A Decoder for Syntax-based Statistical MT

Kenji Yamada and **Kevin Knight**
Information Sciences Institute
University of Southern California
4676 Admiralty Way, Suite 1001
Marina del Rey, CA 90292
{kyamada,knight}@isi.edu

Abstract

This paper describes a decoding algorithm for a syntax-based translation model (Yamada and Knight, 2001). The model has been extended to incorporate phrasal translations as presented here. In contrast to a conventional word-to-word statistical model, a decoder for the syntax-based model builds up an English parse tree given a sentence in a foreign language. As the model size becomes huge in a practical setting, and the decoder considers multiple syntactic structures for each word alignment, several pruning techniques are necessary. We tested our decoder in a Chinese-to-English translation system, and obtained better results than IBM Model 4. We also discuss issues concerning the relation between this decoder and a language model.

1 Introduction

A statistical machine translation system based on the noisy channel model consists of three components: a language model (LM), a translation model (TM), and a decoder. For a system which translates from a foreign language F to English E, the LM gives a prior probability $\mathrm{P}(E)$ and the TM gives a channel translation probability $\mathrm{P}(F|E)$. These models are automatically trained using monolingual (for the LM) and bilingual (for the TM) corpora. A decoder then finds the best English sentence given a foreign sentence that maximizes $\mathrm{P}(E|F)$, which also maximizes $\mathrm{P}(F|E)P(E)$ according to Bayes' rule.

A different decoder is needed for different choices of LM and TM. Since $\mathrm{P}(E)$ and $\mathrm{P}(F|E)$ are not simple probability tables but are parameterized models, a decoder must conduct a search over the space defined by the models. For the IBM models defined by a pioneering paper (Brown et al., 1993), a decoding algorithm based on a left-to-right search was described in (Berger et al., 1996). Recently (Yamada and Knight, 2001) introduced a syntax-based TM which utilized syntactic structure in the channel input, and showed that it could outperform the IBM model in alignment quality. In contrast to the IBM models, which are word-to-word models, the syntax-based model works on a syntactic parse tree, so the decoder builds up an English parse tree $\mathcal{E}$ given a sentence $\mathbf{f}$ in a foreign language. This paper describes an algorithm for such a decoder, and reports experimental results.

Other statistical machine translation systems such as (Wu, 1997) and (Alshawi et al., 2000) also produce a tree $\mathcal{E}$ given a sentence $\mathbf{f}$. Their models are based on mechanisms that generate two languages at the same time, so an English tree $\mathcal{E}$ is obtained as a subproduct of parsing $\mathbf{f}$. However, their use of the LM is not mathematically motivated, since their models do not decompose into $\mathrm{P}(F|E)$ and $P(E)$ unlike the noisy channel model.

Section 2 briefly reviews the syntax-based TM, and Section 3 describes phrasal translation as an extension. Section 4 presents the basic idea for decoding. As in other statistical machine translation systems, the decoder has to cope with a huge search

space. Section 5 describes how to prune the search space for practical decoding. Section 6 shows experimental results. Section 7 discusses LM issues, and is followed by conclusions.

2 Syntax-based TM

The syntax-based TM defined by (Yamada and Knight, 2001) assumes an English parse tree $\mathcal{E}$ as a channel input. The channel applies three kinds of stochastic operations on each node ε_i: reordering children nodes (ρ), inserting an optional extra word to the left or right of the node (ν), and translating leaf words (τ).[1] These operations are independent of each other and are conditioned on the features ($\mathcal{R}$,$\mathcal{N}$,$\mathcal{T}$) of the node. Figure 1 shows an example. The child node sequence of the top node VB is reordered from PRP-VB1-VB2 into PRP-VB2-VB1 as seen in the second tree (Reordered). An extra word *ha* is inserted at the leftmost node PRP as seen in the third tree (Inserted). The English word He under the same node is translated into a foreign word *kare* as seen in the fourth tree (Translated). After these operations, the channel emits a foreign word sentence $\mathbf{f}$ by taking the leaves of the modified tree.

Formally, the channel probability $\mathrm{P}(\mathbf{f}|\mathcal{E})$ is

$$\mathrm{P}(\mathbf{f}|\mathcal{E}) = \sum_{\boldsymbol{\theta}:\mathcal{S}(\theta(\mathcal{E}))=\mathbf{f}} \prod_{i=1}^{n} \mathrm{P}(\theta_i|\varepsilon_i)$$

$$\mathrm{P}(\theta_i|\varepsilon_i) = \begin{cases} t(\tau_i|\mathcal{T}_i)\, n(\nu_i|\mathcal{N}_i) & \text{if } \varepsilon_i \text{ is terminal} \\ r(\rho_i|\mathcal{R}_i)\, n(\nu_i|\mathcal{N}_i) & \text{otherwise} \end{cases}$$

where $\boldsymbol{\theta} = \theta_1, \theta_2, \ldots, \theta_n = \langle \rho_1, \nu_1, \tau_1 \rangle, \langle \rho_2, \nu_2, \tau_2 \rangle, \ldots, \langle \rho_n, \nu_n, \tau_n \rangle$, and $\mathcal{S}(\boldsymbol{\theta}(\mathcal{E}))$ is a sequence of leaf words of a tree transformed by $\boldsymbol{\theta}$ from $\mathcal{E}$.

The model tables $r(\rho|\mathcal{R})$, $n(\nu|\mathcal{N})$, and $t(\tau|\mathcal{T})$ are called the r-table, n-table, and t-table, respectively. These tables contain the probabilities of the channel operations (ρ, ν, τ) conditioned by the features ($\mathcal{R}$, $\mathcal{N}$, $\mathcal{T}$). In Figure 1, the r-table specifies the probability of having the second tree (Reordered) given the first tree. The n-table specifies the probability of having the third tree (Inserted) given the second tree. The t-table specifies the probability of having the fourth tree (Translated) given the third tree.

The probabilities in the model tables are automatically obtained by an EM-algorithm using pairs of $\mathcal{E}$ (channel input) and $\mathbf{f}$ (channel output) as a training corpus. Usually a bilingual corpus comes as pairs of translation sentences, so we need to parse the corpus. As we need to parse sentences on the channel input side only, many X-to-English translation systems can be developed with an English parser alone.

The conditioning features ($\mathcal{R}$,$\mathcal{N}$,$\mathcal{T}$) can be anything that is available on a tree $\mathcal{E}$, however they should be carefully selected not to cause data-sparseness problems. Also, the choice of features may affect the decoding algorithm. In our experiment, a sequence of the child node label was used for $\mathcal{R}$, a pair of the node label and the parent label was used for $\mathcal{N}$, and the identity of the English word is used for $\mathcal{T}$. For example, $r(\rho|\mathcal{R}) = \mathrm{P}(\text{PRP-VB2-VB1}|\text{PRP-VB1-VB2})$ for the top node in Figure 1. Similarly for the node PRP, $n(\nu|\mathcal{N}) = \mathrm{P}(\text{right}, \textit{ha}\,|\text{VB-PRP})$ and $t(\tau|\mathcal{T}) = \mathrm{P}(\textit{kare}|\text{he})$. More detailed examples are found in (Yamada and Knight, 2001).

3 Phrasal Translation

In (Yamada and Knight, 2001), the translation τ is a 1-to-1 lexical translation from an English word e to a foreign word f, i.e., $t(\tau|\mathcal{T}) = t(f|e)$. To allow non 1-to-1 translation, such as for idiomatic phrases or compound nouns, we extend the model as follows. First we use fertility μ as used in IBM models to allow 1-to-N mapping.

$$t(\tau|\mathcal{T}) = t(f_1 f_2 \ldots f_l|e) = \mu(l|e) \prod_{i=1}^{l} t(f_i|e)$$

For N-to-N mapping, we allow direct translation ϕ of an English phrase $e_1 e_2 \ldots e_m$ to a foreign phrase $f_1 f_2 \ldots f_l$ at non-terminal tree nodes as

$$ph(\phi|\Phi) = t(f_1 f_2 \ldots f_l|e_1 e_2 \ldots e_m) = \mu(l|e_1 e_2 \ldots e_m) \prod_{i=1}^{l} t(f_i|e_1 e_2 \ldots e_m)$$

and linearly mix this phrasal translation with the word-to-word translation, i.e.,

$$\mathrm{P}(\theta_i|\varepsilon_i) = \lambda_{\Phi_i} ph(\phi_i|\Phi_i) + (1 - \lambda_{\Phi_i}) r(\rho_i|\mathcal{R}_i)\, n(\nu_i|\mathcal{N}_i)$$

[1] The channel operations are designed to model the difference in the word order (SVO for English vs. VSO for Arabic) and case-marking schemes (word positions in English vs. case-marker particles in Japanese).

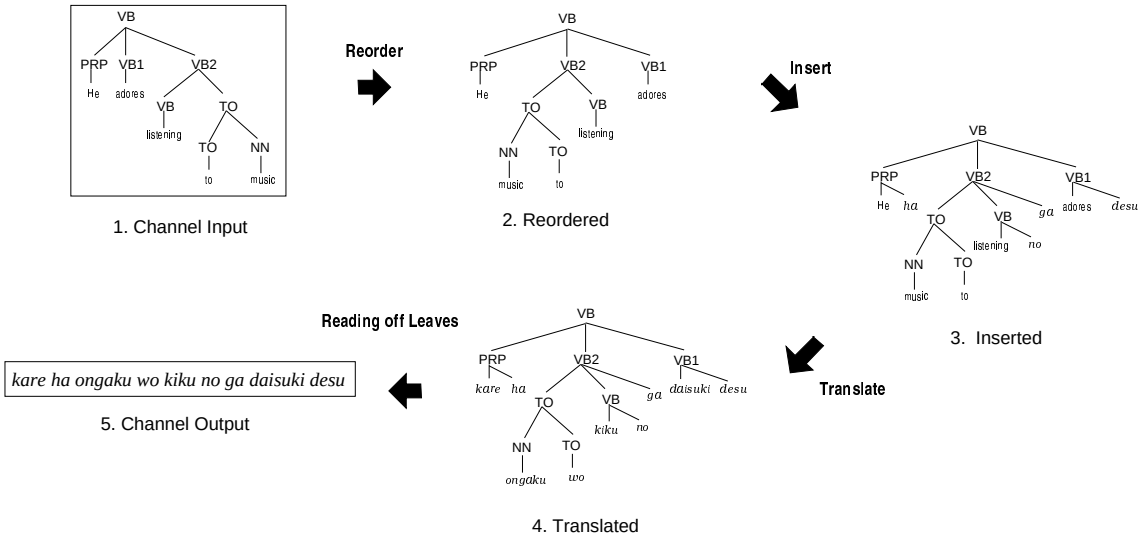

Figure 1: Channel Operations: Reorder, Insert, and Translate

if ε_i is non-terminal. In practice, the phrase lengths (l,m) are limited to reduce the model size. In our experiment (Section 5), we restricted them as $1.1m - 4 \leq l \leq 1.7m+7$, to avoid pairs of extremely different lengths. This formula was obtained by randomly sampling the length of translation pairs. See (Yamada, 2002) for details.

4 Decoding

Our statistical MT system is based on the noisy-channel model, so the decoder works in the reverse direction of the channel. Given a supposed channel output (e.g., a French or Chinese sentence), it will find the most plausible channel input (an English parse tree) based on the model parameters and the prior probability of the input.

In the syntax-based model, the decoder's task is to find the most plausible English parse tree given an observed foreign sentence. Since the task is to build a tree structure from a string of words, we can use a mechanism similar to normal parsing, which builds an English parse tree from a string of English words. Here we need to build an English parse tree from a string of foreign (e.g., French or Chinese) words.

To parse in such an exotic way, we start from an English context-free grammar obtained from the training corpus,[2] and extend the grammar to incorporate the channel operations in the translation model. For each non-lexical rule in the original English grammar (such as "VP $\rightarrow$ VB NP PP"), we supplement it with reordered rules (e.g. "VP $\rightarrow$ NP PP VB", "VP $\rightarrow$ NP VB PP ", etc.) and associate them with the original English order and the reordering probability from the r-table. Similarly, rules such as "VP $\rightarrow$ VP X" and "X $\rightarrow$ *word*" are added for extra word insertion, and they are associated with a probability from the n-table. For each lexical rule in the English grammar, we add rules such as "*englishWord* $\rightarrow$ *foreignWord*" with a probability from the t-table.

Now we can parse a string of foreign words and build up a tree, which we call a *decoded tree*. An example is shown in Figure 2. The decoded tree is built up in the foreign language word order. To obtain a tree in the English order, we apply the reverse of the reorder operation (back-reordering) using the information associated to the rule expanded by the r-table. In Figure 2, the numbers in the dashed oval near the top node shows the original english order. Then, we obtain an English parse tree by removing the leaf nodes (foreign words) from the back-reordered tree. Among the possible decoded trees, we pick the best tree in which the product of the LM probability (the prior probability of the English tree) and the TM probability (the probabilities associated

[2]The training corpus for the syntax-based model consists of pairs of English parse trees and foreign sentences.

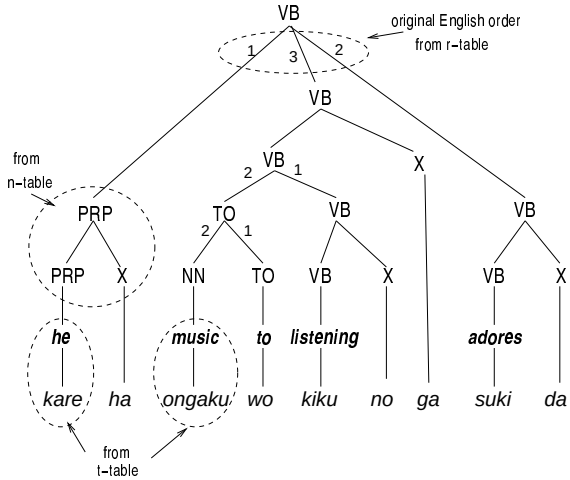

Figure 2: Decoded Tree

with the rules in the decoded tree) is the highest.

The use of an LM needs consideration. Theoretically we need an LM which gives the prior probability of an English parse tree. However, we can approximate it with an n-gram LM, which is well-studied and widely implemented. We will discuss this point later in Section 7.

If we use a trigram model for the LM, a convenient implementation is to first build a decoded-tree forest and then to pick out the best tree using a trigram-based forest-ranking algorithm as described in (Langkilde, 2000). The ranker uses two leftmost and rightmost leaf words to efficiently calculate the trigram probability of a subtree, and finds the most plausible tree according to the trigram and the rule probabilities. This algorithm finds the optimal tree in terms of the model probability — but it is not practical when the vocabulary size and the rule size grow. The next section describes how to make it practical.

5 Pruning

We use our decoder for Chinese-English translation in a general news domain. The TM becomes very huge for such a domain. In our experiment (see Section 6 for details), there are about 4M non-zero entries in the trained $t(f|e)$ table. About 10K CFG rules are used in the parsed corpus of English, which results in about 120K non-lexical rules for the decoding grammar (after we expand the CFG rules as described in Section 4). We applied the simple algorithm from Section 4, but this experiment failed — no complete translations were produced. Even four-word sentences could not be decoded. This is not only because the model size is huge, but also because the decoder considers multiple syntactic structures for the same word alignment, i.e., there are several different decoded trees even when the translation of the sentence is the same. We then applied the following measures to achieve practical decoding. The basic idea is to use additional statistics from the training corpus.

beam search: We give up optimal decoding by using a standard dynamic-programming parser with beam search, which is similar to the parser used in (Collins, 1999). A standard dynamic-programming parser builds up <nonterminal, input-substring> tuples from bottom-up according to the grammar rules. When the parsing cost[3] comes only from the features within a subtree (TM cost, in our case), the parser will find the optimal tree by keeping the single best subtree for each tuple. When the cost depends on the features outside of a subtree, we need to keep all the subtrees for possible different outside features (boundary words for the trigram LM cost) to obtain the optimal tree. Instead of keeping all the subtrees, we only retain subtrees within a beam width for each input-substring. Since the outside features are not considered for the beam pruning, the optimality of the parse is not guaranteed, but the required memory size is reduced.

t-table pruning: Given a foreign (Chinese) sentence to the decoder, we only consider English words e for each foreign word f such that $\mathrm{P}(e|f)$ is high. In addition, only limited part-of-speech labels ℓ are considered to reduce the number of possible decoded-tree structures. Thus we only use the top-5 (e,ℓ) pairs ranked by

$$\begin{aligned}\mathrm{P}(e,\ell|f) &= \mathrm{P}(\ell)\mathrm{P}(e|\ell)\mathrm{P}(f|e,\ell)/\mathrm{P}(f)\\ &\simeq \mathrm{P}(\ell)\mathrm{P}(e|\ell)\mathrm{P}(f|e).\end{aligned}$$

Notice that $\mathrm{P}(f|e)$ is a model parameter, and that $\mathrm{P}(\ell)$ and $\mathrm{P}(e|\ell)$ are obtained from the parsed training corpus.

phrase pruning: We only consider limited pairs $(e_1 e_2 \ldots e_m, f_1 f_2 \ldots f_l)$ for phrasal translation (see

[3] rule-cost = $-log$(rule-probability)

Section 2). The pair must appear more than once in the Viterbi alignments[4] of the training corpus. Then we use the top-10 pairs ranked similarly to t-table pruning above, except we replace $\mathrm{P}(\ell)\mathrm{P}(e|\ell)$ with $\mathrm{P}(e)$ and use trigrams to estimate $\mathrm{P}(e)$. By this pruning, we effectively remove junk phrase pairs, most of which come from misaligned sentences or untranslated phrases in the training corpus.

r-table pruning: To reduce the number of rules for the decoding grammar, we use the top-N rules ranked by $\mathrm{P(rule)P(reord)}$ so that $\sum_{i=1}^{N} \mathrm{P(rule}_i\mathrm{)P(reord}_i\mathrm{)} > 0.95$, where $\mathrm{P(rule)}$ is a prior probability of the rule (in the original English order) found in the parsed English corpus, and $\mathrm{P(reord)}$ is the reordering probability in the TM. The product is a rough estimate of how likely a rule is used in decoding. Because only a limited number of reorderings are used in actual translation, a small number of rules are highly probable. In fact, among a total of 138,662 reorder-expanded rules, the most likely 875 rules contribute 95% of the probability mass, so discarding the rules which contribute the lower 5% of the probability mass efficiently eliminates more than 99% of the total rules.

zero-fertility words: An English word may be translated into a null (zero-length) foreign word. This happens when the fertility $\mu(0|e) > 0$, and such English word e (called a *zero-fertility word*) must be inserted during the decoding. The decoding parser is modified to allow inserting zero-fertility words, but unlimited insertion easily blows up the memory space. Therefore only limited insertion is allowed. Observing the Viterbi alignments of the training corpus, the top-20 frequent zero-fertility words[5] cover over 70% of the cases, thus only those are allowed to be inserted. Also we use syntactic context to limit the insertion. For example, a zero-fertility word *in* is inserted as IN when "PP $\rightarrow$ IN NP-A" rule is applied. Again, observing the Viterbi alignments, the top-20 frequent contexts cover over 60% of the cases, so we allow insertions only in these contexts. This kind of context sensitive insertion is possible because the decoder builds a syntactic tree. Such selective insertion by syntactic context is not easy for a word-for-word based IBM model decoder.

[4]Viterbi alignment is the most probable word alignment according to the trained TM tables.

[5]They are *the, to, of, a, in, is, be, that, on, and, are, for, will, with, have, it, 's, has, i,* and *by*.

system	P1/P2/P3/P4	LP	BLEU
ibm4	36.6/11.7/4.6/1.6	0.959	0.072
syn	39.8/15.8/8.3/4.9	0.781	0.099
syn-nozf	40.6/15.3/8.1/5.3	0.797	0.102

Table 1: Decoding performance

The pruning techniques shown above use extra statistics from the training corpus, such as $\mathrm{P}(\ell)$, $\mathrm{P}(e|\ell)$, and $\mathrm{P(rule)}$. These statistics may be considered as a part of the LM $\mathrm{P}(E)$, and such syntactic probabilities are essential when we mainly use trigrams for the LM. In this respect, the pruning is useful not only for reducing the search space, but also improving the quality of translation. We also use statistics from the Viterbi alignments, such as the phrase translation frequency and the zero-fertility context frequency. These are statistics which are not modeled in the TM. The frequency count is essentially a joint probability $\mathrm{P}(f, e)$, while the TM uses a conditional probability $\mathrm{P}(f|e)$. Utilizing statistics outside of a model is an important idea for statistical machine translation in general. For example, a decoder in (Och and Ney, 2000) uses alignment template statistics found in the Viterbi alignments.

6 Experimental Results: Chinese/English

This section describes results from our experiment using the decoder as described in the previous section. We used a Chinese-English translation corpus for the experiment. After discarding long sentences (more than 20 words in English), the English side of the corpus consisted of about 3M words, and it was parsed with Collins' parser (Collins, 1999). Training the TM took about 8 hours using a 54-node unix cluster. We selected 347 short sentences (less than 14 words in the reference English translation) from the held-out portion of the corpus, and they were used for evaluation.

Table 1 shows the decoding performance for the test sentences. The first system ibm4 is a reference system, which is based on IBM Model4. The second and the third (syn and syn-nozf) are our decoders. Both used the same decoding algorithm and pruning as described in the previous sections, except that syn-nozf allowed no zero-fertility insertions. The

average decoding speed was about 100 seconds[6] per sentence for both syn and syn-nozf.

As an overall decoding performance measure, we used the BLEU metric (Papineni et al., 2002). This measure is a geometric average of n-gram accuracy, adjusted by a length penalty factor LP.[7] The n-gram accuracy (in percentage) is shown in Table 1 as P1/P2/P3/P4 for unigram/bigram/trigram/4-gram. Overall, our decoder performed better than the IBM system, as indicated by the higher BLEU score. We obtained better n-gram accuracy, but the lower LP score penalized the overall score. Interestingly, the system with no explicit zero-fertility word insertion (syn-nozf) performed better than the one with zero-fertility insertion (syn). It seems that most zero-fertility words were already included in the phrasal translations, and the explicit zero-fertility word insertion produced more garbage than expected words.

system	Coverage
r95	92/92
r98	47/92
r100	20/92

system	Coverage
w5	92/92
w10	89/92
w20	69/92

Table 2: Effect of pruning

To verify that the pruning was effective, we relaxed the pruning threshold and checked the decoding coverage for the first 92 sentences of the test data. Table 2 shows the result. On the left, the r-table pruning was relaxed from the 95% level to 98% or 100%. On the right, the t-table pruning was relaxed from the top-5 (e,ℓ) pairs to the top-10 or top-20 pairs. The system r95 and w5 are identical to syn-nozf in Table 1.

When r-table pruning was relaxed from 95% to 98%, only about half (47/92) of the test sentences were decoded, others were aborted due to lack of memory. When it was further relaxed to 100% (i.e., no pruning was done), only 20 sentences were decoded. Similarly, when the t-table pruning threshold was relaxed, fewer sentences could be decoded due to the memory limitations.

Although our decoder performed better than the IBM system in the BLEU score, the obtained gain was less than what we expected. We have thought the following three reasons. First, the syntax of Chinese is not extremely different from English, compared with other languages such as Japanese or Arabic. Therefore, the TM could not take advantage of syntactic reordering operations. Second, our decoder looks for a decoded tree, not just for a decoded sentence. Thus, the search space is larger than IBM models, which might lead to more search errors caused by pruning. Third, the LM used for our system was exactly the same as the LM used by the IBM system. Decoding performance might be heavily influenced by LM performance. In addition, since the TM assumes an English parse tree as input, a trigram LM might not be appropriate. We will discuss this point in the next section.

Phrasal translation worked pretty well. Figure 3 shows the top-20 frequent phrase translations observed in the Viterbi alignment. The leftmost column shows how many times they appeared. Most of them are correct. It even detected frequent sentence-to-sentence translations, since we only imposed a relative length limit for phrasal translations (Section 3). However, some of them, such as the one with *(in cantonese)*, are wrong. We expected that these junk phrases could be eliminated by phrase pruning (Section 5), however the junk phrases present many times in the corpus were not effectively filtered out.

7 Decoded Trees

The BLEU score measures the quality of the decoder output sentences. We were also interested in the syntactic structure of the decoded trees. The leftmost tree in Figure 4 is a decoded tree from the syn-nozf system. Surprisingly, even though the decoded sentence is passable English, the tree structure is totally unnatural. We assumed that a good parse tree gives high trigram probabilities. But it seems a bad parse tree may give good trigram probabilities too. We also noticed that too many unary rules (e.g. "NPB $\rightarrow$ PRN") were used. This is because the reordering probability is always 1.

To remedy this, we added CFG probabilities (PCFG) in the decoder search, i.e., it now looks for a tree which maximizes P(trigram)P(cfg)P(TM). The CFG probability was obtained by counting the rule

[6] Using a single-CPU 800Mhz Pentium III unix system with 1GB memory.

[7] BLEU $= \exp(\sum_{n=1}^{N} w_n \log p_n) \times \mathrm{LP}$. $\mathrm{LP} = \exp(1 - r/c)$ if $c \leq r$, and $\mathrm{LP} = 1$ if $c > r$, where $w_n = 1/N$, $N = 4$, c is the system output length, and r is the reference length.

```
2067  the government -> 政 府
 985  the following -> 下
 862  madam president -> 主 席 女 士
 807  in this connection -> 就 此
 575  madam president -> 主 席
 512  is issued on behalf of the provisional urban council -> 稿 代 临 时 市 政 局 发
 495  hong kong -> 香 港
 464  inform this council -> 可 否 告 知 本 会
 453  effective exchange rate index -> 港 汇 指 数
 436  is issued on behalf of the provisional regional council -> 稿 代 临 时 区 域 市 政 局 发
 340  here is an item of interest to swimmers -> 请 尽 速 播 出 以 下 特 别 报 告
 301  attention tv/radio announcers please broadcast the following as soon as possible -> 电 台 及 电 视 台 当 值 宣 布 员 注 意
 294  mr president -> 主 席 先 生
 286  thank you -> 谢 谢
 238  red flags hoisted -> 海 滩 悬 挂 红 旗
 230  president ( in cantonese ) -> 谢 谢 主 席
 224  thank you madam president -> 谢 谢 主 席
 223  put and agreed to -> 付 诸 表 决 并 获 通 过
 220  proposed amendment -> 拟 议 修 正 案 内 容
 205  thank you mr president -> 谢 谢 主 席
```

Figure 3: Top-20 frequent phrase translations in the Viterbi alignment

frequency in the parsed English side of the training corpus. The middle of Figure 4 is the output for the same sentence. The syntactic structure now looks better, but we found three problems. First, the BLEU score is worse (0.078). Second, the decoded trees seem to prefer noun phrases. In many trees, an entire sentence was decoded as a large noun phrase. Third, it uses more frequent node reordering than it should.

The BLEU score may go down because we weighed the LM (trigram and PCFG) more than the TM. For the problem of too many noun phrases, we thought it was a problem with the corpus. Our training corpus contained many dictionary entries, and the parliament transcripts also included a list of participants' names. This may cause the LM to prefer noun phrases too much. Also our corpus contains noise. There are two types of noise. One is sentence alignment error, and the other is English parse error. The corpus was sentence aligned by automatic software, so it has some bad alignments. When a sentence was misaligned, or the parse was wrong, the Viterbi alignment becomes an over-reordered tree as it picks up plausible translation word pairs first and reorders trees to fit them.

To see if it was really a corpus problem, we selected a good portion of the corpus and re-trained the r-table. To find good pairs of sentences in the corpus, we used the following: 1) Both English and Chinese sentences end with a period. 2) The English word is capitalized at the beginning. 3) The sentences do not contain symbol characters, such as colon, dash etc, which tend to cause parse errors. 4) The Viterbi-ratio[8] is more than the average of the pairs which satisfied the first three conditions.

Using the selected sentence pairs, we retrained only the r-table and the PCFG. The rightmost tree in Figure 4 is the decoded tree using the re-trained TM. The BLEU score was improved (0.085), and the tree structure looks better, though there are still problems. An obvious problem is that the goodness of syntactic structure depends on the lexical choices. For example, the best syntactic structure is different if a verb requires a noun phrase as object than it is if it does not. The PCFG-based LM does not handle this.

At this point, we gave up using the PCFG as a component of the LM. Using only trigrams obtains the best result for the BLEU score. However, the BLEU metric may not be affected by the syntactic aspect of translation quality, and as we saw in Figure 4, we can improve the syntactic quality by introducing the PCFG using some corpus selection techniques. Also, the pruning methods described in Section 5 use syntactic statistics from the training corpus. Therefore, we are now investigating more sophisticated LMs such as (Charniak, 2001) which

[8] Viterbi-ratio is the ratio of the probability of the most plausible alignment with the sum of the probabilities of all the alignments. Low Viterbi-ratio is a good indicator of misalignment or parse error.

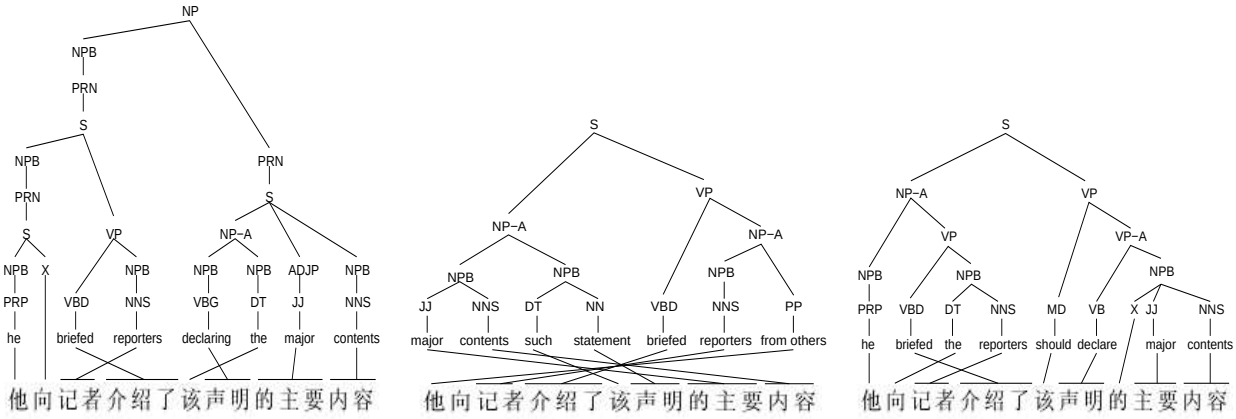

Figure 4: Effect of PCFG and re-training: No CFG probability (PCFG) was used (left). PCFG was used for the search (middle). The r-table was re-trained and PCFG was used (right). Each tree was back reordered and is shown in the English order.

incorporate syntactic features and lexical information.

8 Conclusion

We have presented a decoding algorithm for a syntax-based statistical machine translation. The translation model was extended to incorporate phrasal translations. Because the input of the channel model is an English parse tree, the decoding algorithm is based on conventional syntactic parsing, and the grammar is expanded by the channel operations of the TM. As the model size becomes huge in a practical setting, and the decoder considers multiple syntactic structures for a word alignment, efficient pruning is necessary. We applied several pruning techniques and obtained good decoding quality and coverage. The choice of the LM is an important issue in implementing a decoder for the syntax-based TM. At present, the best result is obtained by using trigrams, but a more sophisticated LM seems promising.

Acknowledgments

This work was supported by DARPA-ITO grant N66001-00-1-9814.

References

H. Alshawi, S. Bangalore, and S. Douglas. 2000. Learning dependency translation models as collections of finite state head transducers. *Computational Linguistics*, 26(1).

A. Berger, P. Brown, S. Della Pietra, V. Della Pietra, J. Gillett, J. Lafferty, R. Mercer, H. Printz, and L. Ures. 1996. *Language Translation Apparatus and Method Using Context-Based Translation Models*. U.S. Patent 5,510,981.

P. Brown, S. Della Pietra, V. Della Pietra, and R. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2).

E. Charniak. 2001. Immediate-head parsing for language models. In *ACL-01*.

M. Collins. 1999. *Head-Driven Statistical Models for Natural Language Parsing*. Ph.D. thesis, University of Pennsylvania.

I. Langkilde. 2000. Forest-based statistical sentence generation. In *NAACL-00*.

F. Och and H. Ney. 2000. Improved statistical alignment models. In *ACL-2000*.

K. Papineni, S. Roukos, T. Ward, and W. Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *ACL-02*.

D. Wu. 1997. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computational Linguistics*, 23(3).

K. Yamada and K. Knight. 2001. A syntax-based statistical translation model. In *ACL-01*.

K. Yamada. 2002. *A Syntax-Based Statistical Translation Model*. Ph.D. thesis, University of Southern California.