to (a) expect to get the initial parse forest from the translation model, rather than level-one parsing, and (b) compute the edge figures of merit according to equation 9. Pruning the edges according to these values reduces the number of edges to a reasonable number (tens of thousands, rather than millions), so that the second pass of the parser can examine all of the remaining edges and pull out the most probable.

## 3.2 The Parser

The remaining changes to the parser fall into three categories,

- incompatibilities with the translation-model parser,
- phrasal translations, and
- non-linear word ordering.

We discuss these in order.

Those readers with nothing better to do than look up each reference at the end of the paper will have noted that the parser used in the translation model was that of Collins (Col97) while Charniak's (Cha00) was used for the decoder/language model. This creates discrepancies that should be fixed by converting to a single parser. However, for the purposes of these experiments it was easier to convert the second of these systems to deal with parse trees produced by the first. Since both are based upon the Penn tree-bank Wall-Street Journal corpus (MSM93) the differences are small, but they do creep in. These incompatibilities were duly noted and fixed, with one exception. The discrepancies in the handling of punctuation is large, and rather than fix it, we restricted consideration to sentences without punctuation. Fortunately the number of sentences in the test set with punctuation is small. However, as noted in Section 4, the current system failed to produce output on 14 sentences upon which the standard n-gram language model did fine. In eight cases the cause was punctuation.

Secondly, the translation model allows for phrasal translations — treating two or more words as a single unit. As shown in Figure 1 phrasal translations are indicated by a single non-terminal rewriting as two or more words. A completely principled response to this challenge would involve exploring the issues concerning parsing with phrases, a very interesting, but quite separate, topic. Thus we chose a remedial response. When the parsing model encounters such a rule it treats the right-hand side as a constituent to be parsed. For example, if the system encountered

NP $\rightarrow$ ”central party committee” it would find the parse

```
(NP   (JJ central) (NN party)
      (NN committee))
```

While this is not a perfect solution, it is good enough not be be a limiting factor in the current, or any immediately foreseeable, MT system.

Finally, and most importantly, it was necessary to eliminate from the parser all traces of the assumption that words and phrases come in a fixed, preestablished, order.

In parsing all constituents must indicate where they start and end, and while it is not necessary, it is convenient to assume that these points are completely ordered. For normal parsing of text, this assumption is valid, since the words to be parsed, are, in fact, completely ordered. Unfortunately, this is not the case when parsing for machine translation. In the MT context an edge such as

$$S \rightarrow NP\ VP$$

cannot "know" where the S starts, or ends, and similarly for the NP and VP, except for constraints such as the NP must start at the same location as the S. Prior to picking the English translation there is no predetermined length, for, say, the NP. Maybe the phrase from the Chinese will be translated as, say, "party committees" but then again it might be "the party committees" or a still longer phrase.

There is nothing in chart-parsing, the basic paradigm used in (Cha00) that commits one to the linearity assumption, but unless one anticipates that one is going to be using a parser far from its traditional shores, the assumption is almost inevitable, and in the event, had to be eliminated.

## 4 Evaluation

We tested on 347 previously unseen Chinese newswire sentences, with lengths ranging from 4 to 16 characters. We evaluated three systems:

- YC: the system described in this paper, i.e., the translation model of (YK01), the language model of (Cha01), and the forest-based decoder described above.
- YT: the translation model of (YK01), a standard word-trigram language model, and the decoder of (YK02).
- BT: the translation model of (BPPM93), a standard word-trigram, and the greedy decoder of (GJM$^+$01).

Figure 2 shows the evaluation results. YC far exceeds the other systems in the number of perfect translations (45 versus 26 for BT), and this is a major strength of the system. These outputs need not be touched by human posteditors. Such evaluation is often used in commercial machine translation, where human posteditors are employed.

YC also produces more grammatical output (112 total versus 37 for BT). In semantic correctness, it is only on par with the other systems (115 versus 113 for BT). It produces far fewer "garbage" outputs that are neither syntactically nor semantically correct (164 versus 223 for BN). The BLEU score varies; it tends to reward local word choice rather than global accuracy. In these experiments, YC failed to return output in some cases, as noted earlier, while BT always returned an output.

Figure 3 gives some sample outputs ("REF" means a correct human reference translation). The ability of YC to hone in on a good translation is apparent, as is its failure on some simple problems.

Also striking is how different the systems' translations are from one another. In SMT experiments, one usually sees minor variations, but in this case, radically different knowledge sources are being brought to bear. Surprisingly, if we use an oracle to vary the system (YC, YT, BT) on a sentence-by-sentence basis, we are able to obtain *77 perfect translations*. This demonstrates good future possibilities for integrating the knowledge sources we have on hand.

## 5 Conclusion

We have presented results showing that syntax-based language modeling is a promising technique for use in noisy-channel statistical machine translation. When combined with the MT-model of (YK01) we were able to increase the number of perfect translations by 45% by improving the English syntax of

| System | Perfect Translation | Syntactically Correct but Semantically Wrong | Semantically Correct Syntactically Wrong | Wrong | BLEU |
|---|---|---|---|---|---|
| YC | 45 | 67 | 70 | 164 | 0.0717 |
| YT | 31 | 19 | 87 | 209 | 0.1031 |
| BT | 26 | 11 | 87 | 223 | 0.0722 |

Figure 2: Results of evaluation on 347 previously unseen Chinese sentences. YC is the system described in this paper.

the translations. Less useful, but still relevant, is the corresponding increase in well-phrased semantic garbage. Of more interest is the differences in translation produced by the three systems considered here, as indicated by the fact that an oracle system could have increased the number of perfect translations by another 70%.

In principle syntax-based language models could be combined with a wide variety of MT systems. In particular it is not necessary that the MT-model uses syntax itself, as shown by the use of syntax-based language modeling in speech recognition (XCJ02). Nevertheless, it is certainly convenient when the MT-model is target-language parsed, as then the input to the language model is already a forest, and a significant proportion of the parser's work has already been done for it. As we expect that syntax-based MT models will become more common in the next few years, the techniques described herein should be of increasing relevance.

## 6 References

Michiko Kosaka Adam Meyers and Ralph Grishman. Chart-based transfer rule application in machine translation. In *Proceedings of COLING*, 2000.

Peter Brown, Stephan Della Pietra, Vincent Della Pietra, and Robert Mercer. Mathematics of machine translation. *Computational Linguistics*, 19(2), 1993.

Eugene Charniak. A maximum-entropy-inspired parser. In *Proceedings of the North American Chapter of the Association for Computational Linguistics 2000*, pages 132–139, New Brunswick NJ, 2000. ACL.

Eugene Charniak. Immediate-head parsing for language models. In *Proceedings of the Assocation for Computational Linguistics 2001*, pages 116–123, New Brunswick NJ, 2001. ACL.

Michael Collins. Three generative lexicalized models for statistical parsing. In *The Proceedings of the 35th Annual Meeting of the ACL*, pages 16–23, New Brunswick NJ, 1997. ACL.

Ulrich Germann, Michael Jahr, Daniel Marcu, Kevin Knight, , and Kenji Yamada. Fast decoding and optimal deciding for machine translation. In *Proceedings of the Conference of the Association for Computational Linguistics 2001*, pages 132–139, New Brunswick NJ, 2001. ACL.

Arul Menezes and Stephen D. Richardson. A best-first alignment algorithm for automatic extraction of transfer mappings from bilingual corpora. In *Proceedings of the Workshop on Data-driven Machine Translation*, pages 39–47, 2001.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330, 1993.

Franz Josef Och and Herman Ney. Discriminative traning and maximum entropy models for statistical machine translation. In *Proceedings of the Conference of the Association for Computational Linguistics 2002*, New Brunswick NJ, 2002. ACL.

Peng Xu, Ciprian Chelba, and Fred Jelinek. A study on richer syntactic dependencies for structured language modeling. In *40th Annual Meeting of the Association for Computational Linguistics*, pages 191–198, New Brunswick NJ, 2002. ACL.

Kenji Yamada and Kevin Knight. A syntax-based

| | |
|---|---|
| REF: | this is the globalization of production |
| BT: | this is a product of globalization |
| YT: | globalised production this |
| YC: | this is globalization of production |
| | |
| REF: | the importance of europe is obvious to all |
| BT: | european importance of view |
| YT: | the importance of europe is well known |
| YC: | the importance of europe is well-known |
| | |
| REF: | this is a very important guiding ideology |
| BT: | this is very important guiding |
| YT: | this is extremely important guiding thought |
| YC: | guiding ideology is very important |
| | |
| REF: | hu jintao said |
| BT: | hu jintao said |
| YT: | hu jintao said |
| YC: | mr hu said breaking |
| | |
| REF: | our utmost financial task is to guarantee the necessary public expenditures |
| BT: | fs ’s foremost task is to guarantee the need of public expenditure |
| YT: | public expenditure must ensure our finances is most important task |
| YC: | the most important task of finance is the guarantee of necessary public expenditure |
| | |
| REF: | in fact the central leadership is satisfied with mr tung chee-hwa ’s administration |
| BT: | in fact central on tung chee - wah mr patten is satisfactory |
| YT: | in fact central mr tung chee-hwa policy is satisfactory |
| YC: | the central authorities in fact are satisfied with the policy of mr tung chee-hwa |

Figure 3: Example outputs

statistical translation model. In *Proceedings of the Conference of the Association for Computational Linguistics 2001*, pages 132–139, New Brunswick NJ, 2001. ACL.

Kenji Yamada and Kevin Knight. A decoder for syntax-based statistical mt. In *Proceedings of the Conference of the Association for Computational Linguistics 2002*, New Brunswick NJ, 2002. ACL.