A Quirk Review of Translation Models

Jianfeng Gao, Microsoft Research

Prepared in connection with the 2010 ACL/SIGIR summer school

July 22, 2011

The goal of machine translation (MT) is to use a computer system to translate a text written in a source language (e.g., Chinese) into a target language (e.g., English). In this section, we will give an overview of translation models that are widely used in state-of-the-art statistical machine translation (SMT) systems. A nice textbook that provides a comprehensive review is Koehn (2010). Although these models are designed for translating regular natural language sentences, they can also be adapted to the task of search query translation for cross-lingual information retrieval (CLIR). The query translation task differs from the regular text translation mainly in its treatment of word order. In text translation word order is crucial to the readability of the translated sentences, which are present to the end users directly. In CLIR, query translation is an intermediate step that provides a set of translated query terms for the search engine to retrieve documents in the target language. Therefore, word order has little impact on the search quality as long as the translation preservers the underlying search intent, not the appearance, of the original query. This section focuses only on statistical translation models for regular text. Readers who are interested in statistical models for query translation may refer to Gao and Nie (2006) and Gao et al. (2001, 2002).

1. SMT and Generative Translation Models

SMT is typically formulated under the framework of the noisy channel model. Given a source sentence (in Chinese) $C = c_1 \ldots c_J$, we want to find the best English translation $E = e_1 \ldots e_I$ among all possible translations:

$$E^* = \underset{E}{\operatorname{argmax}}\, P(E|C) \tag{2.1}$$

where the argmax operation denotes the decoder, i.e., the search algorithm used to find among all possible target sentences the one with the highest probability.

Applying Bayes' decision rule and dropping the constant denominator, we have

$$E^* = \underset{E}{\operatorname{argmax}}\, P(C|E)P(E) \tag{2.2}$$

where $P(E)$ is the language model, assessing the overall well-formedness of the target sentence, and $P(C|E)$ is the translation model, modeling the transformation probability from E to $\boldsymbol{C}$. In this section, we focus our discussion on the translation model only. Notice that, mathematically, the translation direction has changed from $P(E|C)$ in Equation (2.1) to $P(C|E)$ in Equation (2.2). This could cause a lot of confusion since the concept of what constitutes the source language differs between the mathematical model and the actually application. Following Koehn (2010), we try to avoid confusion by sticking to the notation $P(E|C)$ when describing a translation model.

As a significant generalization of the noisy channel model, a log-linear model that models $P(E|C)$ directly has been proposed by Och and Ney (2002), and has been used by most of state-of-the-art SMT systems. The model is of the form

$$P(E|C) = \frac{1}{Z(C,E)} \exp \sum_i \lambda_i h_i(C,E) \tag{2.3}$$

where Z is the normalization constant, $h(.)$ are a set of features computed over the translation and λ's are the feature weights optimized on development data using e.g., minimum error rate training (Och 2003). Notice that the features used in the log-linear model can be binary features or real-value features derived from some probabilistic models. For example, we can define the logarithm of language model and translation model probabilities in Equation (2.2) as two features, thus covering the noisy channel model as a special case. The log-linear model provides a flexible mathematical framework to incorporate a wide variety of features useful for MT.

Now, let us turn our attention to the translation model $P(E|C)$. Conceptually, a translation model tries to remember as much as possible how likely a source sentence is translated into a target sentence in training data. Figure 1 shows a pair of Chinese sentence and its English translation. Ideally, if the translation model could remember such translation pairs for all possible Chinese sentences, we would have a perfect translation system. Unfortunately, no matter how large the training corpus is, it covers only a tiny portion of the sentences. Therefore, we have to break the sentences into smaller *translation units* (e.g., words) whose distribution (i.e., translation probabilities) can be more easily modeled given limited training data. As shown in Figure 1, although the translation of the full sentence is unlikely to occur in training data, the word translation pairs such as (rescue, 救援) do. Given an input sentence that is unseen in training data, an SMT system is expected to perform a translation process as follows: first the input source sentence is broken into smaller translation units, then each unit is translated into a target language, finally the translated units are glued to form a target sentence. The translation models described below differ in how the translation units are defined, translated and glued. The modeling method we use to formulate a translation model is called **generative modeling**. It consists of three steps:

- Story making: coming up with a generative story about how a target sentence is generated from a source sentence step by step.
- Mathematical formulation: modeling each generation steps in the generative story using a probability distribution.
- Parameter estimation: implementing an effective way of estimating the probability distributions from training data.

These three modeling tasks are highly related. For example, the way we break the generation process into smaller steps in our story determines the complexity of the probabilistic models, which in consequence determines the set of the model parameters that need to be estimated. We can view the three tasks as work of art (story making), science (mathematical formulation), and engineering (parameter estimation), respectively. The overall task of generative modeling is to seek a harmonic combination of them, a task that attracts some of the best computer scientists all over the world to devote their talents.

Most of state-of-the-art translation models used for regular text translation can be grouped into three categories: word-based models, phrase-based models, and syntax-based models. In what follows, we will describe them in turn. For each type of the models, we start with a generative story, and then describe the mathematical formulation and the way the model parameters are estimated on training data.

救援(rescue) 人员(staff) 在(in) 倒塌的(collapsed) 房屋(house) 里(in) 寻找(search) 生还者(survivors)。

Rescue workers search for survivors in collapsed houses.

Figure 1. A Chinese sentence and its English translation.

2. Word-Based Models

Word-based models use words as translation units. The models stem from the pioneering work on statistical machine translation conducted by the IBM group in the early 1990s. In their classical paper (Brown et al., 1993), a series of word-based translation models of increasing complexity were proposed. These models are called the IBM Models.

Let us start with IBM Model 1, one of the simplest and most widely used word-based models. This model is also called a *lexical translation model*, where the order of the words in the source and target sentence is ignored. As the first step of modeling, we need to tell a generative story about how the target sentence E is generated from the source sentence C. Here is the story.

1. We first choose the length for the target sentence I, according to the distribution $P(I|C)$.
2. Then, for each position i ($i = 1 \ldots I$) in the target sentence, we choose a position j in the source sentence from which to generate the i-th target word e_i according to the distribution $P(j|C)$, and generate the target word by translating c_j according to the distribution $P(e_i|c_j)$. We include in position zero of the source sentence an artificial "null word", denoted by <null>. The purpose of the null word is to insert additional target words.

Now, let us formulate the above story mathematically. In Step 1, we assume that the choice of the length is independent of C and I, thus we have $P(I|C) = \epsilon$, where ϵ is a small constant. In Step 2, we assume that all positions in the source sentence, including position zero for the null word, are equally likely to be chosen. Thus we have $P(j|C) = \frac{1}{J+1}$. Then the probability of generating e_i given C is the sum over all possible positions, weighted by $P(j|C)$: $P(e_i|C) = \sum_j P(j|C)P(e_i|c_j) = \frac{1}{J+1}\sum_j P(e_i|c_j)$. Assuming that each target word is generated independently from C, we end up with the final form of IBM Model 1

$$P(E|C) = P(I|C)\prod_{\boldsymbol{i=1}}^{\boldsymbol{I}} P(e_i|C) \tag{2.4}$$

$$= \frac{\epsilon}{(J+1)^I}\prod_{\boldsymbol{i=1}}^{\boldsymbol{I}}\sum_{j=0}^{J} P(e_i|c_j) \tag{2.5}$$

We can see that IBM Model 1 has only one type of parameters to estimate, the lexical translation probabilities $P(e|c)$. Now, let us discuss how to automatically obtain the values of $P(e|c)$ from data. If the training data consists of sentence pairs that are word-aligned as shown in Figure 2, $P(e|c)$ can be computed via Maximum Likelihood Estimation (MLE) as follows:

$$P(e|c) = \frac{N(c,e)}{\sum_{e'} N(c,e')} \tag{2.4}$$

where $N(c,e)$ is the number of times that the word pair (c,e) is aligned in training data. In reality, it is more realistic to assume that training data is aligned at the sentence level but not at the word level. Therefore, we resort to the Expectation Maximization (EM) algorithm to compute the values of $P(e|c)$ and the word alignment iteratively. The algorithm works as follows:

1. We initialize the model with a uniform translation probability distribution
2. We apply the model to the data, computing the probabilities of all possible word alignments
3. We (re-)estimate the model by collecting counts for word translation over all possible alignments, weighted by their probabilities computed in the Step 2.
4. We iterate through Steps 2 and 3 until convergence.

Since at every EM iteration the likelihood of the model given the training data is guaranteed not to decrease, the EM algorithm is guaranteed to converge. In the case of IBM Model 1, the EM algorithm is guaranteed to reach a global maximum.

Brown et al. (1993) presents five word-based translation models of increasing complexity, namely IBM Model 1 to 5. As described above, in IBM Model 1 the order of the words in the source and target sentences is ignored. In other words, the model assumes that all word alignments are equally likely. Model 2 improves on Model 1 by adding an absolute alignment model such that words that follow each other in the source language have translations that follow each other in the target language. Models 3, 4, and 5 model the “fertility” of the generation process with increasing complexity. Fertility is a notion reflecting that an input word in a source language produces a specific number of output words in a target language. The fertility model captures the information that some Chinese words are more likely than other words to generate multiple English words. All these models, similar to Model 1, have their generative stories and the corresponding mathematical formulations, and their model parameters are estimated using the EM algorithm. Readers may refer to Brown et al. (1993) for a detailed description.

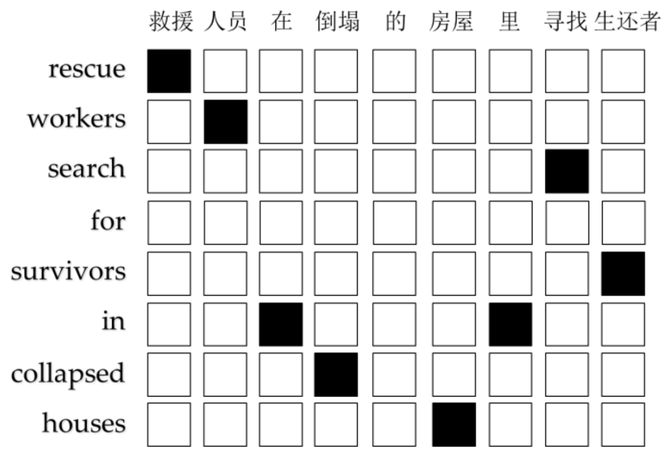


Figure 2. Word alignment: words in the English sentence (rows) are aligned to words in the Chinese sentence (columns) as indicated by the filled boxes in the matrix.

3. Phrase-Based Models

The phrase-based models form the basis for most of state-of-the-art SMT systems. Like the word-based models, the phrase-based models are generative models that translate an input sentence in a source language $\boldsymbol{C}$ into a sentence in a target language E. Unlike the word-based models that translate single words in isolation, the phrase-based models translate sequences of words (i.e., phrases) in C into sequences of words in E. The use of phrases as translation units is motivated by the observation that sometimes one word in a source language translates into multiple words in a target language, or vice versa. Word-based models cannot handle these cases properly. For example, the English phrase "stuffy nose" can be translated from the Chinese word "鼻塞" with relatively high probability, but neither of the individual English word (i.e., "stuffy" and "nose") has a high word translation probability to "鼻塞".

The generative story behind the phrase-based models can be stated as follows. First, the input source sentence C is segmented into K non-empty word sequences $\mathbf{c}_1 \ldots \mathbf{c}_K$. Then each of them is translated to a new non-empty word sequence $\mathbf{e}_1 \ldots \mathbf{e}_K$. Finally these phrases are permuted and concatenated to form the target sentence *E*. Here **c** and **e** denote consecutive sequences of words.

To formalize this generative process, let *S* denote the segmentation of *C* into *K* phrases $\mathbf{c}_{1\ldots}\mathbf{c}_K$, and let *T* denote the *K* translation phrases $\mathbf{e}_1\ldots\mathbf{e}_K$ – we refer to these $(\mathbf{c}_i, \mathbf{e}_i)$ pairs as *bilingual phrases*. Finally, let *M* denote a permutation of *K* elements representing the final reordering step. Figure 3 demonstrates the generative procedure.

Next let us place a probability distribution over translation pairs. Let $B(C, E)$ denote the set of *S*, *T*, *M* triples that translate *C* into *E*. If we assume a uniform probability over segmentations, then the phrase-based translation model can be defined as:

$$P(E|C) \propto \sum_{\substack{(S,T,M)\in \\ B(C,E)}} P(T|C,S) \cdot P(M|C,S,T) \tag{2.5}$$

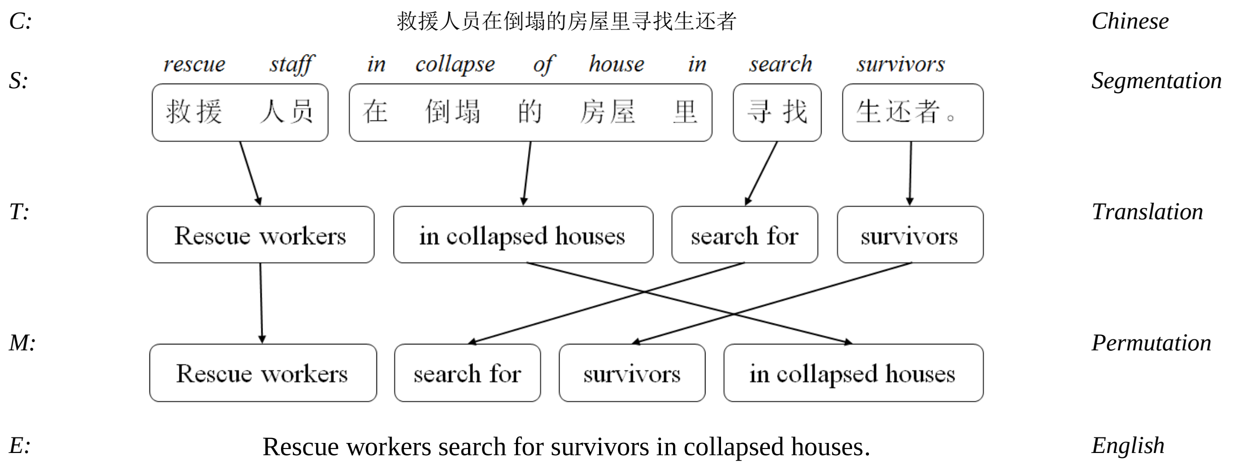


Figure 3: Example demonstrating the generative procedure behind the phrase-based model.

As is common practice in SMT, we use the maximum approximation to the sum, and the maximum probability assignment can be found efficiently by using a dynamic programming approach:

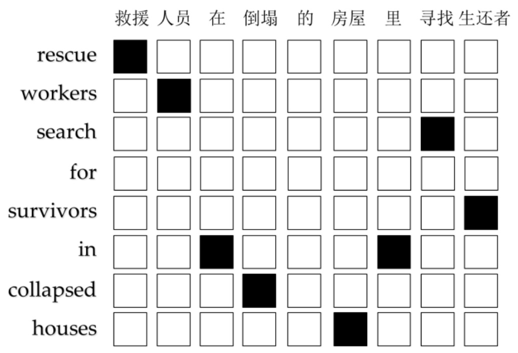
$$P(E|C) \approx \max_{\substack{(S,T,M)\in \\ B(C,E)}} P(T|C,S) \cdot P(M|C,S,T) \tag{2.6}$$

Reordering is handled by a distance-based reordering model (Koehn et al. 2003). We consider reordering relative to the previous phrase. We define $start_i$ as the position of the first word of the Chinese input phrase that translates to the *i*-th English phrase, and end_i as the position of the last word of that Chinese phrase. Reordering distance is computed as $start_i - end_i - 1$, i.e., the number of words skipped when taking foreign words out of sequence. We also make the assumption that a phrase-segmented English sentence $T = \mathbf{e}_1\ldots\ \mathbf{e}_K$ is generated from left to right by translating each phrase $\mathbf{c}_1\ldots\mathbf{c}_K$ independently. We then end up with one of the most well known forms of phrase-based model:

$$P(E|C) \propto \max_{\substack{(S,T,M)\in \\ B(C,Q)}} \prod_{k=1}^{K} P(\mathbf{e}_k|\mathbf{c}_k) d(start_i - end_{i-1} - 1) \tag{2.7}$$

We see from Equation (2.7) that the only type of parameters to estimate is the translation probabilities on the bilingual phrases $P(\mathbf{e}|\mathbf{c})$. In what follows, we will describe the way the bilingual phrases are extracted from the parallel data and the way $P(\mathbf{e}|\mathbf{c})$ is estimated. Our description is based mainly on work by Och and Ney (2002) and Koehn et al. (2003).

First, we learn two word translation models using the EM training of a word-based model (i.e., IBM Model 1 or 4) on sentence pairs in two directions: One is from source to target and the other from target to source. We then perform Viterbi word alignment in each direction according to the models in two directions, respectively. The two alignments are combined as follows: we start from the intersection of the two alignments, and gradually include more alignment links according to a set of heuristic rules described in Och and Ney (2002). Finally, the bilingual phrases that are consistent with the word alignment are extracted. Consistency here implies two things. First, there must be at least one aligned word pair in the bilingual phrase. Second, there must not be any word alignments from words inside the bilingual phrase to words outside the bilingual phrase. That is, we do not extract a phrase pair if there is an alignment from within the phrase pair to outside the phrase pair. Figure 4 illustrates the bilingual phrases we can generate from the word-aligned sentence pair by this process.



(救援, rescue)
(人员, workers)
(在, in)
(倒塌, collapsed)
(房屋, houses)
(里, in)
(寻找, search)
(生还者, survivors)
(救援 人员, rescue workers)
(在 倒塌, in collapsed)
(倒塌 的, collapsed)
(的 房屋, houses)
(寻找, search for)
(寻找 生还者, search for survivors)
(生还者, for survivors)
(倒塌 的 房屋, collapsed houses)

Figure 4. An example of a word alignment and the bilingual phrases containing up to 3 words that are consistent with the word alignment.

After gathering all such bilingual phrases from the training data, we can estimate conditional relative frequency estimates without smoothing. For example, the phrase transformation probability $P(\mathbf{e}|\mathbf{c})$ in Equation (2.7) can be estimated approximately as

$$P(\mathbf{e}|\mathbf{c}) = \frac{N(\mathbf{c},\mathbf{e})}{\sum_{\mathbf{e}'} N(\mathbf{c},\mathbf{e}')} \tag{2.8}$$

where $N(\mathbf{c},\mathbf{e})$ is the number of times that **c** is aligned to **e** in training data. These estimates are useful for contextual lexical selection with sufficient training data, but can be subject to data sparsity issues.

An alternate translation probability estimate not subject to data sparsity issues is the so-called *lexical weight* estimate. Assume we have a word translation distribution $t(e|c)$ (defined over individual words, not phrases), and a word alignment A between **e** and **c**; here, the word alignment contains (i, j) pairs, where $i \in 1..|\mathbf{e}|$ and $j \in 0..|\mathbf{c}|$, with 0 indicating an inserted word. Then we can use the following estimate:

$$P_w(\mathbf{e}|\mathbf{c}, A) = \prod_{i=1}^{|\mathbf{e}|} \frac{1}{|\{j|(j,i) \in A\}|} \sum_{\forall(i,j)\in A} t(e_i|c_j) \tag{2.9}$$

We assume that for every position in **e**, there is either a single alignment to 0, or multiple alignments to non-zero positions in **c**. In effect, this computes a product of per-word translation scores; the per-word scores are averages of all the translations for the alignment links of that word. We estimate the word translation probabilities using counts from the word aligned corpus: $t(q|c) = \frac{N(c,e)}{\sum_{e'} N(c,e)}$. Here $N(c,e)$ is the number of times that the words (not phrases as in Equation (2.8)) c and e are aligned in the training data. These word-based scores of bilingual phrases, though not as effective in contextual selection, are more robust to noise and sparsity. Both model forms of Equation (2.8) and (2.9) are used as features in the log-linear model for SMT as Equation (2.3).

4. Syntax-Based Models

Exploring syntax information for SMT is a long-standing research topic. Recently, some syntax-based translation models start to perform as well as state-of-the-art phrase-based models, and for some language pairs even outperform them. Research on syntax-based models is a fast-moving area, with a lot of open questions. Our description in this section focuses on some basic underlying principles, illustrated by examples of some of the most successful models proposed so far (e.g., Chiang 2005; Galley et al. 2004).

Syntax-based models reply on parsing the sentence in the source or the target language, or in some cases in both languages. Our earlier example sentence pair with both parses is shown in Figure 5. These parses are generated from a statistical parser trained on Penn Treebank. The parsing output in Figure 5 is a constituent parse. It is a rooted tree where the leaves are original words of the sentence and the internal nodes cover a contiguous sequence of the words in the sentence, called a constituent, and to each of these constituents is associated a phrase label describing the syntactic role of the words under this node.

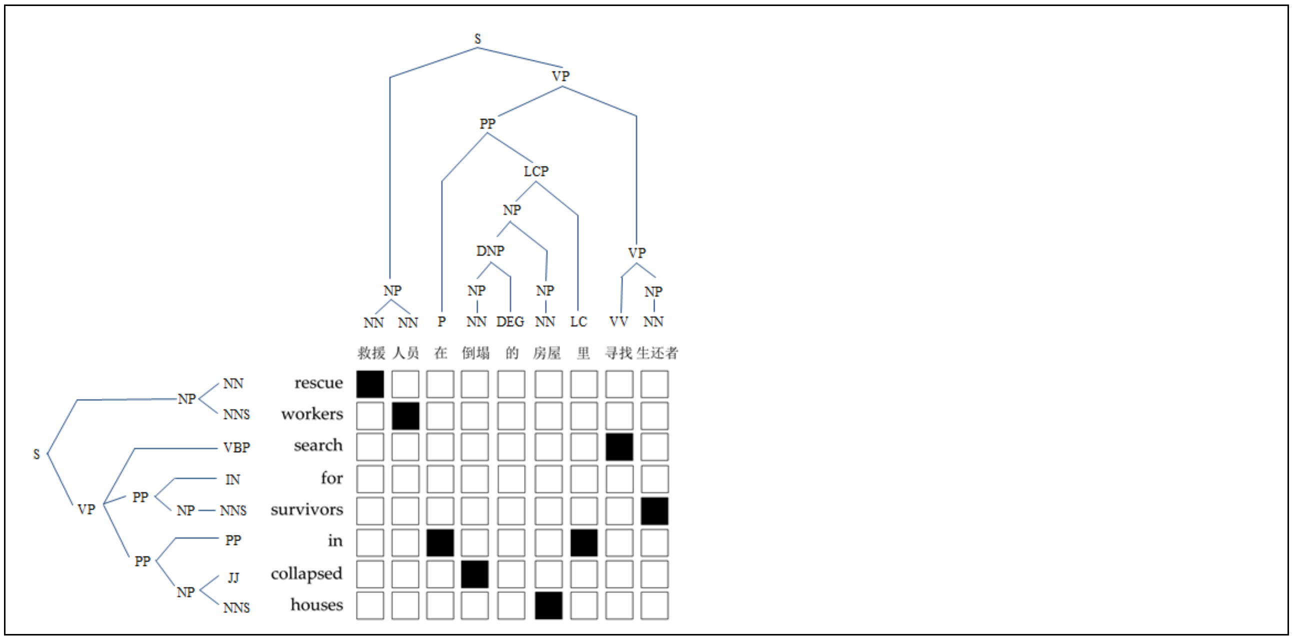


Figure 5. A pair of word-aligned Chinese and English sentences and their parse trees.

The tree-structured parse plays similar roles in the syntax-based models to phrase in the phrase-based models. The first role is identifying translation units in an input sentence. While in the phrase-based models the units are phrases, in the syntax-based models they are constituents, as shown in Figure 5. The second is to guide how to best glue those translated constituents into a well-formed target sentence. We assume a generative story, similar to that for the phrase-based models:

1. Parse an input Chinese sentence into a parse tree
2. Translate each Chinese constituent into English
3. Glue these English constituents into a well-formed English sentence.

This generative process is typically formulated under the framework of weighted synchronous Context Free Grammar (CFG) (Chiang, 2005). The grammar consists of a set of rewriting rules *r* of the form

$$X \to (\gamma, \alpha, \sim) \tag{2.10}$$

where X is a nonterminal, γ and α are both strings of terminals and non-terminals corresponding respectively to source and target strings, and ~ indicates that any non-terminals in the source and target strings are aligned. For example, a rule extracted from the example in Figure 5 is

$$\text{VP} \rightarrow (\text{PP 寻找 NP, search for NP PP}) \tag{2.11}$$

We can see that these non-terminals generalize the phrases used in the phrase-based models described in Section 2.3.

We now define a derivation *D* as a sequence of *K* rewrites $r_1 \ldots r_K$, each of which picks a rewriting rule from the grammar, and rewrites a constituent in Chinese into English, until an English sentence is generated. Let *E*(*D*) be the English strings generated by *D*, and *C*(*D*) be the Chinese strings generated by *D*. Assuming that the parse tree of the input Chinese sentence is $Tree(C)$, the translation model can be formulated as

$$P(E|C, Tree(C)) = \sum_{\substack{D:E(D)=E \\ and\ C(D)=C}} P(D) \tag{2.12}$$

As in formulating the phrase-based models, we use the maximum approximation to the sum:

$$P(E|C, Tree(C)) \propto \max_{\substack{D:E(D)=E \\ and\ C(D)=C}} P(D) \tag{2.13}$$

A synchronous CFG assumes that each rewriting rule application depends only on a non-terminal, and not on any surrounding context. Thus we have

$$P(D) = \prod_{k=1}^{K} P(r_k) \tag{2.14}$$

Note that the rewriting rule (2.11) not only specifies lexical translations but also encapsulates nicely the type of reordering involved when translating Chinese verb complexes into English. As a result, searching for the derivation with the maximum probability assignment, as in Equation (2.13), accomplishes the two tasks, constituent translation and sentence reordering (as in Steps 2 and 3 in our generative story) simultaneously. The search can be achieved by chart parsing.

Now, we discuss the way these rewriting rules are extracted from data and how their probabilities are estimated. We take the synchronous grammar proposed in Chiang (2005) as an example. The grammar does not have any underlying linguistic interpretation and uses only one non-terminal *X*. Assume that we have the word-alignment sentence pair, as shown in figure 4. First, we extract initial bi-phrases that are consistent with the word-alignment, as described in Section 2.3. We write these bilingual phrases in the form of synchronous grammar:

$X \rightarrow$ (在 倒塌 的 房屋 里 寻找 生还者, search for survivors in collapsed house)

We then generalize these rules by replacing some substrings with the nonterminal symbol *X*:

$X \rightarrow$ (在 X_1 里 寻找 X_2, search for X_2 in X_1)

where we use subscript indices to indicate which occurrences of *X* are linked by ~. We can see that this rule captures information about both lexical translation and word reordering. Therefore, the learned grammar can be viewed as a significant generalization of the traditional phrase-based models, and can handle longer range word reordering.

To avoid limit the number of rules generated using the process, the rewrite rules are constrained: (a) to contain at least one lexical item but at most 5 words per language, (b) to have no sequences of non-terminals, (c) to have at most two non-terminals, (d) span at most 15 words. Once the rewrite rules are extracted, their probabilities are estimated on the word-aligned sentence pairs using the similar method to that for the phrase-based models. Readers may refer to Chiang (2005) for a detailed description.

References

Brown, P., Della Pietra, S., Della Pietra, V., and Mercer, R. (1993). The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2), pp. 263-311.

Chiang, D., (2005) A Hierarchical Phrase-Based Model for Statistical Machine Translation. *ACL*.

Galley, M., Hopkins, M., Knight, K., Marcu, D., (2004) What's in a translation rule? *HLT-NAACL*, pp. 273-280

Gao, J., Nie, J.Y., Xun, E., Zhang, J., Zhou, M., and Huang, C. (2001). Improving query translation for cross-language information retrieval using statistical models. In *Proceedings of SIGIR Conf.*, pp. 96-104.

Gao, J., Zhou, M., Nie, J.Y., He, H., Chen, W. (2002) Resolving query translation ambiguity using a decaying co-occurrence model and syntactic dependence relations. *SIGIR*, pp. 183-190

Gao, J., Nie, J.Y., Statistical query translation models for cross-language information retrieval, *ACM Transactions on Asian Information Processing (TALIP)*, 5(4): 296-322, 2006.

Koehn, P., Och, F.J., Marcus, D., (2003) Statistical phrase-based translation, In *Proceedings of HLT-NAACL*, pp. 48-54.

Koehn, P. (2009) *Statistical Machine Translation*. *Cambridge University Press*.

Och, F., and Ney, H. (2002) Discriminative Training and Maximum Entropy Models for Statistical Machine Translation. *ACL*, pp. 295-302

Och, F. (2003). Minimum error rate training in statistical machine translation. In *Proceedings of ACL*. pp. 160-67